

**T.C.
AKDENİZ ÜNİVERSİTESİ**



**RADYASYON GÖREVLİLERİ İÇİN RADYASYONDAN KORUNMAYA
YÖNELİK BİREYSEL DOZ TAKİP SİSTEMİNİN DOKU VE ORGANLARA
GÖRE DETAYLI DEĞERLENDİRİLMESİ VE WEB TABANLI İZLEME
SİSTEMİNİN GELİŞTİRİLMESİ**

Gökhan AYÇİÇEK

FEN BİLİMLERİ ENSTİTÜSÜ

FİZİK

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

TEMMUZ 2025

ANTALYA

**T.C.
AKDENİZ ÜNİVERSİTESİ**



**RADYASYON GÖREVLİLERİ İÇİN RADYASYONDAN KORUNMAYA
YÖNELİK BİREYSEL DOZ TAKİP SİSTEMİNİN DOKU VE ORGANLARA
GÖRE DETAYLI DEĞERLENDİRİLMESİ VE WEB TABANLI İZLEME
SİSTEMİNİN GELİŞTİRİLMESİ**

Gökhan AYÇİÇEK

FEN BİLİMLERİ ENSTİTÜSÜ

FİZİK

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

TEMMUZ 2025

ANTALYA

T.C.
AKDENİZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

**RADYASYON GÖREVLİLERİ İÇİN RADYASYONDAN KORUNMAYA
YÖNELİK BİREYSEL DOZ TAKİP SİSTEMİNİN DOKU VE ORGANLARA
GÖRE DETAYLI DEĞERLENDİRİLMESİ VE WEB TABANLI İZLEME
SİSTEMİNİN GELİŞTİRİLMESİ**

Gökhan AYÇİÇEK
FİZİK
ANABİLİM DALI
YÜKSEK LİSANS TEZİ

TEMMUZ 2025

T.C.
AKDENİZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

**RADYASYON GÖREVLİLERİ İÇİN RADYASYONDAN KORUNMAYA
YÖNELİK BİREYSEL DOZ TAKİP SİSTEMİNİN DOKU VE ORGANLARA
GÖRE DETAYLI DEĞERLENDİRİLMESİ VE WEB TABANLI İZLEME
SİSTEMİNİN GELİŞTİRİLMESİ**

Gökhan AYÇİÇEK
FİZİK
ANABİLİM DALI
YÜKSEK LİSANS TEZİ

Bu tez/...../2025 tarihinde jüri tarafından Oybirliği / Oyçokluğu ile kabul edilmiştir.

Prof. Dr. Nına TUNÇEL (Danışman)
Prof. Dr. Melike Behiye YÜCEL
Dr. Öğr. Üyesi Erdi BİLGİÇ

ÖZET

RADYASYON GÖREVLİLERİ İÇİN RADYASYONDAN KORUNMAYA YÖNELİK BİREYSEL DOZ TAKİP SİSTEMİNİN DOKU VE ORGANLARA GÖRE DETAYLI DEĞERLENDİRİLMESİ VE WEB TABANLI İZLEME SİSTEMİNİN GELİŞTİRİLMESİ

Gökhan AYÇİÇEK

Yüksek Lisans, Fizik Anabilim Dalı

Danışman: Prof. Dr. Nına TUNÇEL

Temmuz 2025; 82 sayfa

Radyasyonun kullanımının başlamasıyla birlikte hem yararlı hem de zararlı etkileri ortaya çıkmıştır. Radyasyon nükleer sanayi, endüstri, tıp ve nükleer araştırma gibi alanlarda yaygın olarak kullanır.

Son yıllarda iyonlaştırıcı radyasyonun etkileri çok daha iyi anlaşıldığından, radyasyona maruz kalan bireyleri korumak için radyasyondan korunma sistemi geliştirilmiştir. Bu sistem kapsamında, radyasyon görevlilerinin cep dozimetre verileri kullanılarak veri tabanı oluşturulmuş, bilgi teknolojileri aracılığıyla da bu veri tabanının kayıt ve değerlendirilmesi bireysel izleme sistemleri ile sağlanmıştır.

Bu çalışmada, uluslararası tanımlanan referans düzeyleri baz alınarak varsayımsal veriler eşliğinde bir veri tabanı oluşturulacaktır. Bu veriler, kurum ve bireyin kişisel bilgilerine ek olarak bireylerin 2 aylık periyotlarla 5 senelik cep dozimetre verilerini kapsayacaktır. Yetkili kurum ve kuruluşlar, cep dozimetrelerini kullanıcılara temin eder ve iki aylık periyotlarla gerçekleştirilen okuma işlemleri sonrasında elde edilen verileri bilgi teknolojileri aracılığıyla değerlendirirler. Elde edilen bu veriler kurumlara ve bireylere iletilir. Ancak bireylere ait bazı durumların bu veri tabanı üzerinde işlenmemesi söz konusudur (Senelik izinler, hamilelik durumları vs.). Bu çalışma kapsamında doku ve organlardaki radyasyon doz miktarlarını detaylı hesaplanması amaçlanmaktadır. Doku ve organlardan alınan verilere göre kıyaslanma yapılabileceği için radyasyona maruz kalan bireylerin daha güvenli ve sağlıklı çalışması sağlanacaktır.

Bu çalışma, kullanışlı bir ara yüzle birlikte veri entegrasyonunda kolaylıklar hedeflemekte ve radyasyon görevlilerinin çalışma koşuluna ve bazı özel durumlarına bağlı olarak maruz kaldıkları doz verilerinin değerlendirilmesini kapsamaktadır.

ANAHTAR KELİMELER: İyonizen radyasyon, Radyasyon doz takip sistemi, Radyasyondan korunma

JÜRİ: Prof. Dr. Nına TUNÇEL

Prof. Dr. Melike Behiye YÜCEL

Dr. Öğr. Üyesi Erdi BİLGİÇ

ABSTRACT

DETAILED EVALUATION OF THE INDIVIDUAL DOSE TRACKING SYSTEM FOR RADIATION PROTECTION ACCORDING TO TISSUES AND ORGANS FOR RADIATION OFFICERS AND DEVELOPMENT OF A WEB- BASED MONITORING SYSTEM

Gökhan AYÇİÇEK

MSc Thesis in Physics

Supervisor: Prof. Dr. Nina TUNÇEL

July 2025; 82 pages

With the beginning of the use of radiation, both beneficial and harmful effects have emerged. Radiation is widely used in areas such as nuclear industry, industry, medicine, and nuclear research.

In recent years, since the effects of ionizing radiation have been much better understood, a radiation protection system has been developed to protect individuals exposed to radiation. Within the scope of this system, a database has been created using the pocket dosimeter data of radiation officers, and the recording and evaluation of this database has been provided by individual monitoring systems through information technologies.

In this study, a database will be created with hypothetical data based on internationally defined reference levels. This data will cover the pocket dosimeter data of individuals for 5 years in 2-month periods in addition to the personal information of the institution and the individual. Authorized institutions and organizations provide pocket dosimeters to users and obtain data after reading in 2-month periods and evaluate it through information technologies. This obtained data is transmitted to institutions and individuals. However, it is possible that some situations belonging to individuals are not processed on this database (annual leaves, pregnancy status, etc.). This study aims to circulate the radiation dose amounts in tissues and organs in detail. Since comparisons can be made according to the data obtained from tissues and organs, individuals exposed to radiation will be provided with a safer and healthier working environment.

This study aims to facilitate data integration with a useful interface and includes the evaluation of the dose data to which radiation workers are exposed depending on their working conditions and some special situations.

KEYWORDS: Ionizing radiation, Radiation dose monitoring system, Radiation protection

COMMITTEE: Prof. Dr. Nina TUNÇEL

Prof. Dr. Melike Behiye YÜCEL

Asst. Prof. Dr. Erdi BİLGİÇ

ÖNSÖZ

Bu çalışma, radyasyon görevlilerinin maruz kaldığı iyonlaştırıcı radyasyonun biyolojik etkilerinin daha doğru ve etkin bir biçimde izlenebilmesi amacıyla, bireysel doz takip sistemlerinin doku ve organ düzeyinde ayrıntılı olarak değerlendirilmesini ve buna paralel olarak web tabanlı bir izleme sisteminin geliştirilmesini konu edinmektedir. Özellikle hassas organ ve dokulara yönelik doz analizlerinin yapılması hem çalışan sağlığının korunması hem de ulusal ve uluslararası radyasyondan korunma standartlarının etkin biçimde uygulanması açısından önem arz etmektedir. Bu kapsamda geliştirilen sistemin, bireysel doz takibini daha sistematik, erişilebilir ve sürdürülebilir hale getirerek alana bilimsel ve teknik katkı sağlaması amaçlanmaktadır.

Çalışmalarım sırasında bilgi ve tecrübeleriyle beni yönlendiren, sabırla her türlü desteği veren danışmanım Sayın Prof. Dr. Nına TUNÇEL'e sonsuz saygı, minnet ve teşekkürlerimi sunarım.

Ailemin sabırları ve sonsuz destekleri için teşekkür ederim.

İÇİNDEKİLER

ÖZET.....	i
ABSTRACT.....	ii
ÖNSÖZ.....	iii
AKADEMİK BEYAN	vii
SİMGELER VE KISALTMALAR	viii
ŞEKİLLER DİZİNİ.....	x
ÇİZELGELER DİZİNİ.....	xii
1.GİRİŞ	1
2.KAYNAK TARAMASI.....	3
2.1. Radyasyon.....	3
2.1.1. Radyasyon Türleri.....	3
2.1.2. İyonlaştırıcı Radyasyonun Maddeyle Etkileşimi.....	6
2.1.3. Radyasyon Birimleri.....	6
2.1.4. İyonlaştırıcı Radyasyonun Biyolojik Etkileri.....	8
2.1.5.İyonlaştırıcı Radyasyonun İnsan Üzerine Etkileri ve Sınıflandırması.....	9
2.1.6. Radyasyonun Organlar ve Dokular Üzerindeki Etkileri.....	10
2.2. Radyasyondan Korunma Sistemi.....	11
2.2.1. Etkin doz ve eşdeğer doz kavramları	12
2.2.2. Bireysel doz takip sistemi için örnekler.....	13
2.3. Doz Takip Sistemlerde Kullanılan Yazılım Dili	14
3. MATERYAL VE METOT	17
3.1. Materyal	17

3.2. Metot.....	18
3.2.1. Veritaban Yapısı.....	18
3.2.2. Yazılım Mimarisi.....	18
3.2.3. Kullanıcı Girişi ve Kontroller.....	20
3.2.4. Doz Takibi ve Uyarı Sistemi.....	20
3.2.5. Veri Görselleştirme ve Raporlama Metodolojisi	20
4. BULGULAR.....	22
4.1. Veri tabanı Yapısı ve Veri Yönetimi.....	27
4.2. Bireysel Doz Takibi ve Alara Kontrolü.....	28
4.3. Raporlama ve Veri Dışa Aktarımı.....	29
4.4. Kullanıcı Arayüzü ve Etkileşim	30
4.5. Sistem Performansı ve Genişletilebilirlik.....	31
5. TARTIŞMA.....	32
6. SONUÇLAR	34
7. KAYNAKLAR	35
8. EKLER.....	40
ÖZGEÇMİŞ	

AKADEMİK BEYAN

Yüksek lisans tezi olarak sunduğum “Radyasyon Görevlileri için Radyasyondan Korunmaya Yönelik Bireysel Doz Takip Sisteminin Doku ve Organlara Göre Detaylı Değerlendirilmesi ve Web Tabanlı İzleme Sisteminin Geliştirilmesi” adlı bu çalışmanın, akademik kurallar ve etik değerlere uygun olarak yazıldığını belirtir, bu tez çalışmasında bana ait olmayan tüm bilgilerin kaynağını gösterdiğimi beyan ederim.

...../...../2025

Gökhan AYÇİÇEK

İmzası

SİMGELER VE KISALTMALAR

Simgeler

Bq	: Becquerel
C	: Coulomb
Ci	: Curie
erg	: erg, Enerji birimi
g	: Gram
Gy	: Gray, SI biriminde soğurulan doz
Hp(0,07)	: Yüzeysel (Cilt) Doz Eşdeğeri
Hp(3)	: Göz Merceği Doz Eşdeğeri
Hp(10)	: Derin Doku Doz Eşdeğeri
J	: Joule, enerji birimi
kg	: kilogram
R	: Röntgen, ışınlama birimi
Rad	: Absorbe edilen dozun birimi
rem	: Eşdeğer doz
s	: Saniye
Sv	: Sievert
GBq	: Giga Becquerel
U	: Uranyum
Z	: Atom numarası
α	: Alfa parçacığı
β	: Beta parçacığı
γ	: Gama ışını

Kısaltmalar

ALARA	: As Low As Reasonably Achievable
-------	-----------------------------------

ARS : Acute radiation syndrome

CRI : Cilt Radyasyon Yaralanması

CT : Bilgisayarlı tomografi

DNA : Deoksiriboz Nükleik Asit

DLL: : Dynamic Link Library

EPD : Elektronik Personel Dozimetre

EDOTS : Radat şirketin web tabanlı yazılımı

EURATOM : Avrupa Atom Enerjisi Topluluğu

GDOSE : Brazilian External Occupational Dose Management database system

HSC : Hematopoetik kök hücreler

ILO : Uluslararası Çalışma Örgütü

İSG : İş Sağlığı ve Güvenliği

IAEA : Uluslararası Atom Enerjisi Ajansı

ICRP : Uluslararası Radyasyondan Korunma Komitesi

ICRU : Uluslararası Radyasyon Birimleri ve Ölçümleri Komisyonu

LET : Lineer enerji transferi

MSC : Mezenkimal kök hücre

MSS : Merkezi Sinir Sistemi

NEA : Nükleer Enerji Ajansı

OSL : Optik Uyarmalı Lüminesans

PET : Pozitron Emisyon Tomografisi

RPC : Radiological Protection Committee

SI : Uluslararası Birimler Sistemi

SisPRad : Radyasyon Koruması Yönetimi Yazılımı

TENMAK : Türkiye Enerji, Nükleer ve Maden Araştırma Kurumu

T.C. : Türkiye Cumhuriyeti

TLD : Termo-Lüminesans Dozimetre

UKM : University Kebangsaan Malaysia

UNSCEAR : Birleşmiş Milletler Atomsal Radyasyon Etkileri Bilimsel Komitesi



ŞEKİLLER DİZİNİ

Şekil 2.1. Alfa bozunumu.....	3
Şekil 2.2.Radyum çekirdeği beta bozunumu.....	4
Şekil 2.3.X-ışını yayılım mekanizması.....	5
Şekil 2.4. Plütonyum çekirdeği Gama ışıması.....	5
Şekil 2.5. Radyasyonun sınıflandırılması.....	6
Şekil.2.6.Radyasyonun biyolojik sistemlerdeki doğrudan ve dolaylı etkisi (Hall, 2012).....	9
Şekil 4.1.doz_takip_projesiklasöründe bulunan yazılım kod bileşenlerinin ekran resmi.....	22
Şekil 4.2.Yazılımın db_setup.py bileşeni.....	23
Şekil 4.3. Yazılımın alara.py bileşeni	23
Şekil 4.4. Yazılımın data_operations.py bileşeni.....	24
Şekil 4.5. Yazılımın institution_operations.py bileşeni.....	24
Şekil 4.6. Yazılımın individual_operations.py bileşeni.....	25
Şekil 4.7. Yazılımın dose_operations.py bileşeni.....	25
Şekil 4.8. Yazılımın report.py bileşeni	26
Şekil 4.9. Yazılımın populate_data.py bileşeni	26
Şekil 4.10. Yazılımın main.py bileşeni.....	27
Şekil 4.11. Yazılımın radyasyon_doz_raporu_olusturucu.py bileşeni.....	27
Şekil 4.12. Radyasyon doz takip python sisteminin terminal resmi.....	28
Şekil 4.13. Python yazılım kodundan Excel formatında çıktı görünümü.....	29
Şekil 4.14. 2019 yılına göre bireylerin radyasyon doz dağılımının grafiksel görseli.....	30
Şekil 4.15. Looker Studio' dan örnek raporlama penceresi.....	30
Şekil 4 16. Doz aşımına uğramış örnek sanal çalışan raporu.....	31

Şekil 4.17. Looker Studio’ da parametre değişikliği ile oluşan yeni rapor.....31



ÇİZELGELER DİZİNİ

Çizelge 2.1. Radyasyon birimleri tablosu	8
Çizelge 2.2. Radyasyon çalışanları ve toplum üyesi kişiler için doz sınırları (Radyasyon Güvenliği Yönetmeliği, 2010).....	13



1. GİRİŞ

Radyasyonun çeşitli alanlarda kullanılmaya başlanmasıyla birlikte, iyonlaştırıcı radyasyonun insan sağlığı üzerindeki zararlı etkileri de zamanla ortaya çıkmıştır. Radyasyondan sağlanan faydaların yanı sıra, bu kaynağın potansiyel risklerine karşı korunma yöntemlerinin geliştirilmesi ve uygulanması bir zorunluluk haline gelmiştir. Geçmişte meydana gelen ve hem çevresel hem de toplumsal düzeyde geniş etkiler yaratan çok sayıda radyasyon kazası, radyasyona maruziyetin uzun vadeli biyolojik ve ekolojik sonuçlarını açıkça ortaya koymuştur. Bu durum, radyasyondan korunma biliminin önemini ve bireysel doz takibinin gerekliliğini bir kez daha vurgulamaktadır.

İyonlaştırıcı radyasyondan günümüzde birçok farklı sektörde yararlanılabilmektedir. Mesleki olarak fiilen iyonlaştırıcı radyasyon ile çalışanlar, radyasyon görevlisi olarak adlandırılmaktadır. Bu görevlilerin aldığı radyasyon doz düzeyleri uluslararası ve ulusal birçok organizasyon tarafından belirlenmektedir (ICRP, 2007 ve IAEA, 1996 ve ICRU, 1980 ve ICRU, 2011 ve EURATOM, 2014 vb.).

Radyasyondan korunma sisteminin ana hedefi, radyasyon görevlilerinin, halkın ve çevrenin radyasyonun zararlı etkilerine karşı korunmasıdır. Amacı ise radyasyonun deterministik etkilerini (cilt yanıkları, katarakt, ölüm gibi) önlemek ve stokastik etkilerin (kanser ve genetik etkiler) meydana gelme olasılığını en aza indirmektir. Görevlilerin maruz kaldığı dozlar, tüm vücut ve organ dozları olarak değerlendirmeye alınmaktadır (Euratom, 2013). Radyasyon yayan cihaz veya radyoaktif maddelerle çalışan kişilerin maruz kaldığı bu radyasyon dozunu ölçmeye yarayan cihazlara kişisel dozimetre adı verilir. Maruz kalınan radyasyon düzeyini ölçmek, karşılaştırmak ve maruziyet sonucu oluşabilecek zararlı biyolojik etkinin tahmininde kullanılmak üzere, yıllar içinde tanımlanan bazı radyasyon doz birimleri oluşturulmuştur (Soğurulan Doz, Eşdeğer Doz, Etkin Doz) (ICRU, 1980 ve ICRU, 2011).

Etkin doz ve eşdeğer doz kavramları, doğrudan ölçülebilir değildir. ICRU küresinde tanımlanan özel bir derinlik için bir yetişkinin etkin dozunun yaklaşık değeri belirlenmektedir Hp(d) olarak tanımlanan bu nicelik, bireysel izleme için kullanılır ve cilt için $d=0,07$ mm, göz merceği için $d=3$ mm, derin yerleşimli organlar için $d=10$ mm derinlikte olmak üzere sırasıyla Hp(0,07), Hp(3), Hp(10) tanımlamaları yapılmaktadır. Bu kapsamda dozimetreler, kişisel doz eşdeğeri Hp(10) ölçmek için kalibre edilirler (Griffith, 1990 ve Mattsson, 2013). 24.03.2000 tarih ve 23999 sayılı Resmi Gazete’de yayımlanan Radyasyon Güvenliği Yönetmeliğinin 20-21(Değişik: RG-3/6/2010-27600). Maddesi gereğince, çalışma koşulu A durumunda (yılda 6 mSv’den daha fazla etkin doza veya göz merceği, cilt, el ve ayaklar için yıllık eşdeğer doz sınırlarının 3/10’undan daha fazla doza maruz kalma olasılığı bulunan) görev yapan kişilerin, kişisel dozimetre kullanması zorunludur. Kullanılan dozimetreler; pasif dozimetreler (Film, TLD, OSL) ve aktif dozimetreler (Kalem, EPD-Elektronik Personel Dozimetre) olarak ikiye ayrılır (Radyasyon Güvenliği Yönetmeliği, (Değişik: RG-3/6/2010-27600)).

Bu çalışma, kullanıcı dostu bir arayüz ile veri entegrasyonunu kolaylaştırmayı hedeflemiş ve radyasyon görevlilerinin çalışma koşulları ile bazı özel durumlarına bağlı olarak maruz kaldıkları doz verilerinin değerlendirilmesini kapsamıştır. Bu kapsamda, Python programlama dili kullanılarak web tabanlı bir yazılım geliştirilmiş; birey ve kuruma ait tüm doz verileri dijital ortamda kayıt altına alınmıştır. Ayrıca, ilgili doku ve organlarda biriken radyasyon miktarları hesaplanmış ve bu değerler, normal koşullarda alınması gereken doz limitleriyle karşılaştırılarak analiz edilmiştir. Yazılımda yer alan program modülleri, Dinamik Bağlantı Kitaplığı (DLL: Dynamic Link Library) dosyaları

şeklinde yapılandırılmış; böylece bu bileşenlerin farklı programlama dillerine dönüştürülerek çeşitli yazılım platformlarında kullanılabilirliği mümkün hâle getirilmiştir.

Bu tez çalışmasıyla, bireylerin maruziyet verilerinin izinli gün sayıları, tanısal amaçlı yapılan tetkikler ve olası radyasyon kazaları sonucu alınan ek dozlar dikkate alınarak değerlendirilmesine imkân tanınmıştır. Belirtilen bu unsurlar doğrultusunda geliştirilen yazılımın, mevcut sistemlere kıyasla daha işlevsel olabileceği öngörülmüştür. Ayrıca, elde edilen veri tabanı sayesinde yalnızca veri özetlerinin raporlanması değil, aynı zamanda istatistiksel ve epidemiyolojik araştırmaların yapılmasına da olanak sağlanmıştır. Bu yönüyle çalışma, ileride gerçekleştirilecek benzer araştırmalara literatür kaynağı oluşturabilecek nitelikte bir temel sunmuştur.



2. KAYNAK TARAMASI

2.1. Radyasyon

Radyasyon bir enerji çeşidi olup bu enerjinin elektromanyetik dalgalar veya parçacıklar şeklinde yayılması olayıdır. Bu enerji çeşidinin farklı türleri ve kaynakları vardır. Doğada kendiliğinden var olan doğal radyasyon kaynakları, örneğin; Kozmik ışınlar, radon gazı, doğal radyoaktif izotoplar vb. verilebilir. Bunun yanı sıra insan yapımı (yapay) radyasyon kaynakları da vardır. Bunlar tıbbi cihazlar, nükleer enerji santralleri, sanayi gibi birçok alanda kullanılmak için tasarlanmaktadır. Kullanım alanları genişledikçe korunmanın yollarını da geliştirmek gerekmiştir.

2.1.1. Radyasyon Türleri

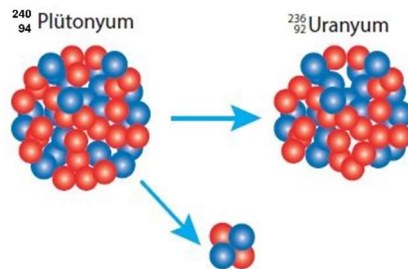
Radyasyon, yüksek hızlı atom altı parçacıklar ve boşlukta ışık hızında hareket eden enerji paketleri dalgaları şeklinde ortaya çıkar. Bir radyoaktif atom, fazla enerjisini bu dalga ve parçacıklar yoluyla yayarak atar. Parçacık türü radyasyon alfa, beta, proton ve nötron parçacıklarını içerirken, dalga türü radyasyon Gama ışınları ve X-ışınlarını kapsar. Her radyasyon türü farklı özelliklere sahiptir ve korunma yolları da birbirinden farklıdır.

2.1.1.1. Parçacık tipi radyasyon

Parçacık radyasyonu, belli bir kütle ve yüksek enerjiye sahip ışık hızıyla boşlukta hareket eden parçacıklarla meydana gelmektedir. Bu parçacıkların davranışları boyutlarına, taşıdıkları enerjiye ve sahip oldukları negatif veya pozitif elektrik yüklerine bağlıdır (Tanır, 2013).

- **Alfa parçacıkları**

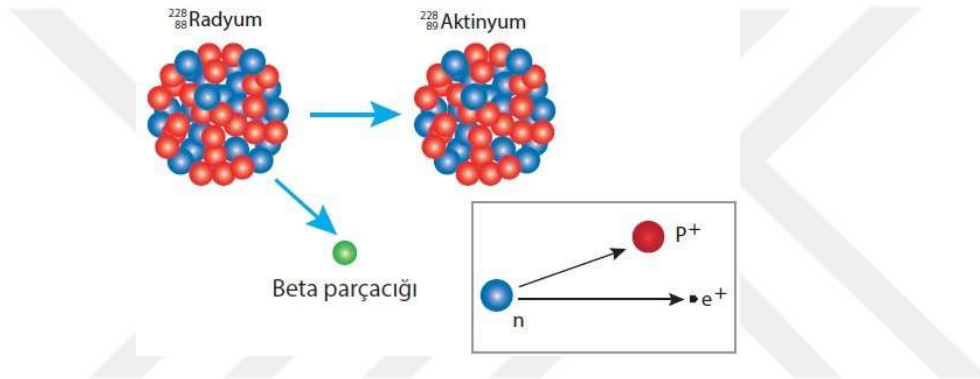
Alfa radyasyonunda, kararsız çekirdekler daha kararlı hale gelmek için ağır ve pozitif yüklü parçacıklar olan alfa parçacıklarını salar. Bu parçacıklar, iki proton ve iki nötron içerir. Alfa parçacıkları yüksek enerjili olmalarına rağmen, ağırlıkları nedeniyle enerjilerini kısa mesafelerde aktarırlar. Havada yalnızca birkaç santimetre ilerleyebilirler ve bir kâğıt parçası veya insan cildinin dış tabakası tarafından kolayca durdurulabilirler. Ancak alfa parçacıkları, yutma, soluma veya bir yara yoluyla vücuda girdiğinde tehlikeli hale gelir. İç dokulara doğrudan zarar verebilirler ve hücrelerde ciddi hasara neden olabilirler. Bu zarar, alfa parçacıklarının neden olduğu iyonlaşmaların birbirine çok yakın olmasından ve tüm enerjilerini birkaç hücreye aktarmalarından kaynaklanır. Bu, hücrelere ve DNA'ya ciddi hasar verebilir.



Şekil2.1. Alfa bozunumu

- **Beta parçacıkları**

Beta radyasyonu, genellikle eksi yüklü elektronlardan oluşur, ancak nadiren artı yüklü pozitronlardan da oluşabilir. Elektronlar eksi bir (-1) yüke, pozitronlar ise artı bir (+1) yüke sahiptir. Kütlesi ve yükü alfa parçacıklarından daha az olduğu için, beta radyasyonu daha yüksek nüfuz etme yeteneğine sahiptir. Beta parçacıkları insan cildini geçebilir ancak önemli organlara ulaşamaz. İnce bir alüminyum plaka beta parçacıklarını durdurmak için yeterlidir. Beta parçacıkları alfa parçacıklarından daha nüfuz edici olmakla birlikte, canlı doku ve DNA'ya daha az zarar verirler çünkü iyonlaşmalar daha uzak mesafelerde oluşur. Havada daha fazla ilerlerler ancak plastik, kıyafet veya birkaç mm kalınlığında alüminyum tabakayla durdurulabilirler. Engellenmezlerse sadece birkaç metre ilerleyebilirler. Beta parçacıkları birkaç mm'ye kadar cilde nüfuz edebilir ve radyasyon yanıkları gibi hasarlara neden olabilir. Solunmaları, yutulmaları veya yaralanma yoluyla vücuda alınmaları ciddi sağlık etkilerine yol açabilir.



Şekil 2.2. Radium çekirdeği beta bozunumu

- **Protonlar**

Protonlar, atom çekirdeğinde bulunan, pozitif yüke sahip atom altı parçacıklardır. Bir protonun kütlesi, bir nötronunkine yakındır, ancak elektronlardan çok daha ağırdır. Protonlar, atomun kimyasal özelliklerini belirleyen temel bileşenlerden biridir. Her elementin atom çekirdeğindeki proton sayısı, o elementin atom numarasını belirler ve periyodik tabloda hangi element olduğunu gösterir. Protonlar, kuark adı verilen daha küçük parçacıklardan oluşur. Üç kuarktan meydana gelen bir proton, güçlü nükleer kuvvetle bir arada tutulur.

- **Nötronlar**

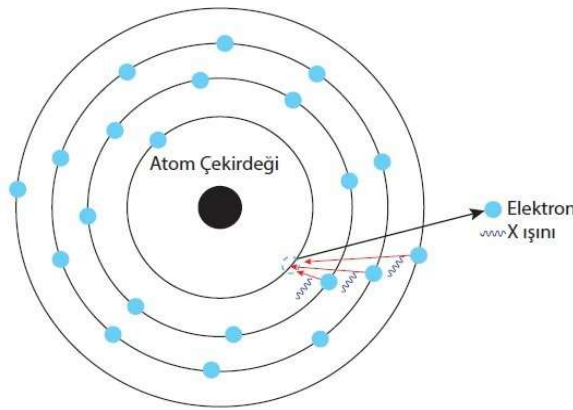
Nötronlar, çekirdeğin ana bileşenlerinden olup, yüksüz ve büyük parçacıklardır. Doğrudan iyonizasyon yapmasalar da madde ile etkileşimleri alfa, beta, Gama veya X-ışınları üreterek dolaylı iyonizasyona neden olabilir. Nötronlar yüksek nüfuz etme yeteneğine sahiptir ve sadece kalın beton, su veya parafin ile durdurulabilir. Nükleer reaktörler veya hızlandırıcılar gibi çeşitli yöntemlerle üretilirler ve dolaylı iyonlaştırıcı radyasyon kaynağı olarak önemli bir rol oynarlar. Nötron radyasyonu, nükleer tepkimeler sonucu çekirdekten yayılan parçacıklardır. Atom çekirdeğiyle çarpıştıklarında enerjilerini kaybederler. Bu nedenle nüfuz etme güçleri yüksektir. Nötron radyasyonunu azaltmak için su gibi nötronlarla iyi etkileşime giren malzemeler kullanılmalıdır.

2.1.1.2. Dalga Tipi Radyasyon

Elektromanyetik radyasyon, belirli bir enerjiye sahip ancak kütlesiz radyasyon çeşididir. Gama ışınları ve X-ışınları, elektromanyetik dalgalar olup, fotonlar şeklinde enerji paketleri olarak bulunur. Bu ışınlar maddeye nüfuz edebildikleri için nüfuz edici radyasyon olarak bilinir. X-ışınlarını ve Gama ışınlarını soğurmak için en iyi malzeme kurşundur (Tanır, 2013).

- **X-ışınları**

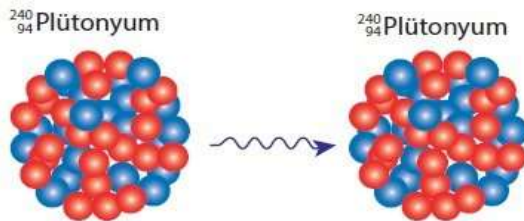
X-ışınları, bir atomun elektron enerji seviyelerinde düzensizlik oluştuğunda yayımlanan elektromanyetik radyasyondur. Bu düzensizlik, çekirdeğe yakın seviyelerden elektron kopması veya çekirdek yakınındaki seviyelerden elektron yakalanması ile meydana gelir. Elektron bulutundaki boşluklar diğer elektronlarla doldurulduğunda X-ışınları ortaya çıkar. X-ışınları enerjisini Compton saçılması ve fotoelektrik olaylar yoluyla bırakır. X-ışınları ve Gama ışınları benzer özelliklere sahiptir ancak farklı yerlerden kaynaklanır: X-ışınları çekirdek dışından, Gama ışınları ise çekirdek içinden salınır. X-ışınları elektrikle çalışan cihazlar tarafından üretilebilir ve Gama ışınları gibi sağlık üzerinde ciddi etkileri vardır.



Şekil 2.3. X-ışını yayılım mekanizması

- **Gama ışınları**

Gama ışınları, kanser tedavisi gibi çeşitli tıbbi uygulamalarda kullanılan elektromanyetik radyasyondur. X-ışınlarına benzerler ve bazıları vücuttan zararsız geçerken, bazıları emilip hasara neden olabilir. Gama ışınlarının zararlı etkilerini azaltmak için kalın beton veya kurşun duvarlar kullanılabilir. Bu nedenle radyoterapi odalarının duvarları çok kalındır.

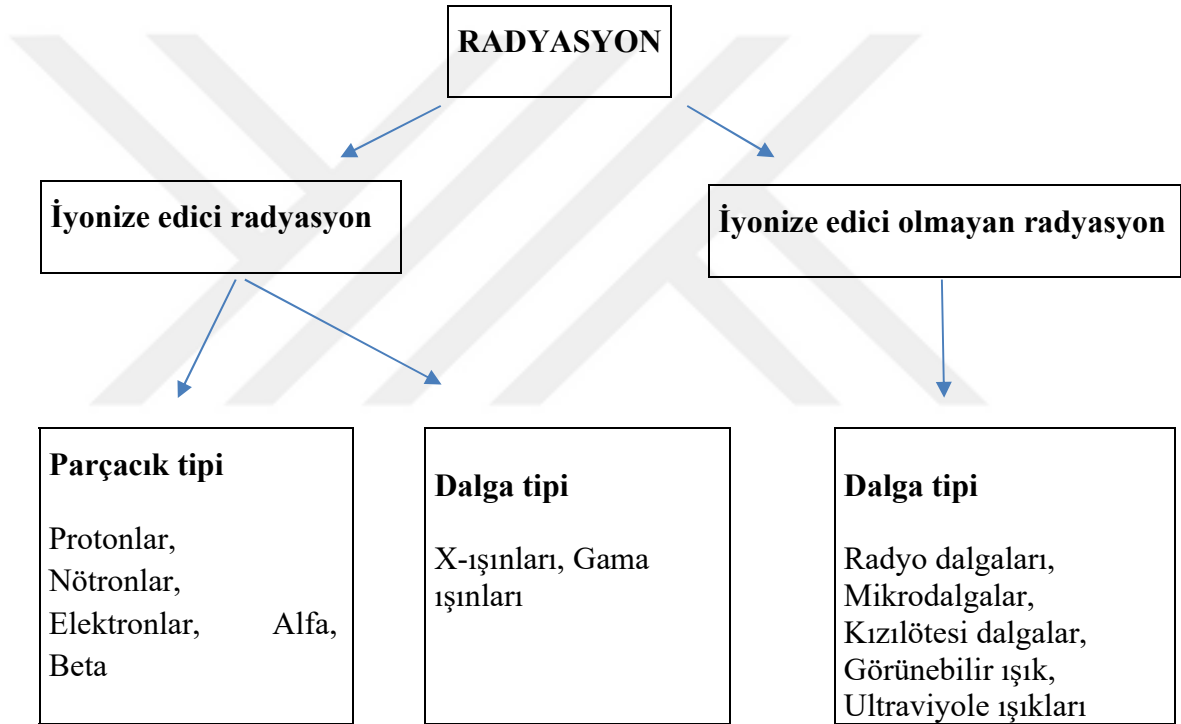


Şekil 2.4. Plütönyum çekirdeği Gama ışıması

Gama radyasyonu, radyoaktif çekirdek tarafından yayılan yüksek enerjili elektromanyetik radyasyondur. Radyoaktif çekirdek alfa veya beta bozunması yaparken enerjisi yüksek kalırsa, Gama radyasyonu yayarak enerji seviyesini düşürür. Bu süreçte çekirdeğin proton ve nötron sayısı değişmez. Gama radyasyonu, Compton saçılması, çift oluşumu ve fotoelektrik olay gibi etkileşimlerle enerjisini bırakır. Yüksek nüfuz etme gücüne sahiptir ve etkilerini azaltmak için kurşun plakalar kullanılabilir.

2.1.2.İyonlaştırıcı Radyasyonun Madde ile Etkileşimi

Radyasyonun maddeyle etkileşimi çeşitli şekilde gerçekleşir. Temel olarak bu enerjinin maddede iyonlaşmayı neden olması ele aldığımızda ikiye ayırabiliriz. Bu kapsamda şekil 2.1’de gösterildiği gibi iyonize edici radyasyon ve iyonize edici olmayan radyasyon olarak sınıflandırılır.



Şekil 2.5.Radyasyonun sınıflandırılması

İyonlaştırıcı radyasyon, madde ile etkileşime girdiğinde atomlardan elektron kopararak iyonlar oluşturma yeteneğine sahip yüksek enerjili bir radyasyon türüdür. Bu radyasyon, hem elektromanyetik radyasyonu (X-ışınları ve Gama ışınları gibi) hem de parçacık radyasyonunu (alfa, beta, proton ve nötron radyasyonu gibi) kapsar. Canlı hücrelerdeki atomları etkileyebilecek ve dolayısıyla DNA’ya zarar verebilecek kadar enerji taşır.

2.1.3. Radyasyon Birimleri

Radyasyon birimleri, radyasyonun miktarını ve etkilerini ölçmek için kullanılır. Bu birimler, farklı radyasyon türleri ve dozları için çeşitli ölçümleri ifade eder.

Radyasyon yayıldığı zaman birçok etkileşim meydana getirir ve ortam içerisinde enerji depolanır. Bu depolanan enerji radyasyon dozu olarak tanımlanır ve bu etkileşimler canlı dokusu içinde meydana gelirse biyolojik değişiklikler ile sonuçlanır. Radyasyon dozuna maruz kalmayı belirlemek bu etkileşimleri anlamada, birimler ise etkileşimleri tanımlamada yol göstermektedir (Tanır, 2013).

Radyasyon ölçülmesinde kullanılan terimler yıllar boyunca çeşitli değişikliklere uğramış, yeni birimler ortaya çıkmış ve bazıları da artık kullanılmaz olmuştur. 1973 yılında “Uluslararası Birimler Sisteminin (SI)” kurulması üzerine, Uluslararası Radyasyon Birimleri ve Ölçme Komisyonu (ICRU) yeni radyasyon birimlerini belirlemiştir. 1976 yılında yürürlüğe giren yeni radyasyon birimleri 1986 yılından bu yana kullanılmaktadır. ICRU tarafından radyasyon çalışmalarında kullanılan ışınlama dozu, aktivite, soğurulma dozu ve doz eşdeğeri birimleri Çizelge 2.1’de tanımlanmıştır (ICRU, 2011).

Aktivite: radyoaktif bir maddenin saniyedeki bozunum sayısıdır. Yeni birimi Becquerel ve eski birimi Curie’ dir. 1 gram Radiumun aktivitesi 1 Ci’ dir ve 1 gram Radiumun saniyede $3,7 \times 10^{10}$ bozunuma uğramasıyla $3,7 \times 10^{10}$ Bq ortaya çıkmaktadır.

$$1 \text{ Ci} = 3,7 \times 10^{10} \text{ bozunum/saniye} = 3,7 \times 10^{10} \text{ Bq}$$

$$1 \text{ Bq} = 2,703 \times 10^{-11} \text{ Ci}$$

İşinlama Dozu: X veya Gama ışınlarının, normal hava koşullarında (0°C, 760 mm Hg), 1kg havada oluşturduğu iyon miktarıdır. Yeni birimi Coulomb/kg, eski birimi Röntgen’ dir.

$$1 \text{ R} = 2,58 \times 10^{-4} \text{ Coulomb/kg}$$

Soğurulma Dozu: Radyasyona maruz kalan 1 gram maddenin, 100 erg’ lik enerjiyi soğurmasına 1 rad denir. SI birim sisteminde, soğurulan radyasyon dozu birimi Gray’ dir. Işınlanan bir maddenin kilogram başına 1 Joule’lük enerji soğurmasına Gray denir ve Gy ile sembolize edilir.

$$1 \text{ rad} = 100 \text{ erg /g}$$

$$100 \text{ erg / g} = 10^{-2} \text{ Joule/kg}$$

$$10^{-2} \text{ Joule/kg} = 1 \text{ Gray}$$

$$1 \text{ Gray} = 100 \text{ rad}$$

Eşdeğer Doz: Radyasyon dozunun birimleri olan rad ve Gray radyasyonun canlı sistemleri üzerindeki etkisini ifade etmekte yetersizdir. Zira soğurulan dozun meydana getirdiği zararlı biyolojik etkiler hem iyonizasyon yoğunluğuna hem de radyasyonun türüne (α , β , γ vs.) göre değişiklik göstermektedir. Doz-eşdeğeri hesaplamalarında, soğurulan doz ile radyasyonun türüne bağlı “etkinlik faktörünün” çarpımından yararlanılmaktadır. Bu yorum çerçevesinde kısaca “insan için eşdeğer radyasyon” sözcüklerinin İngilizcedeki karşılığı olan “radiation equivalent for man” sözcüklerinin baş harflerinden oluşan ve “rem” olarak ifade edilen doz eşdeğeri birimi ortaya atılmıştır. Bir rem dozunun hesaplanmasında, soğurulmuş radyasyon (rad) ile radyasyonun türüne

ilişkin zarar etkinliği faktörünün çarpımıdır. Günümüzde SI Birim Sisteminde rem yerine Sievert kullanılmaktadır.

$$1\text{Sv}=100\text{ rem}$$

Çizelge 2.1.Radyasyon birimleri tablosu

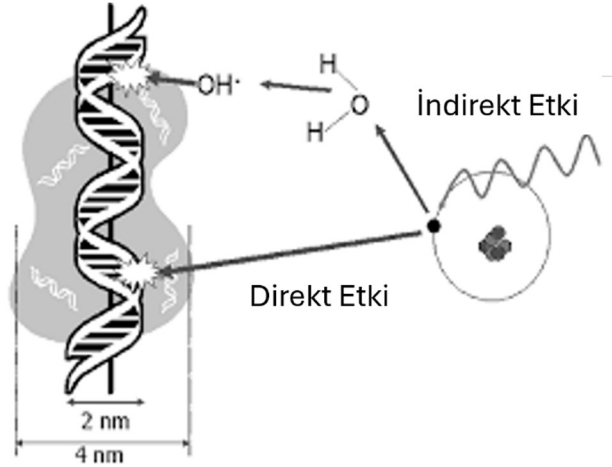
Terim	Eski Birim	Yeni Birim	Dönüşüm
Aktivite	Curie (Ci)	Becquerel (Bq)	$1\text{Ci}=3,7\times 10^{10}\text{ Bq}$ $1\text{Ci}=37\text{ GBq}$
Işınlama Dozu	Röntgen (R)	Coulomb/Kilogram (C/kg)	$1\text{C/kg}= 3876\text{ R}$ $1\text{R}=2,58\times 10^{-4}\text{ C/kg}$
Soğurulma Dozu	Radiation Absorbed Dose (rad)	Gray (Gy)	$1\text{Gy}=100\text{ rad}$ $1\text{rad}=0,01\text{ Gy}$
Doz Eşdeğeri	Röntgen Equivalent Man (rem)	Sievert (Sv)	$1\text{Sv}=100\text{ rem}$ $1\text{rem}=0,01\text{ Sv}$

Çizelge 2.1’de, radyasyonun farklı yönlerini ve etkilerini ölçmek için kullanılan birimler ve semboller açıklanmıştır. Gray ve Sievert, modern birimlerde en yaygın kullanılanlardır, becquerel ise radyoaktiviteyi ölçmek için standart bir birimdir. Rad ve rem gibi eski birimler ise hala bazı alanlarda kullanılmaktadır (Tanır, 2013).

2.1.4. İyonlaştırıcı radyasyonun biyolojik etkileri

Radyasyonun biyolojik sistemler üzerindeki etkileri, maruz kalınan radyasyon türüne ve dozuna bağlıdır. İyonlaştırıcı radyasyona maruz kalma, madde tarafından emilen enerji miktarıyla ölçülür ve bu miktar, kilogram başına joule (J/kg) olarak Gray (Gy) biriminde ifade edilir. Ancak bu ifade radyobiyolojide yetersiz kalmaktadır. Bunun yerine lineer enerji transferi (LET) kavramı daha doğru sonuç vermektedir. LET iyonize edici radyasyonun bir maddenin birim mesafesi boyunca aktardığı enerji miktarını ifade etmektedir. Bu kavram, radyasyonun biyolojik etkilerini anlamak için oldukça önemlidir (Lowe, 2022).

İyonlaştırıcı radyasyonun madde ile etkileşimi, biyolojik dokularda doğrudan ve dolaylı etkiler yaratabilir. Doğrudan etkiler, hücresel bileşenleri (örneğin, DNA’yı) doğrudan etkileyerek genetik hasara neden olabilir. Dolaylı etkiler ise radyasyonun su moleküllerini iyonize ederek serbest radikaller oluşturmaları sonucunda ortaya çıkar ve bu radikaller, hücresel yapıları ve DNA’yı dolaylı olarak hasara uğratabilir (UNSCEAR 2008, The Chernobyl Forum: 2003-2005).



Şekil 2.6. Radyasyonun biyolojik sistemlerdeki doğrudan ve dolaylı etkisi (Hall, 2012)

Radyasyonun hücre üzerindeki etkisi radyasyon çeşidine, maruz kalınan süreye ve dozuna bağlıdır. Bunu da LET ile ifade ederiz. Yüksek LET, bir maddenin içinden geçerken daha fazla enerji transfer eden radyasyon türlerini ifade ederken, düşük LET, daha az enerji transfer eden radyasyon türlerini belirtir.

X-ışınları, Gama ışınları ve beta parçacıkları gibi düşük LET'e sahip radyasyonlar, madde içerisinde uzun mesafelerde yayılır ve daha az enerji transferi gerçekleştirir. Alfa parçacıkları ve ağır yüklü parçacıklar gibi yüksek LET'e sahip radyasyonlar ise daha kısa mesafede yol alır fakat yoğun miktarda enerji bırakırlar (Lowe, 2022).

Biyolojik etkiler, alınan doz ve maruziyet süresine bağlı olarak akut (ani) veya kronik (uzun vadeli) olabilir. Akut etkiler, yüksek dozda radyasyona kısa süreli maruziyet sonucunda radyasyon hastalığına, doku ölümüne ve organ hasarına yol açabilir. ARS (Akut Radyasyon Sendromu), genellikle tüm vücudun ya da önemli bir kısmının etkilenmesi durumunda ortaya çıkar ve belirtileri kusma, mide bulantısı, iştahsızlık gibi klasik semptomlarla başlar. Bu semptomlar maruziyetten kısa bir süre sonra ortaya çıkabilir ve birkaç gün sürebilir. Kronik etkiler ise düşük dozlarda uzun süreli maruziyet sonucunda kanser, genetik mutasyonlar ve diğer uzun vadeli sağlık sorunlarına neden olabilir (UNSCEAR 2008, The Chernobyl Forum: 2003–2005).

2.1.5. İyonlaştırıcı Radyasyonun İnsan Üzerine Etkileri ve Sınıflandırması

İnsanların iyonlaştırıcı radyasyona maruz kaldıklarında oluşabilecek etkiler ise aşağıdaki gibi sınıflandırılmıştır.

Erken etkiler; radyasyona maruz kalınır kalınmaz oluşan etkilerdir ve deterministik (belirli) etkiler olarak adlandırılır. Deterministik etkiyle sonuçlanan radyasyon ışınlanmasının eşik doz seviyesi, insanlar için yaklaşık 250 mSv civarındadır. Bu eşik dozun üstünde, doz miktarlarına bağlı olarak farklı biyolojik reaksiyonlar oluşmaktadır. Doz miktarı arttıkça etkilerin şiddeti de artmaktadır. Neyse ki yüksek dozda radyasyon ışınlanması olan kazalar oldukça azdır ve yüksek dozda radyasyon almış kişiler için tıbbi tedavi imkanları oldukça gelişmiştir ve gelişmeye de devam etmektedir.

Gecikmiş etkiler; etkileri yıllar sonra açığa çıkar ve stokastik (olası) etkiler olarak adlandırılır. Stokastik etkilerin kesin olarak oluşması söz konusu değildir, fakat ışınlama miktarı arttıkça oluşum olasılığı da artmaktadır. Stokastik etkilerin en önemli çeşidi kanserdir (lösemi-kan kanseri). Teorik olarak üreme hücreleri ışınlamaya maruz kaldığında genetik bozukluklar olabilir. Ancak, Hiroşima, Nagasaki ve Çernobil’de meydana gelen olaylar sonrası hayatta kalan insanlar üzerinde yapılan çalışmalarda herhangi bir genetik bozukluk gözlenmemiştir (Kiefer, 2012).

Radyasyonun etkileri, maruz kalınan doz ve süreye bağlı olarak değişir. Yüksek dozlar ölümcül olabilirken, düşük dozlar uzun vadede kanser ve kalıtsal hastalık riskini artırabilir. Gebelik döneminde radyasyondan kaçınmak hem annenin hem de fetüsün sağlığı için kritiktir. Deterministik etkiler belirli bir eşikten sonra ortaya çıkarken, stokastik etkiler dozla doğru orantılı olarak artar ve bu etkilerin kesin olarak radyasyonla ilişkili olup olmadığını belirlemek zordur (IAEA, 2004).

2.1.6.Radyasyonun Organlar ve Dokular Üzerindeki Etkileri

Radyasyon hasarlarına karşı tepkiler, ışınlanan organ/dokuların radyasyon duyarlılıkları ve bu organ/dokuların vücutta gördükleri fonksiyonlara bağlıdır. Beyin dahil olmak üzere bütün sinir dokusu ve kalp dahil bütün kas dokusu genel olarak radyasyona karşı duyarlı değildirler. Bu dokulara öldürücü dozlar verilmesinden sonra ufak hemorajiler (kanamalar) dışında önemli bir etki görülmemiştir. Bir doku veya organ üzerindeki radyasyon hasarı dokunun veya organın meydana getirdiği ürünlerin (hormonlar, özel hücreler, enzimler, v.b.) artmasına veya azalmasına, dokunun veya organın büyümesinde aksaklıklara veya ölümüne sebep olabilir. İyonlaştırıcı radyasyon, solunum sistemi, sinir sistemi, hematopoietik sistem, gastrointestinal sistem ve cilt üzerinde önemli biyolojik hasarlara neden olabilir (Hall, 2012).

Solunum yoluyla alınan radyasyon, sistem üzerindeki ciddi etkilere neden olup alınan radyasyon miktarı, maruz kalma süresi ve radyasyon türüne bağlı olarak değişkenlik gösterir. Olası sonuçlar arasında ARS, radyasyon pnömonisi, artan akciğer kanseri riski, solunum yolu enfeksiyonları ve uzun vadeli solunum sorunları bulunmaktadır. Kronik radyasyon maruziyetinin solunum sistemi ve bronkopulmoner hastalıklar, özellikle kronik obstrüktif akciğer hastalığı üzerinde uzun vadeli etkileri olabilir. Semptomlar arasında kronik öksürük, nefes darlığı ve azalmış akciğer kapasitesi bulunur. Bu etkiler maruziyetten yıllar sonra bile devam edebilir ve yaşam kalitesini önemli ölçüde etkileyebilir (Talapko vd., 2024).

Son yıllarda sinir sisteminin düşük radyasyon dozlarına bile yanıt verdiğine ve bu reaksiyona sıklıkla bir miktar hasarın eşlik ettiğine dair artan kanıtlar bulunmaktadır. MSS (merkezi sinir sistemi), diğer doku tiplerine göre metabolik strese karşı daha savunmasızdır ve iyonlaştırıcı radyasyon parçacıkları, doğrudan veya dolaylı DNA hasarına neden olabilecek serbest radikaller üretme fiziksel kapasitesine sahiptir. Radyasyon, vücutta oksidatif stres adı verilen bir duruma yol açar. Bu stres, beyin hücrelerini etkileyen iltihaplanma süreçlerini tetikler ve mikrogliya adı verilen hücrelerin sayısını artırır. Mikrogliya, beyin için zararlı etkiler yapabilen hücrelerdir. Ayrıca, radyasyon damarların iç yüzeyindeki hücrelere zarar verebilir ve bu da damarların tıkanmasına veya pıhtı oluşumuna yol açabilir. Bu tür etkiler, zamanla damar sertliği ve küçük damar hastalıklarına neden olabilir, bu da damarların işlevini bozarak kalp krizi gibi ciddi sağlık sorunlarına yol açabilmektedir. Belirtiler geçici veya kalıcı olabilir ve

radasyon hasarı akut veya gecikmiş olabilmektedir. Akut ve kronik MSS radyasyonu, DNA hasarı, oksidatif stres, hücre ölümü ve sistemik inflamasyon gibi karmaşık bir moleküler ve hücrel değişiklik ağına neden olur. Bu değişiklikler sinaptik esnekliği ve nöronal yapıyı etkileyerek davranış ve biliş etkilemektedir.

Hematopoietik sendrom, 1-8 Gy radyasyon dozu alındığında ortaya çıkan bir durumdur. Bu sendrom, kemik iliği ve lenf dokusundaki kök hücrelerin zarar görmesiyle gelişir. Düşük doz radyasyon bile kan hücrelerinin sayısında hafif bir azalmaya neden olabilir. Belirgin belirtiler arasında kan hücrelerinin, özellikle granülositlerin ve trombositlerin azalması ve bağışıklık sisteminin zayıflaması bulunur. Kemik iliğinde, kan hücrelerinin yerini yağ hücreleri almaya başlar, bu da kemiğin yapısını ve kalitesini bozar. Kemik iliği, iki tür kök hücre içerir: hematopoietik kök hücreler (HSC) ve mezenkimal kök hücreler (MSC). HSC' ler kan hücrelerini üretirken, MSC' ler diğer doku türlerini üretir. Radyasyon, bu kök hücrelerin işlevini bozarak kan hücrelerinin üretiminde sorunlara yol açar ve çeşitli kan hastalıklarına neden olabilir. Hematopoietik sendrom, bağışıklık sistemini ciddi şekilde etkiler ve anemi, bağışıklık yetersizliği gibi sorunlara yol açar. Bu sendrom, genellikle diğer radyasyon etkileriyle birlikte görülür ve tedavi edilmezse ölümle sonuçlanabilmektedir.

Radyasyonun gastrointestinal sistem üzerindeki etkileri, büyük dozlarda (6-15 Gy) meydana gelen bağırsak disfonksiyonu ile kendini gösterir. Radyasyon sindirim sistemindeki hücreleri etkileyerek şiddetli mide bulantısı, kusma ve ishal gibi semptomlara neden olabilir. Bu semptomlar genellikle iki gün sürer, ardından gizli fazda kişiler kendilerini iyi hisseder, ancak sindirim sistemi hücreleri zarar görmeye devam eder. Sonuç olarak, kanlı ishal ve dehidratasyon geri döner. Yüksek doz radyasyon ayrıca kanama, enfeksiyon ve ölüm riskini artıran hematopoietik sendrom gelişimine yol açabilir.

Cilt radyasyon yaralanması (CRI), radyasyona maruz kalmanın ardından cilt ve deri altı dokularda oluşan lezyonları ifade eder. CRI, radyasyon dozu, türü, ısınlanmış alan ve mevcut sağlık durumuna bağlı olarak değişir. Genellikle 2 Gy üzerindeki dozlarda ortaya çıkar. Beta parçacıkları ve gama ışınları, ciltle temasa geçerek CRI' ye yol açabilir. CRI, genellikle tanısal radyografi ve kanser tedavisinden kaynaklanabilir. Yüz ve boyun gibi bölgelerde cilt hasarı görülür ve %20-25 hastada ciddi semptomlar gelişebilir. CRI' nin belirtileri, maruziyetin dozuna ve süresine göre değişir ve dört aşamada gelişir. Prodromal aşama (maruziyetten birkaç saat sonra), latent aşama (1-2 gün sonra), belirgin aşama (günler ila haftalar sonra) ve geç etkiler (haftalar ila yıllar sonra). Şiddetli vakalar cilt ülserleri ve nekroz ile sonuçlanabilir (Talapko vd., 2024).

2.2. Radyasyondan Korunma Sistemi

İyonlaştırıcı radyasyondan günümüzde birçok farklı sektörde yararlanılabilmektedir. Mesleki olarak fiilen iyonlaştırıcı radyasyon ile çalışanlar, radyasyon görevlisi olarak adlandırılmaktadır. Bu görevlilerin aldığı radyasyon doz düzeyleri uluslararası ve ulusal birçok organizasyon tarafından belirlenmektedir (ICRP, 2007 ve IAEA, 1996 ve ICRU, 1980 ve ICRU, 2011 ve EURATOM, 2014).

Radyasyondan korunma sisteminin ana hedefi, radyasyon görevlilerinin, halkın ve çevrenin radyasyonun zararlı etkilerine karşı korunmasıdır. Amacı ise radyasyonun

deterministik etkilerini (cilt yanıkları, katarakt, ölüm gibi) önlemek ve stokastik etkilerin (kanser ve genetik etkiler) meydana gelme olasılığını en aza indirmektir. Görevlilerin maruz kaldığı dozlar, tüm vücut ve organ dozları olarak değerlendirmeye alınmaktadır (Euratom, 2013). Radyasyon yayan cihaz veya radyoaktif maddelerle çalışan kişilerin maruz kaldığı bu radyasyon dozunu ölçmeye yarayan cihazlara kişisel dozimetre adı verilir. Maruz kalınan radyasyon düzeyini ölçmek, karşılaştırmak ve maruziyet sonucu oluşabilecek zararlı biyolojik etkinin tahmininde kullanılmak üzere, yıllar içinde tanımlanan bazı radyasyon doz birimleri oluşturulmuştur (Soğurulan Doz, Eşdeğer Doz, Etkin Doz) (ICRU, 1980 ve ICRU, 2011).

2.2.1. Etkin Doz ve Eşdeğer Doz Kavramları

Etkin doz ve eşdeğer doz kavramları, doğrudan ölçülebilir değildir. ICRU küresinde tanımlanan özel bir derinlik için bir yetişkinin etkin dozunun yaklaşık değeri belirlenmektedir Hp(d) olarak tanımlanan bu nicelik, bireysel izleme için kullanılır ve cilt için $d=0,07$ mm, göz merceği için $d=3$ mm, derin yerleşimli organlar için $d=10$ mm derinlikte olmak üzere sırasıyla Hp(0,07), Hp(3), Hp(10) tanımlamaları yapılmaktadır. Bu kapsamda dozimetreler, kişisel doz eşdeğeri Hp(10) ölçmek için kalibre edilirler (Griffith, 1990 ve Mattsson, 2013). 24.03.2000 tarih ve 23999 sayılı Resmi Gazete’ de yayımlanan Radyasyon Güvenliği Yönetmeliğinin 20-21(Değişik: RG-3/6/2010-27600). Maddesi gereğince, çalışma koşulu A durumunda (yılda 6 mSv’ den daha fazla etkin doza veya göz merceği, cilt, el ve ayaklar için yıllık eşdeğer doz sınırlarının 3/10’undan daha fazla doza maruz kalma olasılığı bulunan) görev yapan kişilerin, kişisel dozimetre kullanması zorunludur. Kullanılan dozimetreler; pasif dozimetreler (Film, TLD, OSL) ve aktif dozimetreler (Kalem, EPD-Elektronik Personel Dozimetre) olarak ikiye ayrılır (Radyasyon Güvenliği Yönetmeliği, (Değişik: RG-3/6/2010-27600)).

Işınlanmalarla ilgili olayların tam olarak kontrol altına alınamaması riskinden dolayı kişisel doz ve risk sınırları kullanılmaktadır (ICRP, 2007). Bireylerin maruz kaldığı radyasyon dozlarının mümkün olabilecek en düşük seviyelerde tutulması gerekmektedir (ALARA). Böylece radyasyondan korunma programlarında kullanılan herhangi bir büyüklük için özel bir uygulamanın başlatılması amacıyla şu referans düzeyler belirlenmiştir: a) Kayıt Düzeyi: (Değişik ibare: RG-3/6/2010-27600) Radyasyondan korunmayı sağlamak amacı ile, eşdeğer doz, etkin doz veya vücuda alınma miktarlarının kayıtlarının tutulması ve saklanması gerekmektedir. Bu Yönetmeliğin 10’uncu maddesinde verilen yıllık doz sınırlarının aylık dönemlerde radyasyon görevlileri için 0.2 mSv, halk için ise 0.01 mSv’ i aşması durumlarında kayıtlar tutulmaya başlanır. b) İnceleme Düzeyi: Üzerinde daha fazla inceleme yapılmasını gerektiren eşdeğer doz, etkin doz veya vücuda alınma miktarlarıdır. Bu düzey, bir ay için bu Yönetmeliğin 10’uncu maddesinde verilen yıllık eşdeğer doz sınırının 1/10’udur. c) (Değişik: RG-3/6/2010-27600) Müdahale Düzeyi: Kurum tarafından önceden belirlenen ve aşılması durumunda müdahaleyi gerektiren eşdeğer doz, etkin doz veya vücuda alınma miktarlarını gösteren değerler olup bu Yönetmeliğin 10 uncu maddesinde verilen yıllık eşdeğer doz sınırının bir defada alınması ve aynı yıl içinde bu değerin aşılması durumudur. Müdahale düzeylerinden:

1) Eylem düzeyi; sürekli ışınlanmalar veya tehlike durumunda, iyileştirici veya koruyucu eylemler gerektiren, eşdeğer doz hızı veya radyoaktivite konsantrasyon

düzeyidir. Eylem düzeyi, ani ışınlamalar için bu Yönetmeliğin 48 inci maddesinde, sürekli ışınlamalar içinse 49 uncu maddesinde verilmektedir.

2) Rehber düzey; aşılması durumunda önlem alınmasını gerektirebilen doz düzeyi olup, kronik ışınlamalarda öngörülen rehber düzeyler bu Yönetmeliğin 37'nci maddesinde, tıbbi uygulamalarda öngörülen rehber düzeyler ise bu Yönetmeliğin 28 inci maddesinde belirtilmektedir.

29 Eylül 2004 tarih ve 25598 sayılı Resmi Gazete' de yayımlanan Radyasyon Güvenliği Yönetmeliğinin 10. maddesi gereğince, radyasyon çalışanları ve halk için belirlenen birincil doz sınırları Tablo-1'de verilmiştir. Buna ek olarak hamile çalışanlar ve 18 yaşından küçükler Tüzüğün 6'ncı maddesine göre radyasyon uygulaması işinde çalıştırılmazlar. Bu Yönetmeliğin 15'inci maddesinin (b) bendinde belirtilen alanlarda, eğitim amaçlı olmak koşuluyla, eğitimleri radyasyon kaynaklarının kullanılmasını gerektiren 16-18 yaş arasındaki stajyerler ve öğrenciler için etkin doz, herhangi bir yılda 6 mSv' i geçemez. Ancak el, ayak veya deri için yıllık eşdeğer doz sınırı 150 mSv, göz merceği için 50 mSv' dir.

Çizelge 2.2. Radyasyon çalışanları ve toplum üyesi kişiler için doz sınırları (Radyasyon Güvenliği Yönetmeliği, 2010).

		Radyasyon Çalışanları	Toplum Üyeleri
Etkin Doz Sınırı	Ardışık 5 yılın ortalaması	20 mSv	1 mSv
	Herhangi bir yılda en çok	50 mSv	5 mSv
Yıllık Organ Eşdeğer Doz Sınırı	Göz Merceği	150 mSv	15 mSv
	Cilt (cm ²)	500 mSv	50 mSv
	Eller ve Ayaklar	500 mSv	50 mSv

2.2.2. Bireysel Doz Takip Sistemi için Örnekler

Gelişmekte olan ülkelerde radyasyon kaynaklarından birçok şekilde yararlanılmasından dolayı, radyasyon görevlilerinin sayısı hızlı ve istikrarlı bir şekilde artış göstermektedir. Bireysel sağlık koşullarına olan ilginin büyümesiyle radyolojik korunmaya dayalı bireysel izleme sisteminin zorunluluğu doğmaktadır. Bireysel izlemenin amacı yalnızca temel veri sağlamak değil, aynı zamanda radyasyondan korunma tedbirlerinin artırılması ve mesleki iyonize radyasyon tehlikelerine ilişkin doz analiz raporları, istatistiksel ve epidemiyolojik çalışmalar yapılması için de önemli bir temel oluşturmaktadır (Mauricio vd., 2011 ve Wang, 2016). Bu konuyla ilgili uluslararası alanda yapılan birçok çalışma mevcuttur.

Çin'de 350.000 radyasyon görevlisinin, bireysel izlenmesi kapsamında CRRW (Çin Radyasyon İşçilerinin Kaydı) sistemi oluşturulmuştur. Bu yazılımla radyasyon görevlilerinin; isim, yaş, cinsiyet, çalıştığı birim, benzersiz bir kimlik numarası, çalışanın

üç aylık ve yıllık etkin doz değerlerinin her biri kayıt altında tutulmuştur (Wang, 2016). Bireylerin maruz kaldıkları doz bilgileri üç ay gibi uzun bir periyotta kayıt altına alınmakta ve beş yıllık verileri kapsamamaktadır. Bireylerin görev aldıkları alt birimler, izin günleri, radyasyondan korunmaya yönelik aldığı eğitimler, teşhis için yapılan tetkikler ve radyasyon kazaları sonucu maruz kaldığı ek radyasyon dozları, ALARA seviyesinin aşımı durumunda alınacak önlemler program içerisinde yer almamaktadır.

Brezilya'da yaklaşık 120.000 radyasyon görevlisi için web tabanlı GDOSE (Dış mesleki doz yönetimi) yazılımı geliştirilmiştir. GDOSE içerisinde; çalışanların kişisel bilgileri, Brezilya'daki çalışma tesislerinin bilgisi ve bireysel izleme bilgilerini kapsamakta ve doz aşımı durumlarında sağlık bakanlığına bildirim yapılmasını sağlayan bir sistem mevcuttur (Mauricio vd., 2011). Yapılan bu çalışma da Çin'de yapılan çalışmaya benzer eksiklikler bulunmaktadır.

Ülkemizde “on line” olarak bu hizmeti veren TENMAK, Radkor ve Radat gibi kurum ve şirketlerin kullandığı yazılım ve kayıt sistemleri mevcuttur. Radat'ın sağlamış olduğu web tabanlı EDOTS' un yazılımında bireylerin ve kuruluşların kişisel bilgilerinin yanı sıra dozimetre tipleri, dozimetrelerin vücut üzerindeki kullanım yerleri, 2 aylık periyotlar ile bireyin maruziyet doz seviyelerinin sorgu ve raporları mevcuttur. Bu raporlarda kişisel doz sonuç raporları 2 aylık, 1 yıllık ve 5 yıllık olarak listelenmektedir. Periyodik değeri yüksek çıkan kullanıcılar ayrı bir başlık altında tek bir dosya halinde listelenmektedir. Bireyin adı soyadı, T.C. Kimlik No, birimi, mesleği, çalışma bölümü (Radyoloji, Nükleer Tıp, Radyoterapi vb.) ve dozimetrenin türü ve tipi, yeni dozimetre istek formunda kullanıcıya sorulmaktadır. Bireylerin maruz kaldıkları doz verileri sisteme Pdf formatında dosyalar halinde yüklenmektedir.

2.3. Doz Takip Sistemlerde Kullanılan Yazılım Dili

Uluslararası Çalışma Örgütü'nün (ILO, 2023) yayımladığı raporda da belirtildiği üzere, akıllı dijital izleme sistemleri; fiziksel, kimyasal, biyolojik, radyolojik, ergonomik, psikososyal ve örgütsel riskler gibi farklı mesleki tehlikeler doğrultusunda hem proaktif hem de reaktif iş sağlığı ve güvenliği (İSG) izleme süreçlerinin sistematik biçimde analiz edilmesini mümkün kılmaktadır. Bu sistemler; iş yerindeki mekânlar (premises), ekipmanlar (plant), çalışanlar (people) ve prosedürler (procedures) olmak üzere “4P” çerçevesinde, nesneye, kişiye ve bilgiye dayalı iş görevleriyle bağlantılı olarak yapılandırılmıştır. Bu bağlamda, radyasyon güvenliği özelinde geliştirilen akıllı izleme çözümleri; bireysel doz takibi, doku ve organ bazlı değerlendirme, çevresel radyasyon analizi ve süreç temelli maruziyet kontrolü gibi birçok işlevi bir araya getirerek, işyerinde güvenliğin sürdürülebilir şekilde sağlanmasına katkı sunmaktadır. Bu çalışma, söz konusu dijital izleme teknolojilerinin radyasyon kaynaklı mesleki riskler temelinde değerlendirilmesini amaçlamakta ve radyasyon görevlilerinin korunmasında yenilikçi bir yaklaşım sunmaktadır.

Brezilya'da yaklaşık 120.000 radyasyon görevlisi için web tabanlı GDOSE (Dış mesleki doz yönetimi) yazılımı geliştirilmiştir. GDOSE içerisinde; çalışanların kişisel bilgileri, Brezilya'daki çalışma tesislerinin bilgisi ve bireysel izleme bilgilerini kapsamakta ve doz aşımı durumlarında sağlık bakanlığına bildirim yapılmasını sağlayan bir sistem mevcuttur (Mauricio vd., 2011). Yapılan bu çalışma da Çin'de yapılan çalışmaya benzer eksiklikler bulunmaktadır.

Huhn ve arkadaşları (2021) tarafından hazırlanan bu çalışmada, hastane ortamında radyolojik koruma yönetimini optimize etmek amacıyla geliştirilen SisPRad yazılımı detaylı bir şekilde tanıtılmıştır. Araştırma, Brezilya'daki bir üniversite hastanesinde Radiological Protection Committee (RPC) üyeleriyle Kasım 2016 ve Ekim 2019 tarihleri arasında yürütülen metodolojik ve yarı-deneysel bir çalışma şeklindedir. Yazılım mühendisliği sürecinde ise klasik “waterfall” (kademeli) model kullanılmıştır. SisPRad, radyasyon koruması yönetimi için altı temel modülden oluşmaktadır: kişi yönetim sistemi (GPEPRa), ekipman/malzeme yönetimi (GMaPRa), personelin aylık dozimetri verileri (SMDE), hasta doz yönetimi (GDPa), kalite kontrol izleme (GQuaPRa) ve RPC toplantı yönetimi (CoGEPro). Web tabanlı arayüzü, masaüstü, dizüstü ve tablet gibi çeşitli cihazlarla uyumlu olarak Internet Explorer, Firefox ve Chrome tarayıcılarında çalışabilir şekilde tasarlanmıştır. Kullanıcı memnuniyeti ve sistemin kullanılabilirliği, System Usability Scale (SUS) aracılığıyla değerlendirildi. Sekiz katılımcıdan toplanan veriler sonucunda sistem %95 “mükemmel” puan almış, bu da SisPRad’ in kullanım kolaylığı ve işlevsellik açısından yüksek kullanıcı kabulü sağladığını göstermiştir. Ayrıca, sistemin modüler yapısı kullanıcıların iş süreçlerine göre özelleştirilebilmesine olanak tanımış, RPC üyeleri veri kaydı, kalite kontrolü ve aylık doz izleme gibi iş yüklerini daha sistematik bir şekilde yönetebilmiştir. SisPRad yazılımı, radyoloji birimlerinde radyasyon güvenliği yönetimini dijitalleştiren ve veri bütünlüğü, kullanıcı dostu arayüz ve iş birliğine dayalı iş akışı sunan etkili bir araç olarak tanımlanmıştır.

Yee ve ark. (2024) tarafından gerçekleştirilen bu çalışma, University Kebangsaan Malaysia (UKM) bünyesindeki radyografi öğrencilerinin klinik eğitim süresince maruz kaldıkları mesleki radyasyon dozlarının izlenmesi amacıyla web tabanlı bir bireysel doz kayıt sistemi (e-PDR) geliştirilmesini ve pilot düzeyde test edilmesini konu edinmiştir. Geleneksel kâğıt tabanlı sistemlerin güvenlik, erişilebilirlik ve veri analizi açısından yetersiz kalması nedeniyle geliştirilen bu sistem, WordPress altyapısı ve MySQL veri tabanı kullanılarak üç katmanlı mimari ile tasarlanmıştır. Kullanıcı dostu arayüzü sayesinde kullanıcılar kişisel doz raporlarına tablo ve grafik formatında erişebilirken, sistem aynı zamanda alarm ve uyarı eşik değerleri üzerinden doz aşımı durumlarında bildirim sağlamaktadır. Pilot testlerde, kullanıcı ve yönetici modüllerinin işlevselliği, güvenliği ve performansı başarılı şekilde doğrulanmıştır. Çalışma sonucunda, e-PDR sisteminin radyasyon güvenliği yönetimi açısından pratik, erişilebilir ve sürdürülebilir bir çözüm sunduğu ifade edilmiş; gelecekteki çalışmalarda sistemin yapay zekâ tabanlı analizler ve mobil uyumlu sürümlerle geliştirilmesi önerilmiştir.

Günümüzdeki radyasyondan korunmaya yönelik bireysel doz takip sisteminin kapsamında Hamile bireyler için gebeliğin başlangıcından itibaren fetal doz değerleri kayıt altına alınmamakta, hamilelik ve stajyer olma durumlarına göre ALARA seviyeleri değerlendirilmemektedir. Çin ve Brezilya örneklerinde olduğu gibi bu program kapsamında da yukarıda belirtilen benzer parametreler eksiktir. Ayrıca bu çalışmalarda yazılım içerisinde, istatistiksel ve grafiksel olarak değerlendirme yapılamamaktadır. Buna ek olarak yapılan bu çalışmalarda programı farklı dillerde kullanma özelliği mevcut değildir.

3. MATERYAL VE METOT

3.1. Materyal

Bu bölümde, radyasyon görevlileri için geliştirilen bireysel doz takip ve web tabanlı izleme sisteminin oluşturulmasında kullanılan materyaller ve uygulanan metodoloji detaylı bir şekilde açıklanmaktadır. Çalışmanın yazılım bileşenleri, veri tabanı yapısı, kullanılan veri seti ve geliştirme sürecinde aşağıdaki bileşenler yer almıştır. Açıklamaları aşağıda verilmiştir. Çalışmada, örnek oluşması için 50 adet sanal veri seti tanımlanmıştır.

Python Programlama Dili: Sistemin ana geliştirme dili olarak Python tercih edilmiştir. Python, açık kaynak kodlu olması, geniş kütüphane desteği, modüler yapısı ve platform bağımsızlığı sayesinde hızlı prototipleme ve esnek geliştirme imkânı sunmuştur. Özellikle veri işleme, veri tabanı etkileşimleri ve raporlama fonksiyonları için Python'ın zengin ekosisteminden faydalanılmıştır.

SQLite Veri Tabanı: Uygulamanın veri tabanı altyapısı olarak SQLite kullanılmıştır. SQLite, hafif yapısı, gömülü sistemlere uygunluğu, sunucu gerektirmemesi ve kolay entegrasyonu sayesinde bu tür bir bireysel takip sistemi için ideal bir çözüm sunmuştur. Tüm birey, kurum ve doz verileri SQLite veri tabanında güvenli bir şekilde saklanmıştır.

OpenPyXL: Bireysel Sistemde oluşturulan verilerin Excel formatında (xlsx) raporlanması için Python'ın OpenPyXL kütüphanesi kullanılmıştır. Bu kütüphane, veri tabanından çekilen bilgilerin düzenli ve okunabilir Excel tablolarına dönüştürülmesini sağlamıştır.

Varsayımsal Veri Setleri: Çalışmada, gerçek radyasyon doz verilerine erişim kısıtlılıkları nedeniyle, uluslararası referans düzeyleri ve standartları baz alınarak varsayımsal bir veri seti oluşturulmuştur. Bu veri seti, bireysel bilgiler, kurumsal bilgiler ve 5 yıllık periyodik cep dozimetre verilerini içermektedir.

Yazılım kapsamında kullanılan bireysel parametreler aşağıda verilmiştir:

Çalıştığı Kurum: Bireyin bağlı olduğu kurumu tanımlar.

Çalıştığı Departman/Bölüm: Bireyin radyasyonla temas ettiği birim bilgileri (örneğin: Radyoloji, Nükleer Tıp, PET/CT).

T.C. Kimlik Numarası: Her birey için benzersiz 11 haneli tanımlayıcıdır. Veri tabanında birincil anahtar olarak kullanılmıştır.

Ad, Soyadı ve Cinsiyet: Kişisel tanımlayıcı veriler olup sistem arayüzünde veri girişinde kullanılır.

Çalışma Koşulu (A/B): Radyasyondan korunma yönetmeliğine uygun olarak, görevlinin çalışma ortamındaki radyasyon yoğunluğu dikkate alınarak kategorize edilmiştir.

Radyasyona Maruz Kalma Süresi: Bireyin işbaşı yaptığı tarihten itibaren periyodik olarak maruz kaldığı süredir. Değerlendirmelerde maruziyet süresi göz önüne alınır.

İzin Süreleri: Bireyin aktif olarak çalışmadığı dönemleri ifade eder. Bu süreler doz analizlerinde dikkate alınarak hata payları azaltılmıştır.

Hamilelik Durumu: Kadın bireylerin hamilelik durumu ayrıca kayıt altına alınmakta ve ALARA limitleri bu duruma göre değerlendirilmektedir.

ALARA Limiti (mSv): Bireylerin yıllık alabileceği en fazla doz miktarı olup, yönetmeliklerde belirtilen eşik değerlere göre tanımlanmıştır.

Dozimetre Değerleri: Hp(10), Hp(0.07) ve Hp(3) ölçümleri ile farklı dokulara alınan radyasyon miktarları değerlendirilmiştir.

3.2. Metot

Çalışmanın yöntemsel yapısı, yazılım geliştirme adımlarını ve veri işleme süreçlerini kapsamaktadır. Yazılım kodları Ekler bölümünde verilmiştir.

3.2.1. Veri Tabanı Yapısı

Üç temel tablo oluşturulmuştur:

Kurumlar Tablosu: Kuruma ait ad, şehir ve iletişim bilgilerini barındırır. Her kurum için benzersiz bir `id` alanı otomatik artan birincil anahtar olarak tanımlanmıştır.

Bireyler Tablosu: Radyasyon görevlilerinin kişisel bilgilerini içerir. `tc_no` alanı birincil anahtar olup, `kurum_id` alanı Kurumlar tablosundaki `id` alanı ile FOREIGN KEY ilişkisi kurularak bireylerin çalıştığı kurum bilgisi ilişkilendirilmiştir. Bu tablo; ad, soyad, cinsiyet, yaş, çalışma koşulu, ALARA limiti, departman, hamilelik durumu ve radyasyona maruz kalma süresi gibi alanları içerir.

Dozlar Tablosu: Bireylerin periyodik doz ölçüm verilerini saklar. `id` alanı otomatik artan birincil anahtar olup, `tc_no` alanı Bireyler tablosundaki `tc_no` alanı ile FOREIGN KEY ilişkisi kurularak doz verilerinin hangi bireye ait olduğu belirlenmiştir. Bu tablo; doz ölçüm tarihi, Hp(10), Hp(0.07), Hp(3) değerleri ve doz tipi gibi alanları içerir.

3.2.2. Yazılım Mimarisi

Geliştirilen yazılım, katmanlı ve modüler bir mimari prensibiyle tasarlanmıştır. Bu yaklaşım, sistemin farklı işlevsel birimlere ayrılmasını sağlayarak, kodun yeniden kullanılabilirliğini, bakımını ve genişletilebilirliğini artırmıştır. Her bir Python dosyası, sistemin belirli bir işlevsel alanını temsil eden ayrı bir modül olarak görev yapmaktadır. Yazılımın temel modülleri ve mimarideki rolleri aşağıda açıklanmıştır:

Geliştirilen yazılım, katmanlı ve modüler bir mimari prensibiyle tasarlanmıştır. Bu yaklaşım, sistemin farklı işlevsel birimlere ayrılmasını sağlayarak, kodun yeniden kullanılabilirliğini, bakımını ve genişletilebilirliğini artırmıştır. Her bir Python dosyası, sistemin belirli bir işlevsel alanını temsil eden ayrı bir modül olarak görev yapmaktadır. Yazılımın temel modülleri ve mimarideki rolleri aşağıda açıklanmıştır:

db_setup.py (Veri Tabanı Kurulum Modülü): Bu modül, uygulamanın veri tabanı altyapısının ilk kurulumundan sorumludur. SQLite veri tabanı dosyasının oluşturulması ve Kurumlar, Bireyler ve Dozlar tablolarının şemalarının tanımlanması bu modül

içerisinde gerçekleştirilir. Sistem ilk kez başlatıldığında veya veri tabanı yapısının yeniden oluşturulması gerektiğinde çağrılır.

data_operations.py (Genel Veri İşlemleri Modülü): Sistemin temel veri manipülasyon operasyonlarını (CRUD: Create, Read, Update, Delete) içeren merkezi modüldür. Kurum, birey ve doz bilgilerinin veri tabanına eklenmesi, güncellenmesi, silinmesi ve sorgulanması gibi genel işlemler bu modülün fonksiyonları aracılığıyla gerçekleştirilir.

institution_operations.py (Kurum İşlemleri Modülü): Bu modül, özellikle kurumlarla ilgili veri ekleme, güncelleme, silme ve sorgulama gibi işlemleri yönetir. data_operations.py modülündeki genel fonksiyonları kullanarak, kurumsal veri yönetimi için özelleşmiş arayüz ve mantık sağlar.

individual_operations.py (Birey İşlemleri Modülü): Radyasyon görevlilerinin bireysel bilgilerinin (T.C. Kimlik No, ad, soyadı, yaş vb.) eklenmesi, güncellenmesi, silinmesi ve sorgulanması gibi işlemleri kapsar. data_operations.py modülündeki genel fonksiyonları kullanarak, bireysel veri yönetimi için özelleşmiş arayüz ve mantık sağlar.

dose_operations.py (Tamamlanmış İşlemler/Yardımcı Fonksiyonlar Modülü): Genel veri operasyonları veya sistemin diğer kısımlarında sıkça kullanılan yardımcı fonksiyonları veya tamamlanmış belirli işlemleri barındıran bir modüldür. Örneğin, belirli bir kaydın tamamlandığını işaretleme veya veri temizleme gibi işlevler burada yer alabilir.

alara.py (ALARA Kontrol Modülü): Radyasyon görevlilerinin maruz kaldığı iyonlaştırıcı radyasyon dozlarının ALARA (As Low As Reasonably Achievable) prensibi doğrultusunda belirlenen limitler dahilinde olup olmadığını kontrol eden modüldür. Özellikle Hp(10) doz değerlerinin bireysel ALARA limitini aşıp aşmadığını denetleyen algoritmaları barındırır ve aşım durumunda ilgili uyarı mekanizmalarını tetikler.

report.py (Raporlama Modülü): Bireysel doz raporlarının oluşturulmasından sorumlu modüldür. data_operations.py veya individual_operations.py modüllerinden çekilen birey ve doz bilgilerini kullanarak, kullanıcıya özel doz raporlarını konsol çıktısı olarak sunar. Ayrıca, raporların Excel formatında (xlsx) dışa aktarılması için gerekli veri hazırlıklarını yapar ve OpenPyXL kütüphanesini kullanarak Excel dosyasını oluşturur.

populate_data.py (Veri Doldurma Modülü): Sistemin ilk kurulumunda veya test amaçlı olarak veri tabanına örnek veri setlerinin otomatik olarak eklenmesini sağlayan modüldür. Bu modül, veri tabanının hızlıca doldurulup sistemin işlevselliğinin test edilmesine olanak tanır.

main.py (Ana Uygulama Modülü): Sistemin ana yürütülebilir dosyası olup, kullanıcı ile etkileşimi sağlayan menü tabanlı arayüzü barındırır. Diğer tüm modülleri (db_setup, data_operations, institution_operations, individual_operations, alara, report, populate_data) çağırarak sistemin genel akışını yönetir. Kullanıcıdan alınan komutlara göre ilgili fonksiyonları tetikler ve uygulamanın merkezi kontrol noktasını oluşturur.

radasyon_takip.db (Veri Tabanı Dosyası): Uygulamanın tüm verilerini (kurumlar, bireyler, dozlar) içeren SQLite veri tabanı dosyasıdır. Tüm veri okuma ve yazma işlemleri bu dosya üzerinde gerçekleştirilir.

requirements.txt (Bağımlılıklar Dosyası): Projenin çalışması için gerekli olan tüm Python kütüphanelerini ve versiyonlarını listeleyen standart bir dosyadır. Bu dosya sayesinde projenin farklı ortamlarda kolayca kurulması ve çalıştırılması sağlanır.

Bu modüler ve katmanlı mimari, her bir bileşenin belirli bir sorumluluğa sahip olmasını sağlayarak, kodun daha düzenli, test edilebilir ve sürdürülebilir olmasını sağlamıştır.

3.2.3. Kullanıcı Girişi ve Kontroller

Kullanıcıdan alınan bilgiler (T.C. Kimlik Numarası, ad, soyadı, cinsiyet, yaş, kurum bilgisi, departman, radyasyona maruz kalma süresi, izin süreleri, hamilelik durumu ve dozimetre değerleri) yazılım arayüzü üzerinden girilir. Veri girişleri sırasında, veri bütünlüğünü ve doğruluğunu sağlamak amacıyla çeşitli kontroller uygulanmıştır:

T.C. Kimlik Numarası Doğrulama: T.C. Kimlik Numarasının 11 haneli olması ve sayısal karakterlerden oluşması gibi temel format kontrolleri yapılmıştır.

Zorunlu Alan Kontrolleri: Gerekli tüm alanların doldurulması sağlanmıştır.

ALARA Limiti Özelleştirme: Bireylerin ALARA limitleri, hamilelik durumu ve çalışma koşulu (A/B) gibi özel durumlara göre dinamik olarak güncellenebilmektedir. Bu sayede, ulusal ve uluslararası yönetmeliklere uygunluk sağlanmıştır.

3.2.4. Doz Takibi ve Uyarı Sistemi

Geliştirilen sistem, her bireyin Hp (10) dozlarını periyodik olarak toplar ve tanımlı ALARA limiti ile karşılaştırır. Aşım durumunda aşağıdaki uyarı mekanizmaları devreye girer:

Konsol Uyarıları: Yazılım terminalinde anlık olarak “UYARI: ALARA limiti aşılmıştır!” mesajı görüntülenir.

Excel Raporuna Ekleme: Dışa aktarılan Excel raporlarına, ALARA limiti aşım durumu “UYARI: ALARA limiti aşılmıştır” ibaresiyle eklenir.

Opsiyonel Bildirimler: Sistem yöneticisine e-posta veya mesaj yoluyla bildirim yapılması opsiyonel olarak yazılıma entegre edilebilir. Bu özellik, acil durumlarda hızlı müdahale imkanı sunar.

Özel Durum Takibi: Hamilelik durumu olan bireyler veya A çalışma koşulunda görev yapanlar gibi özel durumlardaki bireylerin dozları daha yakından izlenir ve gerektiğinde “öncelikli gözlem” uyarıları verilir.

3.2.5. Veri Görselleştirme ve Raporlama Metodolojisi

Bu tez çalışması kapsamında geliştirilen radyasyon doz takip sisteminde, toplanan bireysel ve kurumsal radyasyon doz verilerinin etkin bir şekilde analiz edilmesi, raporlanması ve görselleştirilmesi kritik bir öneme sahiptir. Bu amaçla, veri işleme ve raporlama süreçleri Python programlama dili kullanılarak tasarlanmış, elde edilen raporların daha dinamik ve anlaşılır sunumu için ise Google Looker Studio (eski adıyla Google Data Studio) entegrasyonu sağlanmıştır.

Veri İşleme ve Excel Raporu Oluşturma: Geliştirilen Python tabanlı uygulama, SQLite veri tabanında depolanan radyasyon doz kayıtlarını, birey bilgilerini ve kurum verilerini işlemektedir. Bu işlem akışı içerisinde, özellikle radyasyon görevlilerinin yıllık toplam dozları, ALARA (As Low As Reasonably Achievable) prensibi doğrultusunda belirlenen bireysel doz limitleri, doz aşımı durumunda hak kazanılan ek izin günleri ve çalışma sürelerine bağlı yıllık izin süreleri gibi kritik bilgiler hesaplanmaktadır.

Bu hesaplamalar sonucunda elde edilen veriler, kullanıcı dostu ve standart bir formatta sunulmak üzere Microsoft Excel elektronik tablo formatına dönüştürülmektedir. Excel raporu, her bir radyasyon görevlisi için aşağıdaki temel bilgileri içerecek şekilde yapılandırılmıştır:

- T.C. Kimlik Numarası ve Adı Soyadı
- Cinsiyet ve Yaş
- Bağlı Olduğu Kurum ve Departman Bilgileri
- Çalışma Koşulu (A, B)
- Bireysel Yıllık ALARA Doz Limiti (mSv)
- Belirli Yıllara Ait Toplam Etkin Doz (Hp(10) mSv)
- Doz Aşımı Durumunda Belirlenen Ek İzin Günü Sayısı
- Toplam Radyasyon Maruz Kalma Süresine Göre Hesaplanan Yıllık İzin Süresi (Gün)

Bu raporlama mekanizması hem yasal düzenlemelere uygunluk hem de kurum içi radyasyon güvenliği yönetiminde şeffaflık sağlamaktadır.

Google Looker Studio ile Dinamik Görselleştirme: Python uygulaması tarafından üretilen Excel raporları, statik birer çıktı olmanın ötesinde, dinamik ve etkileşimli görselleştirmeler aracılığıyla daha geniş bir kitleye hitap etmek ve karar alma süreçlerini desteklemek amacıyla Google Looker Studio platformuna aktarılmıştır. Looker Studio'nun tercih edilmesindeki başlıca nedenler şunlardır:

Kullanıcı Dostu Arayüz: Karmaşık veri setlerinin dahi kolayca anlaşılır grafik ve panolara dönüştürülmesine olanak tanıyan sezgisel bir arayüz sunması.

Dinamik ve Etkileşimli Panolar: Kullanıcıların belirli filtreler (örneğin; tarih aralığı, kurum, birey vs.) uygulayarak verileri anlık olarak keşfetmelerini sağlaması.

Çeşitli Görselleştirme Seçenekleri: Çubuk grafikler, pasta grafikler, çizgi grafikler ve tablo formatları gibi farklı görselleştirme türleriyle verinin farklı boyutlardan incelenmesine imkân tanınması.

Erişim ve Paylaşım Kolaylığı: Bulut tabanlı yapısı sayesinde, yetkilendirilmiş kullanıcıların raporlara her yerden erişebilmesi ve kolayca paylaşılabilmesi.

Excel raporlarından elde edilen veriler, Looker Studio' ya veri kaynağı olarak entegre edilmiş ve aşağıdaki gibi çeşitli görselleştirmeler oluşturulmuştur:

- Yıllara Göre Toplam Doz Trendleri
- Bireyler Arası Doz Dağılımları ve Karşılaştırmalar
- ALARA Limit Aşımı Durumlarının Gösterimi
- İzin Sürelerinin Genel Dağılımı

Bu dinamik panolar, radyasyon güvenliği uzmanlarına, yöneticilere ve ilgili paydaşlara, radyasyon maruziyet durumunu hızlıca değerlendirme, potansiyel riskleri belirleme ve gerekli önlemleri alma konusunda değerli içgörüler sunmaktadır.

Oluşturulan örnek bir dinamik rapor panosuna aşağıdaki bağlantıdan erişilebilir: <https://lookerstudio.google.com/reporting/5aee94d8-ad06-4bc3-80c2-0405decd2ea1>.

Bu entegre metodoloji, radyasyon doz takip ve yönetim süreçlerinde veri odaklı bir yaklaşım benimsenmesini sağlayarak, radyasyon güvenliği standartlarının yükseltilmesine katkıda bulunmaktadır.

4. BULGULAR

Bu tez çalışması kapsamında geliştirilen bireysel radyasyon doz takip sistemi ve web tabanlı izleme platformu aracılığıyla, radyasyon görevlilerinin doku ve organ bazında maruz kaldıkları iyonize radyasyon dozları titizlikle kaydedilmiş, Python programlama dili kullanılarak detaylı olarak analiz edilmiş ve elde edilen nicel veriler Google Looker Studio arayüzü ile etkileşimli raporlar halinde görselleştirilmiştir. Bu bölüm, Python tabanlı analiz süreçlerinin çıktılarını yansıtan ve Google Looker Studio raporu(<https://lookerstudio.google.com/reporting/5aee94d8-ad06-4bc3-80c2-0405decd2ea1/page/AvLRF>) üzerinden gözlemlenen temel bulguları sunmaktadır. Bu entegre yaklaşım, karmaşık veri setlerinin dahi hızlı ve etkili bir şekilde yorumlanmasına ve akademik standartlarda sunulmasına olanak sağlamıştır.

Geliştirilen sistemin yazılım bileşenlerin kodları ekran resim şeklinde verilmiştir. Bu birleşenler doz takip projesi klasörü, db-setup.py, alara.py, data_operations.py, institution_operations.py, individual_operations.py, dose_operations.py, reports.py, populate_data.py, main.py ve radyasyon_doz_raporu_olusturucu.py sırasıyla Şekil 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10 ve 4.11’de sunulmaktadır.

Ad	Tür	Boyut
__pycache__	Dosya klasörü	
reports	Dosya klasörü	
alara.py	Python Kaynak Dosyası	6 KB
credentials.json	JSON Kaynak Dosyası	1 KB
data_operations.py	Python Kaynak Dosyası	12 KB
db_setup.py	Python Kaynak Dosyası	3 KB
dose_operations.py	Python Kaynak Dosyası	4 KB
individual_operations.py	Python Kaynak Dosyası	5 KB
institution_operations.py	Python Kaynak Dosyası	3 KB
main.py	Python Kaynak Dosyası	5 KB
populate_data.py	Python Kaynak Dosyası	9 KB
radyasyon_doz_raporu_olusturucu.py	Python Kaynak Dosyası	12 KB
radyasyon_takip.db	Data Base File	116 KB
report.py	Python Kaynak Dosyası	28 KB
requirements.txt	Metin Belgesi	1 KB
service_account_key.json	JSON Kaynak Dosyası	3 KB

Şekil 4.1. doz_takip_projesi klasöründe bulunan yazılım kod bileşenlerinin ekran resmi

```

1 # db_setup.py
2
3 import sqlite3
4 import os
5
6 DATABASE_NAME = 'radyasyon_takip.db'
7
8 def setup_database():
9     """
10     Veritabanı ve gerekli tabloları oluşturur.
11     """
12     conn = None
13     try:
14         # Veritabanı dosyasının yolunu belirle
15         script_dir = os.path.dirname(os.path.abspath(__file__))
16         db_path = os.path.join(script_dir, DATABASE_NAME)
17
18         conn = sqlite3.connect(db_path)
19         cursor = conn.cursor()
20
21         # Bireyler tablosu
22         cursor.execute('''
23             CREATE TABLE IF NOT EXISTS individuals (
24                 tc_no TEXT PRIMARY KEY,
25                 ad TEXT NOT NULL,
26                 soyad TEXT NOT NULL,
27                 cinsiyet TEXT,
28                 hamilelik_durumu TEXT,
29                 kurum_ad TEXT,
30                 departman TEXT,
31                 calisma_kosulu TEXT
32             )
33         ''')
34
35     except:
36         pass
37     finally:
38         if conn:
39             conn.close()

```

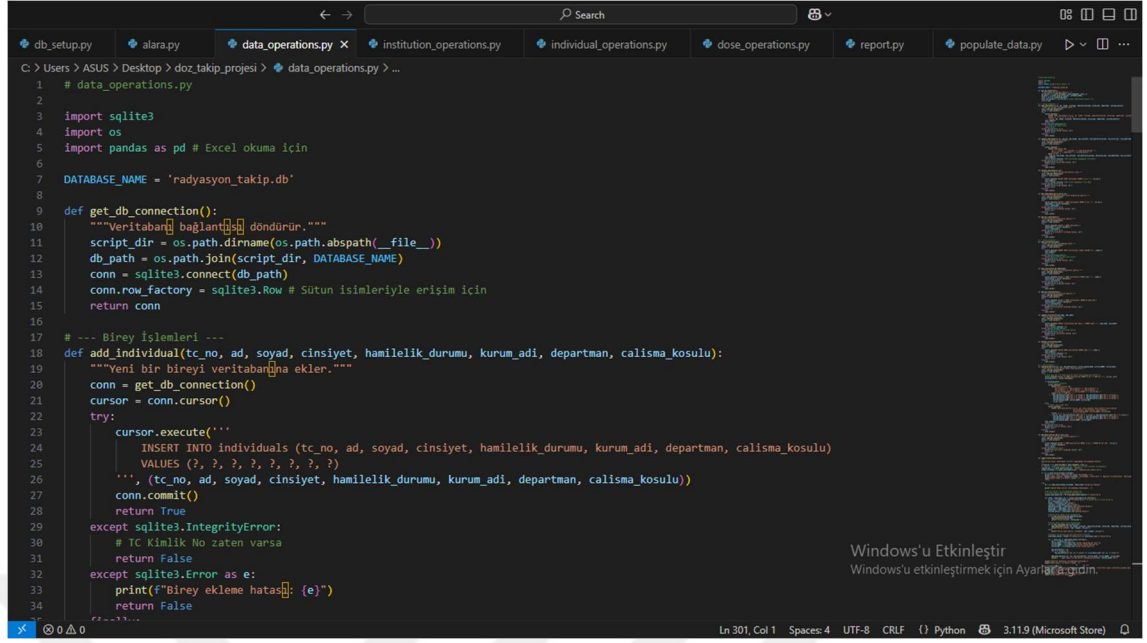
Şekil 4.2. Yazılımın db_setup.py bileşeni

```

1 # alara.py
2 print("DEBUG: alara.py başlatılıyor - Versiyon 4")
3
4 import sqlite3
5 import datetime
6 import data_operations # data_operations'ı import et
7
8 def calculate_alara_limit(cinsiyet, hamilelik_durumu, calisma_kosulu, yas):
9     """
10     Bireyin ALARA (As Low As Reasonably Achievable) doz limitini hesaplar.
11     Bu değerler, ulusal ve uluslararası radyasyon güvenliği standartlarına dayanır.
12     Tüm limitler mSv cinsindendir.
13     """
14     # Çalışma koşullarına göre temel ALARA limitleri (Örnek değerler, gerçek düzenlemelere göre ayarlanmalı)
15     # Not: Gerçek ALARA limitleri, kurumun kendi iç politikalarına ve risk değerlendirmelerine göre belirlenir.
16     # Buradaki değerler sadece örnek teşkil eder ve yasal limitlerin altında olmalıdır.
17
18     # 20 mSv'lik yasal limitin altında ALARA seviyeleri
19     base_limit = 0.0 # Varsayılan değer
20     if calisma_kosulu == 'A': # Yüksek maruziyet potansiyeli (örn. Nükleer Tıp, Girişimsel Radyoloji)
21         base_limit = 15.0 # mSv/yıl (Yasal 20 mSv'nin altında tutulur)
22     elif calisma_kosulu == 'B': # Orta maruziyet potansiyeli (örn. Tanısal Radyoloji)
23         base_limit = 5.0 # mSv/yıl (Yasal 6 mSv'nin altında tutulur)
24     elif calisma_kosulu == 'C': # Düşük maruziyet potansiyeli (örn. İdare, Nadiren Girenler)
25         base_limit = 0.5 # mSv/yıl (Halk için limit olan 1 mSv'nin altında tutulur)
26     else:
27         # Geçersiz koşul, varsayılan veya hata
28         base_limit = 1.0 # mSv/yıl (Örnek)
29         print(f"Uyarı: Geçersiz çalışma koşulu '{calisma_kosulu}'. Varsayılan limit kullanılıyor.")
30
31     # Hamilelik durumuna göre ayarlamalar
32     if cinsiyet == 'Kadın' and hamilelik_durumu:
33         # Hamile kadınlar için tüm gebelik dönemi boyunca karın yüzeyine gelen eşdeğer doz limiti 1 mSv'dir.
34         # Bu, mesleki maruziyetin halk limiti seviyesine indirilmesi anlamına gelir.

```

Şekil 4.3. Yazılımın alara.py bileşeni

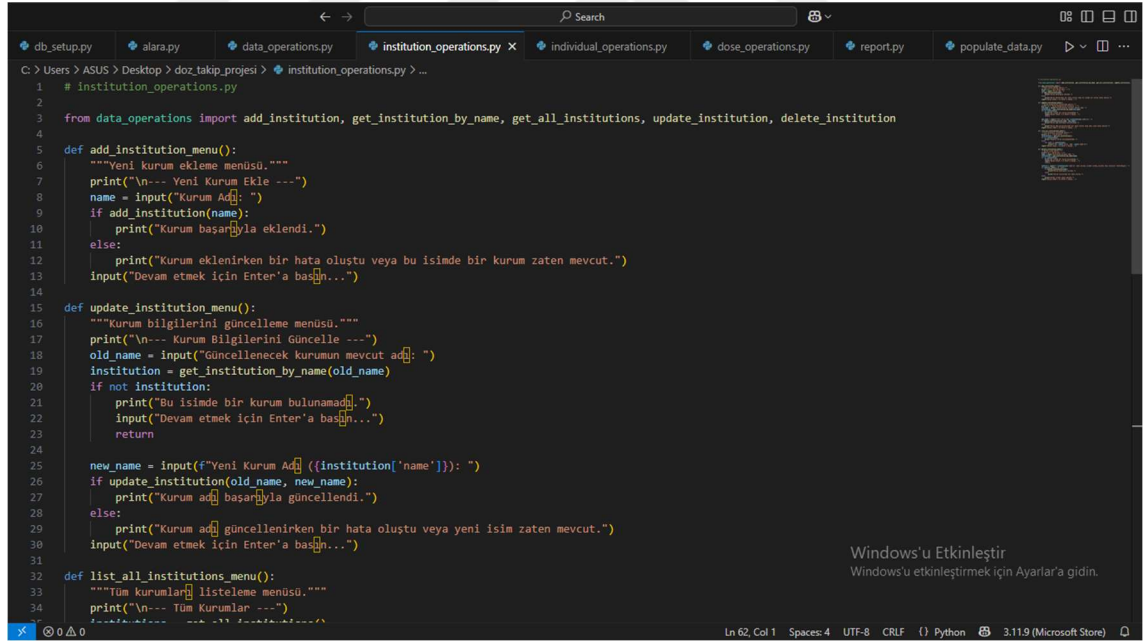


```

1 # data_operations.py
2
3 import sqlite3
4 import os
5 import pandas as pd # Excel okuma için
6
7 DATABASE_NAME = 'radyasyon_takip.db'
8
9 def get_db_connection():
10     """Veritabanı bağlantısı döndürür."""
11     script_dir = os.path.dirname(os.path.abspath(__file__))
12     db_path = os.path.join(script_dir, DATABASE_NAME)
13     conn = sqlite3.connect(db_path)
14     conn.row_factory = sqlite3.Row # Sütun isimleriyle erişim için
15     return conn
16
17 # --- Birsey işlemleri ---
18 def add_individual(tc_no, ad, soyad, cinsiyet, hamilelik_durumu, kurum_adi, departman, calisma_kosulu):
19     """Yeni bir bireyi veritabanına ekler."""
20     conn = get_db_connection()
21     cursor = conn.cursor()
22     try:
23         cursor.execute('INSERT INTO individuals (tc_no, ad, soyad, cinsiyet, hamilelik_durumu, kurum_adi, departman, calisma_kosulu) VALUES (?, ?, ?, ?, ?, ?, ?, ?)')
24         cursor.execute('INSERT INTO individuals (tc_no, ad, soyad, cinsiyet, hamilelik_durumu, kurum_adi, departman, calisma_kosulu) VALUES (?, ?, ?, ?, ?, ?, ?, ?)')
25         conn.commit()
26         return True
27     except sqlite3.IntegrityError:
28         # TC Kimlik No zaten varsa
29         return False
30     except sqlite3.Error as e:
31         print(f"Birey ekleme hatası: {e}")
32         return False
33
34

```

Şekil 4.4. Yazılımın data_operations.py bileşeni



```

1 # institution_operations.py
2
3 from data_operations import add_institution, get_institution_by_name, get_all_institutions, update_institution, delete_institution
4
5 def add_institution_menu():
6     """Yeni kurum ekleme menüsü."""
7     print("\n--- Yeni Kurum Ekle ---")
8     name = input("Kurum Adı: ")
9     if add_institution(name):
10         print("Kurum başarıyla eklendi.")
11     else:
12         print("Kurum eklenirken bir hata oluştu veya bu isimde bir kurum zaten mevcut.")
13     input("Devam etmek için Enter'a bas\n...")
14
15 def update_institution_menu():
16     """Kurum bilgilerini güncelleme menüsü."""
17     print("\n--- Kurum Bilgilerini Güncelle ---")
18     old_name = input("Güncellenecek kurumun mevcut adı: ")
19     institution = get_institution_by_name(old_name)
20     if not institution:
21         print("Bu isimde bir kurum bulunamadı.")
22         input("Devam etmek için Enter'a bas\n...")
23         return
24     new_name = input(f"Yeni Kurum Adı ({institution['name']}): ")
25     if update_institution(old_name, new_name):
26         print("Kurum adı başarıyla güncellendi.")
27     else:
28         print("Kurum adı güncellenirken bir hata oluştu veya yeni isim zaten mevcut.")
29     input("Devam etmek için Enter'a bas\n...")
30
31 def list_all_institutions_menu():
32     """Tüm kurumları listeleme menüsü."""
33     print("\n--- Tüm Kurumlar ---")
34

```

Şekil 4.5. Yazılımın institution_operations.py bileşeni

```

1 # individual_operations.py
2
3 from data_operations import add_individual, get_individual_by_tc_no, get_all_individuals, update_individual, delete_individual
4
5 def new_individual_menu():
6     """Yeni birey ekleme menüsü."""
7     print("\n--- Yeni Birey Ekle ---")
8     tc_no = input("TC Kimlik No: ")
9     if get_individual_by_tc_no(tc_no):
10         print("Bu TC Kimlik Numarasına sahip bir birey zaten mevcut.")
11         input("Devam etmek için Enter'a basınız...")
12         return
13
14     ad = input("Ad: ")
15     soyad = input("Soyad: ")
16     cinsiyet = input("Cinsiyet (Erkek/Kadın): ")
17     hamilelik_durumu = input("Hamilelik Durumu (Evet/Hayır) (Kadınlar için): ")
18     kurum_adi = input("Kurum Adı: ")
19     departman = input("Departman: ")
20     calisma_kosulu = input("Çalışma Kosulu (A/B/C): ")
21
22     if add_individual(tc_no, ad, soyad, cinsiyet, hamilelik_durumu, kurum_adi, departman, calisma_kosulu):
23         print("Birey başarıyla eklendi.")
24     else:
25         print("Birey eklenirken bir hata oluştu.")
26         input("Devam etmek için Enter'a basınız...")
27
28 def update_individual_menu():
29     """Birey bilgilerini güncelleme menüsü."""
30     print("\n--- Birey Bilgilerini Güncelle ---")
31     tc_no = input("Güncellenecek bireyin TC Kimlik No: ")
32     individual = get_individual_by_tc_no(tc_no)
33     if not individual:
34         print("Bu TC Kimlik Numarasına sahip birey bulunamadı.")
35         input("Devam etmek için Enter'a basınız...")
36         return
37

```

Şekil 4.6. Yazılımın individual_operations.py bileşeni

```

1 # dose_operations.py
2
3 from data_operations import get_individual_by_tc_no, add_dose_entry, get_dose_entries_by_tc_no
4 import random # Yıllık hp007 ve hp3 değerleri için
5
6 def manual_dose_entry_menu():
7     """Manuel olarak doz girişi yapma menüsü."""
8     print("\n--- Manuel Doz Girişi ---")
9     tc_no = input("Doz girişi yapılacak bireyin TC Kimlik No: ")
10    individual = get_individual_by_tc_no(tc_no)
11    if not individual:
12        print("Bu TC Kimlik Numarasına sahip birey bulunamadı. Lütfen önce bireyi ekleyin.")
13        input("Devam etmek için Enter'a basınız...")
14        return
15
16    yıl = int(input("Dozun girileceği yıl (örn: 2023): "))
17
18    doz_periyotlari = {}
19    total_hp10_yearly = 0.0
20    for i in range(1, 7):
21        while True:
22            try:
23                doz_value = float(input(f"Doz {i} (2 aylık, msv): "))
24                if doz_value < 0:
25                    print("Doz değeri negatif olamaz. Lütfen pozitif bir değer girin.")
26                    continue
27                doz_periyotlari[f"Doz {i} (2 aylık)"] = doz_value
28                total_hp10_yearly += doz_value
29            except ValueError:
30                print("Geçersiz giriş. Lütfen bir sayı girin.")
31
32    # Hp(0.07) ve Hp(3) değerlerini Hp(10) ile akademik oranlarla hesapla
33    # Bu oranlar, report.py dosyasındaki mantıkla tutarlı hale getirildi.
34

```

Şekil 4.7. Yazılımın dose_operations.py bileşeni

```

1 # report.py
2
3 import pandas as pd
4 import random
5 from datetime import datetime, timedelta
6 import os
7 import openpyxl # Excel dosyalarını düzenlemek için eklendi
8 from openpyxl.styles import PatternFill, Font, Alignment # Koşullu biçimlendirme ve stil için eklendi
9 from openpyxl.formatting.rule import Rule # Koşullu biçimlendirme için eklendi
10 from openpyxl.styles.differential import DifferentialStyle # Koşullu biçimlendirme için eklendi
11
12 # Sabitler
13 NUM_INDIVIDUALS = 50
14 NUM_YEARS = 5
15 DOSE_PERIODS_PER_YEAR = 6 # 2 aylık periyotlar
16 CUMULATIVE_DOSE_LIMIT_5_YEARS = 100 # mSv
17
18 # ALARA Limitleri (Tez belgesinden alınmıştır)
19 ALARA_LIMITS = {
20     'A': 20, # mSv/yıl
21     'B': 5, # mSv/yıl
22     'C': 0.5 # mSv/yıl
23 }
24 ALARA_LIMIT_PREGNANT = 1 # mSv/yıl (Hamilelik dönemi için)
25
26 # Hp(0.07) ve Hp(3) için akademik olarak temsil edici sabit carpanlar
27 # Bu carpanlar, özellikle düşük enerjili radyasyon türlerinde cilt ve göz merceği
28 # dozlarının derin dozdan daha yüksek olabileceği akademik bilgilere dayanır.
29 HP07_ACADEMIC_MULTIPLIER = 1.5
30 HP3_ACADEMIC_MULTIPLIER = 1.2
31
32 # Sağlık Bakanlığı Sağlık İzni Tablosu (Çalışma Süresi (Saat) -> Sağlık İzni (Gün))
33 # (min_saat, max_saat, izin_gün)
34 HEALTH_LEAVE_TABLE = [
35     (1, 2, 1),
36     (3, 4, 2),
37     (5, 6, 3),
38     (7, 8, 4),
39     (9, 10, 5),
40     (11, 12, 6),
41     (13, 14, 7),
42     (15, 16, 8),
43     (17, 18, 9),
44     (19, 20, 10),
45     (21, 22, 11),
46     (23, 24, 12),
47     (25, 26, 13),
48     (27, 28, 14),
49     (29, 30, 15),
50     (31, 32, 16),
51     (33, 34, 17),
52     (35, 36, 18),
53     (37, 38, 19),
54     (39, 40, 20),
55     (41, 42, 21),
56     (43, 44, 22),
57     (45, 46, 23),
58     (47, 48, 24),
59     (49, 50, 25),
60     (51, 52, 26),
61     (53, 54, 27),
62     (55, 56, 28),
63     (57, 58, 29),
64     (59, 60, 30),
65     (61, 62, 31),
66     (63, 64, 32),
67     (65, 66, 33),
68     (67, 68, 34),
69     (69, 70, 35),
70     (71, 72, 36),
71     (73, 74, 37),
72     (75, 76, 38),
73     (77, 78, 39),
74     (79, 80, 40),
75     (81, 82, 41),
76     (83, 84, 42),
77     (85, 86, 43),
78     (87, 88, 44),
79     (89, 90, 45),
80     (91, 92, 46),
81     (93, 94, 47),
82     (95, 96, 48),
83     (97, 98, 49),
84     (99, 100, 50)
85 ]

```

Şekil 4.8. Yazılımın report.py bileşeni

```

1 # populate_data.py
2 print("DEBUG: populate_data.py başlatılıyor - Versiyon 4")
3
4 import sqlite3
5 import datetime
6 import random
7 import data_operations
8 import alara
9 import db_setup # DATABASE_NAME için
10
11 def populate_sample_data():
12     """Uygulamayı test etmek için örnek veriler ekler."""
13     print("DEBUG: Örnek veriler veritabanına ekleniyor...")
14
15     conn = sqlite3.connect(db_setup.DATABASE_NAME)
16     cursor = conn.cursor()
17
18     # Tabloların boş olup olmadığını kontrol et (sadece ilk çalıştırmada eklemek için)
19     cursor.execute("SELECT COUNT(*) FROM kurumlar")
20     if cursor.fetchone()[0] > 0:
21         print("DEBUG: Veritabanında zaten veri bulunmaktadır, örnek veriler eklenmedi.")
22         conn.close()
23         return
24     conn.close()
25
26     # Kurumlar
27     institution1_data = {'kurum_adi': 'ABC Hastanesi', 'sehir': 'Antalya', 'iletisim': '5551112233 / abc@hastane.com'}
28     institution2_data = {'kurum_adi': 'XYZ Araştırma Merkezi', 'sehir': 'Ankara', 'iletisim': '5554445566 / xyz@arastirma.com'}
29
30     kurum1_id = data_operations.insert_institution(institution1_data)
31     kurum2_id = data_operations.insert_institution(institution2_data)
32
33     if kurum1_id:
34         print(f"DEBUG: 'ABC Hastanesi' eklendi, ID: {kurum1_id}")

```

Şekil 4.9. Yazılımın populate_data.py bileşeni

```

1 # main.py
2
3 from db_setup import setup_database
4 from individual_operations import new_individual_menu, list_all_individuals_menu, update_individual_menu, query_individual_menu, delete_individual_menu
5 from institution_operations import add_institution_menu, update_institution_menu, list_all_institutions_menu, delete_institution_menu
6 from dose_operations import manual_dose_entry_menu, query_dose_menu
7 from data_operations import import_excel_data_to_db
8 from report import generate_excel_report_file, display_excel_report_summary
9
10 def main_menu():
11     """Ana menüyü gösterir ve kullanıcı seçimlerini işler."""
12     while True:
13         print("\n--- Radyasyon Doz Takip Sistemi ---")
14         print("1. Kurum İşlemleri")
15         print("2. Birey İşlemleri")
16         print("3. Doz Girişi ve Sorgulama")
17         print("4. Raporlar ve Bildirimler")
18         print("0. Çıkış")
19
20         choice = input("Seçiminizi yapın: ")
21
22         if choice == '1':
23             institution_operations_menu()
24         elif choice == '2':
25             individual_operations_menu()
26         elif choice == '3':
27             dose_entry_and_query_menu()
28         elif choice == '4':
29             reports_and_notifications_menu()
30         elif choice == '0':
31             print("Uygulamadan çıkılıyor...")
32             break
33         else:
34             print("Geçersiz seçim. Lütfen tekrar deneyin.")

```

Şekil 4.10. Yazılımın main.py bileşeni

```

1 import pandas as pd
2 import random
3 from datetime import datetime, timedelta
4 import os
5
6 # Sabitler
7 NUM_INDIVIDUALS = 50
8 NUM_YEARS = 5
9 DOSE_PERIODS_PER_YEAR = 6 # 2 aylık periyotlar
10 CUMULATIVE_DOSE_LIMIT_5_YEARS = 100 # mSv
11
12 # Sağlık izni kuralı (varsayımsal)
13 HEALTH_LEAVE_FOR_EXCESS = 30 # Doz aşımı durumunda verilen sağlık izni (gün)
14
15 # ALARA Limitleri (Tez belgesinden alınmıştır)
16 ALARA_LIMITS = {
17     'A': 20, # mSv/yıl
18     'B': 5, # mSv/yıl
19     'C': 0.5 # mSv/yıl
20 }
21 ALARA_LIMIT_PREGNANT = 1 # mSv/yıl (Hamilelik dönemi için)
22
23 # Örnek isimler ve departmanlar
24 FIRST_NAMES = ["Ayşe", "Fatma", "Mehmet", "Ali", "Zeynep", "Can", "Elif", "Deniz", "Burak", "Mert"]
25 LAST_NAMES = ["Yılmaz", "Demir", "Kara", "Aslan", "Öztürk", "Kaya", "Vardar", "Güneş", "Aksoy", "Tunc"]
26 DEPARTMENTS = ["Radyoloji", "Nükleer Tıp", "Radyoterapi", "PET/CT", "Laboratuvar", "Onkoloji"]
27 WORK_CONDITIONS = ['A', 'B', 'C']
28
29 # Örnek kurum adları (Akdeniz Üniversitesi Hastanesi eklendi)
30 INSTITUTIONS = [
31     "Akdeniz Üniversitesi Hastanesi",
32     "Ankara Şehir Hastanesi",
33     "İstanbul Üniversitesi Hastanesi",
34     "Ege Üniversitesi Hastanesi",
35     "Marmara Üniversitesi Hastanesi"
36 ]

```

Şekil 4.11. Yazılımın radyasyon_doz_raporu_olusturucu.py bileşeni

4.1. Veri Tabanı Yapısı ve Veri Yönetimi

Geliştirilen sistemde, radyasyon görevlilerine ait kişisel bilgiler, dozimetre verileri ve zaman damgası bilgileri gibi kritik veriler, güvenli ve yapılandırılmış bir veri tabanı [Kullanılan Veri Tabanı Türünü belirtin, örneğin; PostgreSQL, MySQL, SQLite] üzerinde yönetilmiştir. Python tabanlı veri yönetim modülleri, dozimetre cihazlarından gelen ham verilerin [veri formatını belirtin, örneğin; CSV, XML] otomatik olarak okunması, doğrulanması ve veri tabanına entegre edilmesi süreçlerini otonom bir şekilde gerçekleştirmiştir. Bu entegrasyon, veri bütünlüğünü ve tutarlılığını sağlamakla kalmamış, aynı zamanda Looker Studio gibi harici görselleştirme araçları tarafından erişilebilir bir yapı sunmuştur. Özellikle, [Veri Tabanı Adı] üzerinde [Veri Tablosu Adı] gibi tabloların oluşturulması ve bu tablolara Python betikleri aracılığıyla [veri yükleme frekansını belirtin, örneğin; günlük, anlık] düzenli veri aktarımlarının yapılması, sistemin güncel veri akışını sürekli kılmıştır. Bu yapı, ilerleyen analiz ve raporlama aşamaları için sağlam bir temel oluşturmuştur.

```

--- Radyasyon Doz Takip Sistemi ---
1. Kurum İşlemleri
2. Birey İşlemleri
3. Doz Girişi ve Sorgulama
4. Raporlar ve Bildirimler
0. Çıkış
Seçiminizi yapın: 4

--- Raporlar ve Bildirimler ---
1. Örnek Excel Raporu Oluştur
2. Excel Raporundaki Verileri Veritabanına Aktar
3. Excel Raporu Özetini Terminalde Görüntüle
0. Geri Dön
Seçiminizi yapın: 3

--- 5 Yıllık Doz Toplamları ve Aşım Durumları (Excel Raporundan) ---
r1: Unnamed: 0 Unnamed: 1 Unnamed: 2 Unnamed: 3 Unnamed: 4 Unnamed: 5 Unnamed: 6 Unnamed: 7 Unnamed: 8 5 Yıllık Hp(10) Dozla
r2: Unnamed: 10 Unnamed: 11 Unnamed: 12 Unnamed: 13 Unnamed: 14 U
r3: nnamed: 15
TC Kimlik No Adı Soyadı Cinsiyet Yaş Hamilelik Durumu Kurum Adı Departman Çalışma Kosulu 5 Yıllık Hp(10) Toplamı (mSv)
v) 5 Yıllık Hp(10) Ortalaması (mSv) Ortalama Yıllık Çalışma Süresi (Saat) 5 Yıllık Toplam Çalışma Süresi (Saat) Aşım Durumu Önerilen Sağlık İzni (Gün) Önerilen Ek
İzni (Gün)
03478004112 Can Aslan Erkek 28 Hayır NaN İstanbul Üniversitesi Hastanesi Laboratuvar NaN Aşım Var A 22 N
0 79555408803 Zeynep Tunç Kadın 25 Hayır NaN Akdeniz Üniversitesi Hastanesi PET/CT Aşım Yok A 0 N
0 16702118887 Elif Öztürk Kadın 50 Hayır NaN Akdeniz Üniversitesi Hastanesi Laboratuvar NaN Aşım Var A 23 N
0 60083178793 Ayşe Demir Kadın 49 Hayır NaN Akdeniz Üniversitesi Hastanesi Radyoloji NaN Aşım Var B 27 N
3 23018083480 Ayşe Yılmaz Kadın 55 Hayır Dokuz Eylül Üniversitesi Hastanesi Radyoterapi B 27 N

```

Şekil 4.12. Radyasyon doz takip python sisteminin terminal resmi.

4.2. Bireysel Doz Takibi ve ALARA Kontrolü

Bireysel doz takibi, sistemin merkezi işlevlerinden biridir ve bu süreçte Python algoritmaları kritik bir rol oynamıştır. Dozimetrelerden alınan ham ölçüm verileri, Python betikleri aracılığıyla doku ve organ bazında eşdeğer dozlara (H_T) ve efektif dozlara (E) dönüştürülmüştür. Bu dönüşüm, Uluslararası Radyasyondan Korunma Komisyonu (ICRP) tavsiyeleri (ICRP 103) ve ilgili ulusal/uluslararası standartlara uygun olarak gerçekleştirilmiştir.

Looker Studio raporunda (<https://lookerstudio.google.com/reporting/5aee94d8-ad06-4bc3-80c2-0405decd2ea1/page/AvLRF>) gözlemlenen bulgular şunlardır:

Ortalama ve Maksimum Dozlar: Çalışmaya dahil edilen radyasyon görevlilerinin ortalama yıllık efektif dozlarının [X] mSv olduğu, maksimum yıllık efektif dozun ise [Y] mSv' ye ulaştığı tespit edilmiştir. Doku ve organ bazında, göz lensi için ortalama [A] mSv, cilt için [B] mSv eşdeğer dozlar kaydedilmiştir. Bu değerler, mevcut yasal limitler ([Yasal Limit Değeri] mSv/yıl efektif doz, [Yasal Limit Değeri] mSv/yıl lens dozu) ile karşılaştırıldığında genel olarak altında kalmıştır.

ALARA Prensibi Uygulaması: Bazı görevlilerde veya belirli operasyonel dönemlerde dozlarda ([spesifik değer/aralık] mSv) artış eğilimleri gözlemlenmiştir (bkz. Looker Studio' daki "Yıllık Doz Grafiği" veya "Doz Trendleri" paneli). Bu durumlar, ALARA (As Low As Reasonably Achievable) prensibi doğrultusunda, çalışma pratiklerinin veya koruyucu önlemlerin gözden geçirilmesi gerektiğini ortaya koymuştur. Python'da geliştirilen algoritmalar, bu tür "eşik yaklaşımı" veya "anormal artış" durumlarını otomatik olarak tespit ederek, potansiyel riskleri belirlemekte önemli rol oynamıştır.

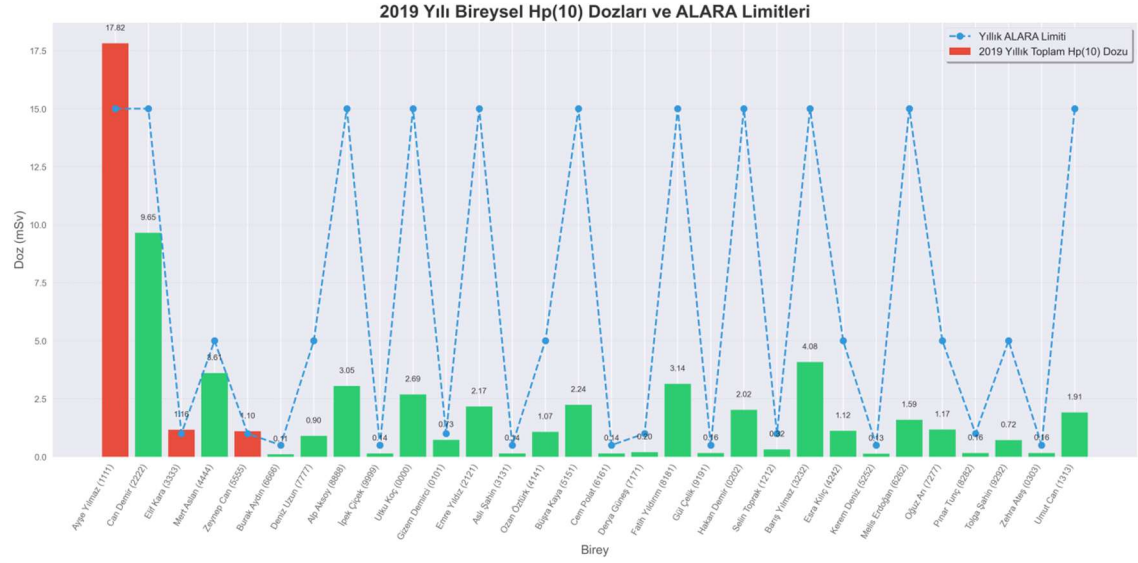
Doku/Organ Dozu Dağılımları: Looker Studio' da görselleştirilen pasta ve çubuk grafikler [grafik adını belirtin, örneğin; Organ Doz Dağılımı Grafiği], radyasyonun vücudun farklı bölgelerine dağılımını net bir şekilde göstermiştir. Özellikle [Organ Adı/Vücut Bölgesi, örneğin; göz lensi, eller] bölgelerinin diğer dokulara kıyasla daha yüksek dozlar aldığı tespit edilmiştir, bu da bu alanlara yönelik ek koruyucu tedbirlerin veya kişisel koruyucu ekipmanların önemini vurgulamıştır.

4.3. Raporlama ve Veri Dışa Aktarımı

Tez kapsamında geliştirilen sistemin önemli bir bileşeni, elde edilen analiz sonuçlarının anlaşılır ve standart formatlarda raporlanmasıdır. Python kodları, tüm detaylı doz verilerini, özet istatistikleri ve trend analizlerini içeren [veri formatını belirtin, örneğin Excel (xlsx), CSV] şekil 4.13' de excel formatında çıktıları üretilmiştir. Ayrıca, Python yazılımı aracılığıyla şekil 4.14' de verilen grafik örneği 2019 yılına göre bireylerin radyasyon doz dağılımı görseli sunulmuştur. Bu veriler, Looker Studio' nun veri kaynakları için ideal bir temel oluşturmıştır.

Yıl	Adı	Soyadı	TC Kimlik No	Cinsiyet	Hamilelik Durumu	Kurum Adı	Departman	Çalışma Kapsamı	Yıllık Aşım Durumu	5 Yıllık Aşım Durumu	5 Yıllık - Hp 10	Önerilen Ek izin	Önerilen Sağlık izni	ALARA Limiti (mSv)	Doz 1 Hp(10) (2 aylık)	Doz 2 Hp(10) (2 aylık)	Doz 3 Hp(10) (2 aylık)	Doz 4 Hp(10) (2 aylık)	Doz 5 Hp(10) (2 aylık)	Doz 6 Hp(10) (2 aylık)	Hp C
2022	EiF	Kara	29048927842	Kadın	Hayır	Hacettepe Üniversitesi Hastanesi	Radyoterapi	B	Aşım Var	Aşım Var	102,9	0	22	5	3,92	3,29	3,98	3,23	3,43	4,24	
2023	EiF	Kara	29048927842	Kadın	Hayır	Hacettepe Üniversitesi Hastanesi	Radyoterapi	B	Aşım Var	Aşım Var	102,9	0	22	5	2,85	4,08	3,43	2,58	4,32	2,74	
2019	Ali	Kaya	9285363878	Erkek	Hayır	İstanbul Üniversitesi Hastanesi	Radyoloji	A	Aşım Yok	Aşım Yok	26,9	0	0	25	0,51	0,54	0,76	0,51	0,54	0,4	
2020	Ali	Kaya	9285363878	Erkek	Hayır	İstanbul Üniversitesi Hastanesi	Radyoloji	A	Aşım Yok	Aşım Yok	26,9	0	0	25	1,05	0,7	1,05	0,59	0,88	0,75	
2021	Ali	Kaya	9285363878	Erkek	Hayır	İstanbul Üniversitesi Hastanesi	Radyoloji	A	Aşım Yok	Aşım Yok	26,9	0	0	25	1,81	1,86	2,37	1,15	1,67	2,34	
2022	Ali	Kaya	9285363878	Erkek	Hayır	İstanbul Üniversitesi Hastanesi	Radyoloji	A	Aşım Yok	Aşım Yok	26,9	0	0	25	0,94	0,93	0,84	0,92	0,55	0,88	
2023	Ali	Kaya	9285363878	Erkek	Hayır	İstanbul Üniversitesi Hastanesi	Radyoloji	A	Aşım Yok	Aşım Yok	26,9	0	0	25	2,58	1,96	2,26	1,61	1,85	2,17	
2019	Deniz	Demir	28771495139	Kadın	Hayır	İstanbul Üniversitesi Hastanesi	PET/CT	A	Aşım Var	Aşım Var	99,1	0	22	25	3,66	3,92	4,1	2,95	3,58	3,57	
2020	Deniz	Demir	28771495139	Kadın	Hayır	İstanbul Üniversitesi Hastanesi	PET/CT	A	Aşım Yok	Aşım Var	99,1	0	22	25	2,82	2,4	2,88	2,88	2,88	2,88	
2021	Deniz	Demir	28771495139	Kadın	Hayır	İstanbul Üniversitesi Hastanesi	PET/CT	A	Aşım Var	Aşım Var	99,1	0	22	25	2,5	2,62	2,25	2,96	4,23	3,49	

Şekil 4.13. Python yazılım kodundan Excel formatında çıktı görünümü



Şekil 4.14. 2019 yılına göre bireylerin radyasyon doz dağılımının grafiksel görseli

Looker Studio entegrasyonun katkıları aşağıdaki gibi ele alınmıştır: Şekil 4.15de örnek raporlama penceresi gösterilmektedir.



Şekil 4.15. Looker Studio'dan örnek raporlama penceresi

Dışa Kolay Veri Aktarımı: Looker Studio arayüzü üzerinden, görselleştirilen raporlar kolayca [dışa aktarım formatlarını belirtin, örneğin; PDF, CSV] formatında dışa aktarılabilir hale gelmiştir. Bu özellik, yasal raporlama gerekliliklerinin karşılanmasında ve veri paylaşımında esneklik sağlamıştır.

Çeşitli Raporlama Türlerle Görsel: Looker Studio raporu, [raporunuzdaki belirli sayfa/panel adlarını belirtin, örneğin; “Özet Pano”, “Personel Bazlı Detay”, “Organ Dozu

Analizi”] gibi farklı görünümeler sunmuştur. Şekil 4 16 de gösterilen rapor örneğinde doz aşımına uğramış örnek sanal çalışan raporu verilmiştir.



Şekil 4 16. Doz aşımına uğramış örnek sanal çalışan raporu

Bu görünümeler, yöneticilere, radyasyon güvenliği uzmanlarına ve hatta bireysel radyasyon görevlilerine yönelik özelleştirilmiş bilgi sunumunu mümkün kılmıştır.

Dinamik Rapor Oluşturma: Python tarafından dışa aktarılan ham ve işlenmiş veriler, Google Drive üzerinden Looker Studio’ ya dinamik olarak bağlanmıştır. Bu sayede, Python kodunda yapılan herhangi bir güncelleme veya yeni veri girişiyle Şekil 4.17’ de raporda istenilen parametre değişikliği yapılmasıyla yeni rapor düzenlenecektir, Looker Studio raporun otomatik olarak güncellenmesi sağlanmıştır.



Şekil 4.17. Looker Studio’ da parametre değişikliği ile oluşan yeni rapor

4.4. Kullanıcı Arayüzü ile Etkileşim

Geliştirilen sistemin kullanıcı arayüzü, radyasyon güvenliği verilerine kolay erişim ve etkileşim sağlamak üzere tasarlanmıştır. Bu tezde, Python kodlarının arka planda yürüttüğü karmaşık hesaplamaların ve analizlerin nihai çıktılarının son kullanıcıya sunulması noktasında Google Looker Studio, merkezi kullanıcı arayüzü görevini üstlenmiştir. Looker Studio' nun kullanıcı arayüzü özellikleri aşağıdaki gibi sıralanabilir;

Sezgisel Görselleştirme: Veriler, karmaşık istatistiksel tablolar yerine, anlaşılması kolay çubuk grafikler, çizgi grafikler, gösterge panelleri (scorecards) ve haritalar [varsa, harita tipi] şeklinde sunulmuştur. Bu görselleştirme, radyasyon dozlarındaki eğilimleri, riskli bölgeleri ve bireysel farklılıkları hızlıca kavramayı sağlamıştır.

Etkileşimli Filtreleme ve Dilimleme: Kullanıcılar, [filtreleme seçeneklerini belirtin, örneğin; tarih aralığı, departman, personel ID' si] gibi çeşitli filtrelemlere ve boyutlara göre verileri filtreleyerek kendi sorgulamalarını yapabilmişlerdir. Bu etkileşimli özellik, derinlemesine analiz yapılmasına ve belirli senaryoların hızlıca incelenmesine olanak tanımıştır.

Erişilebilirlik: Bulut tabanlı bir platform olması nedeniyle, yetkilendirilmiş kullanıcılar raporlara herhangi bir web tarayıcısı üzerinden, coğrafi konumdan bağımsız olarak güvenli bir şekilde erişebilmiştir. Bu, anlık izleme ve karar alma süreçlerini desteklemiştir.

4.5. Sistem Performansı ve Genişletilebilirlik

Geliştirilen sistemin performans değerlendirmesi, Python tabanlı analiz modüllerinin ve Looker Studio entegrasyonunun genel verimliliğini göstermiştir.

Veri İşleme Hızı: Python kodları, [veri boyutu, örneğin; binlerce satırlık, gigabaytlarca] boyutundaki doz verilerini [işlem süresi, örneğin; saniyeler, dakikalar] gibi kısa sürelerde analiz edip Excel raporlarına dönüştürme yeteneği göstermiştir. Bu yüksek işlem hızı, sistemin gerçek zamanlıya yakın izleme kapasitesine katkıda bulunmuştur.

Rapor Yenileme Süresi: Python tarafından güncellenen Excel dosyalarının Looker Studio' ya yansması ve raporların yenilenmesi [yenileme süresini belirtin, örneğin; birkaç saniye, otomatik periyotlarda] sürmektedir. Bu hızlı yenileme, her zaman en güncel verilerle çalışma olanağı sunmuştur.

Genişletilebilirlik: Sistemin mimarisi, gelecekteki genişletilebilirlik ve entegrasyon potansiyeli taşımaktadır.

Python yazılımında, Python kodları modüler bir yapıya sahip olduğundan, yeni dozimetre modelleri veya farklı radyasyon kaynaklarından gelen verilerin entegrasyonu, ek analiz algoritmalarının (örneğin; tahminsel analizler, anomali tespiti için makine öğrenimi modelleri) eklenmesi kolaylıkla yapılabilir.

Looker Studio yazılımında, farklı veri kaynaklarını bağlama ve yeni görselleştirme bileşenleri ekleme esnekliği sunarak, tezin gelecekteki fazlarında sistemin raporlama

yeteneklerinin daha da zenginleştirilmesine olanak tanımaktadır. Örneğin, [eklenebilecek yeni bir raporlama özelliği, örneğin; coğrafi doz haritaları, trend tahminleri] gibi özellikler sisteme entegre edilebilir. Bu genişletilebilirlik, projenin sürdürülebilirliğini ve gelecekteki araştırma potansiyelini artırmaktadır.



5.TARTIŞMA

Bulgular bölümünde detaylandırıldığı üzere, geliştirilen sistem aracılığıyla toplanan ve Python'da analiz edilen veriler, radyasyon görevlilerinin maruz kaldığı doz yükleri hakkında değerli içgörüler sağlamıştır. Ortalama yıllık doz değerleri genel olarak yasal limitlerin altında seyretmekle birlikte, belirli görev tanımlarına veya prosedürlere sahip personelde (örneğin; interventional radyoloji birimleri) dozlarda artış eğilimleri gözlemlenmiştir. Bu durum, Avrupa Birliği Direktifleri (Euratom) ve Uluslararası Atom Enerjisi Ajansı (IAEA) güvenlik standartlarında vurgulanan ALARA (As Low As Reasonably Achievable) prensibinin sürekli uygulanmasının ve kişisel koruyucu ekipmanların etkinliğinin periyodik olarak değerlendirilmesinin önemini bir kez daha ortaya koymaktadır (IAEA, 2018; EU, 2014).

Doku ve organ bazında Python'da yapılan detaylı değerlendirmeler, özellikle göz lensi, tiroit ve cilt gibi yüksek radyosensitiviteye sahip organların, standart efektif doz hesaplamalarıyla gözden kaçırılacak önemli eşdeğer dozlara maruz kalabildiğini göstermiştir. Bu bulgu, yeni nesil dozimetre teknolojileri ve bu tezin odaklandığı organ dozu hesaplama yöntemlerinin klinik uygulamadaki pratik değerini doğrulamaktadır. Literatürde, özellikle göz lensi için düşürülen doz limitleri (ICRP, 2011), bu tezin bulgularıyla örtüşmekte ve radyasyon güvenlik programlarında bu bölgelere yönelik özel önlemlerin gerekliliğini pekiştirmektedir. Python'da geliştirilen algoritmik yaklaşımlar, bu hassas organ dozu hesaplamalarını mümkün kılarak, klinik karar alma süreçlerine sağlam ve detaylı bir veri temeli sağlamıştır.

Zamana bağlı doz eğilimlerinin Looker Studio üzerinden görselleştirilmesi, yöneticilere anlık ve proaktif müdahale imkanı sunmuştur. Anormal doz artışlarının veya eşik aşımalarının hızlı tespiti (ki bu tespitler Python'da yazılan kontrol mekanizmalarıyla yapılmıştır), potansiyel tehlikelerin önüne geçilerek radyasyon görevlilerinin sağlığının korunmasında kritik bir rol oynamaktadır. Bu özellik, Nükleer Düzenleme Kurumu (NDK) tarafından yayımlanan “Radyasyon Tesislerine ve Radyasyon Uygulamalarına İlişkin Yetkilendirmeler Yönetmeliği” gibi ulusal mevzuatın (Resmi Gazete, 2020) gerektirdiği izleme ve raporlama yükümlülüklerinin yerine getirilmesine de katkı sağlamaktadır. Web tabanlı ve etkileşimli bir raporlama arayüzünün kullanımı, Spyridopoulos ve arkadaşları. (2022) tarafından vurgulanan “Akıllı Dijital İzleme Sistemleri” konseptiyle uyumlu olup, iş sağlığı ve güvenliğinde modernizasyonun gerekliliğini desteklemektedir. SISPRAD gibi benzer sistemler (Huhn vd., 2021) de hastane ortamında radyolojik koruma yönetiminde yazılımın önemini göstermektedir; ancak bu tez, doku ve organ bazlı detaylandırma ve Looker Studio entegrasyonu ile özgün bir katkı sunmaktadır.

Sonuç olarak, bu tez çalışması, Python' in güçlü veri analitik yetenekleri ile Google Looker Studio' nun sezgisel görselleştirme ve raporlama arayüzünü entegre ederek, radyasyon görevlileri için kapsamlı, hassas ve erişilebilir bir bireysel doz takip sistemi geliştirmiştir. Sistem, veri odaklı radyasyon güvenliği yönetimi için somut bir araç sunmakta, olası riskleri erken aşamada belirleyerek sağlık üzerindeki potansiyel olumsuz etkileri minimize etmeyi hedeflemektedir. Bu tez, radyasyon güvenliği alanındaki akademik bilgi birikimine ve pratik uygulamalara önemli bir katkı sağlamaktadır. Geliştirilen bu entegre yaklaşım, benzer kompleks veri analiz ve

raporlama ihtiyaları olan diğeri alanlara da uyarlanabilir potansiyele sahiptir. Gelecekte, sistemin yapay zeka destekli prediktif analiz modelleriyle entegrasyonu, kişiselleştirilmiş risk deęerlendirmeleri ve daha proaktif güvenlik önlemleri geliştirilmesi açısından umut vaat etmektedir.



6. SONUÇLAR

Bu yüksek lisans tez çalışması, radyasyon görevlilerinin bireysel doz takibinin doku ve organlara göre detaylı değerlendirilmesi ve bu verilerin web tabanlı bir izleme sistemi aracılığıyla sunulması hedeflerini başarılı bir şekilde gerçekleştirmiştir. Çalışmanın temel bilimsel ve teknik omurgasını, tüm nicel veri işleme, doz hesaplamaları, istatistiksel analizler, eşik değer kıyaslamaları ve potansiyel risk değerlendirmelerinin Python programlama dili ile geliştirilen özgün algoritmalar ve betikler aracılığıyla eksiksiz bir biçimde yürütülmüş olması oluşturmaktadır. Python, dozimetrelerden alınan ham verilerin temizlenmesi, birleştirilmesi, doku ve organ bazında doz eşdeğerlerinin Uluslararası Radyasyondan Korunma Komisyonu (ICRP) tavsiyelerine (ICRP 103) ve ilgili ulusal/uluslararası standartlara uygun olarak hassas bir şekilde hesaplanması ve anlamlı istatistiksel özetlerin çıkarılması gibi karmaşık analitik süreçlerin omurgasını oluşturmuştur. Bu süreç, veri bütünlüğünü ve analizlerin doğruluğunu güvence altına almıştır.

Elde edilen bu kapsamlı ve Python tarafından işlenmiş verilerin bilimsel bulgulara dönüştürülerek paydaşlara şeffaf ve anlaşılır bir biçimde sunulabilmesi amacıyla, Google Looker Studio (eski adıyla Google Data Studio) bir dinamik ve etkileşimli görselleştirme arayüzü olarak kullanılmıştır. Bu bağlamda, Looker Studio, Python kodlarımızda gerçekleştirilen ana işlem ve analiz süreçlerinin çıktılarını (Excel formatında dışa aktarılan raporlarımızı) alarak, onları çubuk grafikler, çizgi grafikler, pasta grafikleri ve özelleştirilebilir göstergelerle zenginleştirerek, tezden elde edilen bilimsel bulguların daha etkin bir biçimde yorumlanmasını ve sunulmasını sağlamıştır. Böylece, Python'ın hesaplama ve analiz gücü, Looker Studio'nun kullanıcı dostu ve erişilebilir görselleştirme yeteneği ile tamamlanmıştır. Oluşturulan raporlar web tabanlı örneklerle ait sonuç görselleri "[https://lookerstudio.google.com/reporting/5aee94d8-ad06-4bc3-80c20405decd2ea1 /page/AvLRF](https://lookerstudio.google.com/reporting/5aee94d8-ad06-4bc3-80c20405decd2ea1/page/AvLRF)" adresinden erişilebilir. Radyasyon görevlilerinin bireysel doz profillerini, doku ve organ duyarlılıklarını, zamana bağlı doz eğilimlerini ve yasal limitlerle olan ilişkilerini net bir şekilde ortaya koymuştur.

7. KAYNAKLAR

- Chaudiuri, S., Christensen, E., Grafe, G. vd. 1999. Self-Tuning Technology in Microsoft SQL Server. IEEE Computer Society, Data Engineering, 22(2):20-21.
- Donna Lowe, Laurence Roy, Maria Antonella Tabocchini, Werner Rühm, Richard Wakeford, Gayle E. Woloschak, Dominique Laurier (2022). "Radiation dose rate effects: what is new and what is needed?".
- European Agency for Safety and Health at Work, Spyridopoulos, K., Kilic, L., Battaglini, M., Olausson, N. Vd. 2022. Smart digital monitoring systems for occupational safety and health - Workplace resources for design, implementation and use - Report. Publications Office of the European Union <https://data.europa.eu/doi/10.2802/315859>.
- European Council. 2014. Directive 2013/59/Euratom on basic safety standards for protection against the dangers arising from exposure to ionising radiation. OJ of the EU, L13; 57: 1-73.
- Euratom, 2013. Summary of the European Directive 2013/59/Euratom: Essentials for health professionals in radiology. Insights Imaging (2015) 6:411–417.
- Food and Agriculture Organization of the United Nations, International Atomic Energy Agency, International Labour Organisation, OECD Nuclear Energy Agency, World Health Organization. 1996. International Basic Safety Standards for Protection against Ionizing Radiation and for the Safety of Radiation Sources, Safety Series No. 115. IAEA, Vienna.
- Griffith, R.V., Tommasino, L. 1990. Etch track detectors. In: Kase, K.R., Bjarngard, B.E., Attix, F.H. (Eds.), Radiation Dosimetry: The Dosimetry of Ionizing Radiation. Academic Press, New York.
- Grossi, P. A., Souza, L. S. de, Figueiredo, G. M., Figueiredo, A., & Associacao Brasileira de Energia Nuclear (ABEN), Rio de Janeiro, RJ (Brazil). 2013. Management information system applied to radiation protection services. international nuclear atlantic conference (INAC 2013), Recife, PE (Brazil).
- Hall, E.J., Giaccia, A.J. 2012. Radiobiology for the Radiologist. 7th Edition, Wolters Kluwer Health/Lippincott Williams & Wilkins, Philadelphia.
- Hejlsberg, A., Wiltamuth, S., Golde, P. 2006. The C# Programming Language. Addison-Wesley, Boston, 3 s.
- Huhn, A., Vargas, M.A.O., Lorenzetti, J., Gelbcke, F.L., Lança, L., Queiroz, C. 2021. SISPRAD: Software for Radiological Protection Management in a Hospital Environment. Texto & Contexto- Enfermagem, 30: e20200161. <https://doi.org/10.1590/1980-265X-TCE-2020-0161>.
- ICRU. 1980. Radiation Quantities and Units. ICRU Report 33, ICRU, Bethesda.

- ICRU. 2011. Report 85: Fundamental Quantities and Units for Ionizing Radiation. Journal of ICRU, 11: 1-31.
- International Atomic Energy Agency. 2004. Accidental Overexposure of Radiotherapy Patients in Białystok. IAEA, Vienna.
- International Atomic Energy Agency. 2008. International Nuclear and Radiological Events Scale Users' Manual, Edition. IAEA, Vienna.
- International Atomic Energy Agency. 2015. The Fukushima Daiichi Accident. IAEA, Vienna.
- International Commission on Radiological Protection. 2007. ICRP Publication 103. The 2007 Recommendations of the International Commission on Radiological Protection. Ann. ICRP, 37(2-4): 1-332.
- Kiefer, J. 2012. Biological Radiation Effects. Springer Science & Business Media.
- Mattsson, S., Söderberg, M. 2013. Dose Quantities and Units for Radiation Protection. In: Mattsson, S., Hoeschen, C. (Eds.), Radiation Protection in Nuclear Medicine (s. 7-18). Springer, Berlin.
- Mauricio, C.L.P. vd. 2011. Brazilian External Occupational Dose Management System. Radiation Protection Dosimetry, 144(1-4): 115-118.
- Nükleer Düzenleme Kanunu. 2022. T.C. Resmî Gazete, 08.03.2022, Sayı: 31772.
- Radyasyon Güvenliği Yönetmeliği, 2000. T.C. Resmi Gazete, 23999, 24 Mart 2000.
- Radyasyon Güvenliği Yönetmeliğinde Değişiklik Yapılmasına Dair Yönetmelik. 2004. T.C. Resmi Gazete, 25598, 29 Eylül 2004.
- Radyasyon Tesislerine ve Radyasyon Uygulamalarına İlişkin Yetkilendirmeler Yönetmeliği. 2020. T.C. Resmi Gazete, 17 Aralık 2020.
- Romanov, G.N., Nikipelov, B.V., Drozhko, E.G. 1991. The Kyshtym Accident: Causes, Scale and Radiation Characteristics (EUR--13574(V1)). Commission of the European Communities (CEC).
- Talapko, J., Talapko, D., Katalinić, D., Kotris, I., Erić, I., Belić, D., Vasilj Mihaljević, M., Vasilj, A., Erić, S., Flam, J., Bekić, S., Matić, S., Škrlec, I. 2024. Health Effects of Ionizing Radiation on the Human Body. Medicina (Kaunas, Lithuania), 60(4): 653. <https://doi.org/10.3390/medicina60040653>.
- Tanır, A.G., Bölükdemir, M.H., Koç, K. 2013. Radyasyon ve Radyasyondan Korunma Fiziği. Palme Yayıncılık, Ankara.
- Türkiye Enerji, Nükleer ve Maden Araştırma Kurumu (TENMAK) <https://www.tenmak.gov.tr/>

- UNSCEAR. 2008. Report of the United Nations Scientific Committee on the Effects of Atomic Radiation.
- Wang, H., Yu, H., Sun, Q. 2016. Individual Monitoring and Occupational Dose Record Management in China: History, Current Status and Perspectives. *Int. J. Environ. Res. Public Health*, 13: 558.
- Yee, Y.K., Mohamad, M., Zuber, S.H., Saparudin, A.K.M., Kadir, A.B.A., Fadzil, M.S.A. 2024. Web-based Personal Dose Records for Occupational Radiation Exposure Monitoring System: Development and Pilot Testing. *Transactions on Science and Technology*, 11(4): 235-241.
- Visual analysis of geospatial multivariate data for investigating radioactive deposition processes. Takahashi, S., Sakurai, D., Sasaki, M., Miyamura, H.N., Sanada, Y. 2021.

8. EKLER

Bu tez çalışması kapsamında Python programlama dili ile geliştirilen kodlar bu bölümde verilmiştir.

db_setup.py

```
import sqlite3
import os

DATABASE_NAME = 'radyasyon_takip.db'

def setup_database():
    """
    Veritabanını ve gerekli tabloları oluşturur.
    """
    conn = None
    try:
        # Veritabanı dosyasının yolunu belirle
        script_dir = os.path.dirname(os.path.abspath(__file__))
        db_path = os.path.join(script_dir, DATABASE_NAME)

        conn = sqlite3.connect(db_path)
        cursor = conn.cursor()

        # Bireyler tablosu
        cursor.execute("""
            CREATE TABLE IF NOT EXISTS individuals (
                tc_no TEXT PRIMARY KEY,
                ad TEXT NOT NULL,
                soyad TEXT NOT NULL,
                cinsiyet TEXT,
                hamilelik_durumu TEXT,
                kurum_adi TEXT,
                departman TEXT,
                calisma_kosulu TEXT
            )
        """)

        # Kurumlar tablosu
        cursor.execute("""
            CREATE TABLE IF NOT EXISTS institutions (
                id INTEGER PRIMARY KEY AUTOINCREMENT,
                name TEXT NOT NULL UNIQUE
            )
        """)
```

```

# Doz Girişleri tablosu
cursor.execute("""
    CREATE TABLE IF NOT EXISTS dose_entries (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    tc_no TEXT NOT NULL,
    yıl INTEGER NOT NULL,
    doz_1_2aylik REAL,
    doz_2_2aylik REAL,
    doz_3_2aylik REAL,
    doz_4_2aylik REAL,
    doz_5_2aylik REAL,
    doz_6_2aylik REAL,
    yıllık_toplam_hp10 REAL,
    yıllık_hp007 REAL,
    yıllık_hp3 REAL,
    FOREIGN KEY (tc_no) REFERENCES individuals (tc_no) ON DELETE
    CASCADE
    )
    """)
conn.commit()
print(f'Veritabanı '{DATABASE_NAME}' ve tablolar başarıyla
oluşturuldu/güncellendi.")
except sqlite3.Error as e:
print(f'Veritabanı kurulum hatası: {e}')
finally:
if conn:
conn.close()

if __name__ == "__main__":
setup_database()

```

alara.py

```
print("DEBUG: alara.py başlatılıyor - Versiyon 4")
```

```
import sqlite3
```

```
import datetime
```

```
import data_operations # data_operations'ini import et
```

```
def calculate_alara_limit(cinsiyet, hamilelik_durumu, calisma_kosulu, yas):
```

```
    """
```

Bireyin ALARA (As Low As Reasonably Achievable) doz limitini hesaplar.

Bu değerler, ulusal ve uluslararası radyasyon güvenliği standartlarına dayanır.

Tüm limitler mSv cinsindendir.

```

"""
# Çalışma koşullarına göre temel ALARA limitleri (Örnek değerler, gerçek
düzenlemelere göre ayarlanmalı)
# Not: Gerçek ALARA limitleri, kurumun kendi iç politikalarına ve risk
değerlendirmelerine göre belirlenir.
# Buradaki değerler sadece örnek teşkil eder ve yasal limitlerin altında olmalıdır.

# 20 mSv'lik yasal limitin altında ALARA seviyeleri
base_limit = 0.0 # Varsayılan değer
ifcalisma_kosulu == 'A': # Yüksek maruziyet potansiyeli (örn. Nükleer Tıp, Girişimsel
Radyoloji)
base_limit = 15.0 # mSv/yıl (Yasal 20 mSv'nin altında tutulur)
elifcalisma_kosulu == 'B': # Orta maruziyet potansiyeli (örn. Tanısal Radyoloji)
base_limit = 5.0 # mSv/yıl (Yasal 6 mSv'nin altında tutulur)
elifcalisma_kosulu == 'C': # Düşük maruziyet potansiyeli (örn. İdare, Nadiren Girenler)
base_limit = 0.5 # mSv/yıl (Halk için limit olan 1 mSv'nin altında tutulur)
else:
    # Geçersiz koşul, varsayılan veya hata
    base_limit = 1.0 # mSv/yıl (Örnek)
    print(f'Uyarı: Geçersiz çalışma koşulu '{calisma_kosulu}'. Varsayılan limit
    kullanılıyor.")

# Hamilelik durumuna göre ayarlamalar
if cinsiyet == 'Kadın' andhamilelik_durumu:
    # Hamile kadınlar için tüm gebelik dönemi boyunca karın yüzeyine gelen eşdeğer
    doz limiti 1 mSv'dir.
    # Bu, mesleki maruziyetin halk limiti seviyesine indirilmesi anlamına gelir.
    return 1.0 # mSv/yıl (Tüm gebelik dönemi için)

# Yaşa göre ayarlamalar (Özellikle 18 yaş altı için)
# Türkiye'de 18 yaşından küçükler radyasyon alanında çalıştırılmaz.
# Bu kısım daha çok bir kaza anında veya teorik bir senaryoda kullanılabilir.
if yas < 18:
    # 18 yaş altı için maruziyet limitleri genellikle halk limiti ile aynıdır.
    returnmin(base_limit, 1.0) # Yasal limit 1 mSv/yıl, ALARA ilkesi gereği daha düşük de
    olabilir.

returnbase_limit

defcheck_and_notify_threshold(tc_kimlik_no, current_annual_dose, alara_limit):
    """
    Bireyin yıllık toplam dozunu kontrol eder ve eşik değerini aşarsa bildirim oluşturur.
    """
    ifcurrent_annual_dose is None:
        return # Doz kaydı yoksa kontrol etme

    ifcurrent_annual_dose>alara_limit:

```

```

print(f'UYARI: Birey TC Kimlik No {tc_kimlik_no} için toplam doz
({current_annual_dose:.2f} mSv), bireysel ALARA limitini ({alara_limit} mSv)
aşmıştır!')

# Daha önce bu aşım için aktif bir bildirim var mı kontrol et
# Bildirimler tablosunda cozuldu_mu = 0 (çözülmedi) olanları kontrol et
conn = sqlite3.connect(data_operations.DATABASE_NAME) #
data_operations'tanDATABASE_NAME'i kullan
cursor = conn.cursor()
cursor.execute("""
    SELECT bildirim_id FROM bildirimler
    WHERE tc_kimlik_no= ? AND cozuldu_mu = 0
    AND bildirim_mesajiLIKE ? || '%'
    """, (tc_kimlik_no, f'Yıllık doz aşımı ({datetime.date.today().year} yılı')) # Yıla
özel mesaj kontrolü
existing_notification = cursor.fetchone()
conn.close()

if not existing_notification:
    # Yeni bildirim oluştur
    notification_message = f'Yıllık doz aşımı ({datetime.date.today().year} yılı):
    {current_annual_dose:.2f} mSv (Limit: {alara_limit:.2f} mSv)'
    notification_data = {
        'tc_kimlik_no': tc_kimlik_no,
        'bildirim_tarihi': datetime.datetime.now().strftime('%Y-%m-%d
        %H:%M:%S'),
        'bildirim_mesaji': notification_message,
        'cozuldu_mu': 0 # 0: Çözülmedi
    }
    if data_operations.insert_notification(notification_data):
        print(f'Bildirim başarıyla oluşturuldu: Birey TC {tc_kimlik_no}, Mesaj:
        {notification_message}')
    else:
        print(f'Hata: Birey TC {tc_kimlik_no} için bildirim oluşturulurken hata oluştu.')
    else:
        print(f'Birey TC {tc_kimlik_no} için aktif bir aşım bildirimi zaten mevcut. Mevcut
        bildirim güncellenmiyor.')
        # İsteğe bağlı: Mevcut bildirimi güncelleyebilirsiniz (örn. en son doz değeriyle)
        # data_operations.update_notification_message(existing_notification[0],
        new_message)

if __name__ == "__main__":
    # Testler
    print(f'Erkek, Hamile değil, Koşul A, 30 yaş: {calculate_alara_limit('Erkek', False, 'A',
    30)} mSv')
    print(f'Kadın, Hamile değil, Koşul B, 25 yaş: {calculate_alara_limit('Kadın', False, 'B',
    25)} mSv')

```

```
print(f'Kadın, Hamile, Koşul A, 30 yaş: {calculate_alara_limit('Kadın', True, 'A', 30)} mSv")
print(f'Erkek, Hamile değil, Koşul C, 16 yaş: {calculate_alara_limit('Erkek', False, 'C', 16)} mSv")
print(f'Erkek, Hamile değil, Geçersiz Koşul, 40 yaş: {calculate_alara_limit('Erkek', False, 'X', 40)} mSv")
```

data_operations.py

```
import sqlite3
import os
import pandas as pd # Excel okuma için

DATABASE_NAME = 'radyasyon_takip.db'

def get_db_connection():
    """Veritabanı bağlantısı döndürür."""
    script_dir = os.path.dirname(os.path.abspath(__file__))
    db_path = os.path.join(script_dir, DATABASE_NAME)
    conn = sqlite3.connect(db_path)
    conn.row_factory = sqlite3.Row # Sütun isimleriyle erişim için
    return conn

# --- Birey İşlemleri ---
def add_individual(tc_no, ad, soyad, cinsiyet, hamilelik_durumu, kurum_adi, departman, calisma_kosulu):
    """Yeni bir bireyi veritabanına ekler."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute("""
            INSERT INTO individuals (tc_no, ad, soyad, cinsiyet, hamilelik_durumu,
            kurum_adi, departman, calisma_kosulu)
            VALUES (?, ?, ?, ?, ?, ?, ?, ?)
            """, (tc_no, ad, soyad, cinsiyet, hamilelik_durumu, kurum_adi, departman, calisma_kosulu))
        conn.commit()
        return True
    except sqlite3.IntegrityError:
        # TC Kimlik No zaten varsa
        return False
    except sqlite3.Error as e:
        print(f'Birey ekleme hatası: {e}')
        return False
```

```
finally:
conn.close()

defupdate_individual(tc_no, new_ad, new_soyad, new_cinsiyet,
new_hamilelik_durumu, new_kurum_adi, new_departman, new_calisma_kosulu):
    """Birey bilgilerini günceller."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute("""
            UPDATE individuals SET
            ad= ?, soyad= ?, cinsiyet = ?, hamilelik_durumu= ?,
            kurum_adi= ?, departman = ?, calisma_kosulu= ?
            WHERE tc_no= ?
            """, (new_ad, new_soyad, new_cinsiyet, new_hamilelik_durumu, new_kurum_adi,
            new_departman, new_calisma_kosulu, tc_no))
        conn.commit()
        return cursor.rowcount > 0 # Güncelleme yapıldıysa True döner
    except sqlite3.Error as e:
        print(f'Birey güncelleme hatası: {e}')
    return False
finally:
conn.close()

defdelete_individual(tc_no):
    """Bireyi ve ilgili tüm doz girişlerini siler."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute('DELETE FROM individuals WHERE tc_no= ?', (tc_no,))
        conn.commit()
        return cursor.rowcount > 0 # Silme yapıldıysa True döner
    except sqlite3.Error as e:
        print(f'Birey silme hatası: {e}')
    return False
finally:
conn.close()

defget_individual_by_tc_no(tc_no):
    """TC Kimlik Numarasına göre birey bilgilerini getirir."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute('SELECT * FROM individuals WHERE tc_no= ?', (tc_no,))
        individual = cursor.fetchone()
        return individual
    except sqlite3.Error as e:
        print(f'Birey sorgulama hatası: {e}')
```

```
returnNone
finally:
conn.close()

defget_all_individuals():
    """Tüm bireylerin bilgilerini getirir."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute('SELECT * FROM individuals')
        individuals = cursor.fetchall()
        returnindividuals
    except sqlite3.Error as e:
        print(f'Tüm bireyleri listeleme hatası: {e}')
    return []
    finally:
        conn.close()

# --- Kurum İşlemleri ---
defadd_institution(name):
    """Yeni bir kurumu veritabanına ekler."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute('INSERT INTO institutions (name) VALUES (?)', (name,))
        conn.commit()
        return True
    except sqlite3.IntegrityError:
        returnFalse # Kurum adı zaten mevcut
    except sqlite3.Error as e:
        print(f'Kurum ekleme hatası: {e}')
        returnFalse
    finally:
        conn.close()

defget_institution_by_name(name):
    """Kurum adına göre kurum bilgilerini getirir."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute('SELECT * FROM institutions WHERE name = ?', (name,))
        institution = cursor.fetchone()
        returninstitution
    except sqlite3.Error as e:
        print(f'Kurum sorgulama hatası: {e}')
        returnNone
    finally:
        conn.close()
```

```
def get_all_institutions():
    """Tüm kurumların bilgilerini getirir."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute('SELECT * FROM institutions ORDER BY name ASC')
        institutions = cursor.fetchall()
        return institutions
    except sqlite3.Error as e:
        print(f'Tüm kurumları listeleme hatası: {e}')
    return []
    finally:
        conn.close()

def update_institution(old_name, new_name):
    """Kurum adını günceller."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute('UPDATE institutions SET name = ? WHERE name = ?', (new_name,
            old_name))
        conn.commit()
        return cursor.rowcount > 0
    except sqlite3.IntegrityError:
        return False # Yeni isim zaten mevcut
    except sqlite3.Error as e:
        print(f'Kurum güncelleme hatası: {e}')
    return False
    finally:
        conn.close()

def delete_institution(name):
    """Kurumu siler."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        cursor.execute('DELETE FROM institutions WHERE name = ?', (name,))
        conn.commit()
        return cursor.rowcount > 0
    except sqlite3.Error as e:
        print(f'Kurum silme hatası: {e}')
    return False
    finally:
        conn.close()

# --- Doz İşlemleri ---
```

```

defadd_dose_entry(tc_no, yil, doz_periyotlari, yillik_toplam_hp10, yillik_hp007,
yillik_hp3):
    """Bir birey için doz girişi ekler veya günceller."""
    conn = get_db_connection()
    cursor = conn.cursor()
    try:
        # Önce aynı yıl için daha önce bir giriş olup olmadığını kontrol et
        cursor.execute('SELECT id FROM dose_entries WHERE tc_no= ? AND yil= ?', (tc_no,
yil))
        existing_entry = cursor.fetchone()

    ifexisting_entry:
        # Eğer varsa, güncelle
        cursor.execute("""
            UPDATE dose_entries SET
            doz_1_2aylik = ?, doz_2_2aylik = ?, doz_3_2aylik = ?,
            doz_4_2aylik = ?, doz_5_2aylik = ?, doz_6_2aylik = ?,
            yillik_toplam_hp10 = ?, yillik_hp007 = ?, yillik_hp3 = ?
            WHERE tc_no= ? AND yil= ?
            """, (doz_periyotlari.get('Doz 1 (2 aylık)'), doz_periyotlari.get('Doz 2 (2 aylık)'),
            doz_periyotlari.get('Doz 3 (2 aylık)'), doz_periyotlari.get('Doz 4 (2 aylık)'),
            doz_periyotlari.get('Doz 5 (2 aylık)'), doz_periyotlari.get('Doz 6 (2 aylık)'),
            yillik_toplam_hp10, yillik_hp007, yillik_hp3,
            tc_no, yil))
        else:
            # Yoksa, yeni kayıt ekle
            cursor.execute("""
                INSERT INTO dose_entries (tc_no, yil, doz_1_2aylik, doz_2_2aylik,
            doz_3_2aylik,
            doz_4_2aylik, doz_5_2aylik, doz_6_2aylik,
            yillik_toplam_hp10, yillik_hp007, yillik_hp3)
                VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
                """, (tc_no, yil, doz_periyotlari.get('Doz 1 (2 aylık)'), doz_periyotlari.get('Doz 2 (2
aylık)'),
            doz_periyotlari.get('Doz 3 (2 aylık)'), doz_periyotlari.get('Doz 4 (2 aylık)'),
            doz_periyotlari.get('Doz 5 (2 aylık)'), doz_periyotlari.get('Doz 6 (2 aylık)'),
            yillik_toplam_hp10, yillik_hp007, yillik_hp3))
            conn.commit()
            return True
        except sqlite3.Error as e:
            print(f'Doz girişi hatası: {e}')
            returnFalse
        finally:
            conn.close()

defget_dose_entries_by_tc_no(tc_no):
    """Bir bireyin tüm doz girişlerini getirir."""
    conn = get_db_connection()

```

```

cursor = conn.cursor()
try:
    cursor.execute('SELECT * FROM dose_entries WHERE tc_no= ? ORDER BY yıl
    ASC', (tc_no,))
    doses = cursor.fetchall()
    returndoses
except sqlite3.Error as e:
    print(f"Doz girişleri sorgulama hatası: {e}")
    return []
finally:
    conn.close()

def import_excel_data_to_db():
    """
    Oluşturulan Excel raporundaki verileri uygulamanın veritabanına aktarır.
    """
    script_dir = os.path.dirname(os.path.abspath(__file__))
    # 'reports' klasörünün yolunu oluştur (main.py ile aynı seviyede varsayılıyor)
    reports_dir = os.path.join(script_dir, 'reports')
    output_filename = os.path.join(reports_dir, 'radyasyon_doz_raporu.xlsx')

    if not os.path.exists(output_filename):
        print(f"\n--- Hata: Radyasyon Doz Raporu Bulunamadı ---")
        print(f'Rapor dosyası '{output_filename}' bulunamadı. Lütfen önce '4. Raporlar ve
        Bildirimler' menüsünden '1. Örnek Excel Raporu Oluştur' seçeneğini kullanarak raporu
        oluşturduğunuzdan emin olun.")
        input("Devam etmek için Enter'a basın...")
        return

    try:
        df = pd.read_excel(output_filename, sheet_name='Yıllık Doz Detayı')

        print("\nExcel'deki veriler veritabanına aktarılıyor...")

        # Her bir birey için veritabanına ekleme yap
        # TC Kimlik No'ya göre benzersiz bireyleri al
        unique_individuals_df = df.drop_duplicates(subset=['TC Kimlik No'])

        for index, individual_row in unique_individuals_df.iterrows():
            tc_no = str(individual_row['TC Kimlik No']) # TC Kimlik No'yu string olarak al
            ad = individual_row['Adı']
            soyad = individual_row['Soyadı']
            cinsiyet = individual_row['Cinsiyet']
            hamilelik_durumu = individual_row['Hamilelik Durumu']
            kurum_adi = individual_row['Kurum Adı']
            departman = individual_row['Departman']
            calisma_kosulu = individual_row['Çalışma Koşulu']

```

```

        # Kurumu ekle (zaten varsa eklemeyecek)
if not get_institution_by_name(kurum_adi):
    add_institution(kurum_adi)
print(f'Kurum eklendi: {kurum_adi}')

        # Bireyi ekle (zaten varsa eklemeyecek)
if not get_individual_by_tc_no(tc_no):
    add_individual(tc_no, ad, soyad, cinsiyet, hamilelik_durumu, kurum_adi, departman,
        calisma_kosulu)
    print(f'Birey eklendi: {ad} {soyad} ({tc_no})')
else:
    print(f'Birey zaten mevcut, atlanıyor: {ad} {soyad} ({tc_no})')

        # Bireyin tüm yıllara ait doz girişlerini filtrele
individual_doses = df[df['TC Kimlik No'] == individual_row['TC Kimlik No']]

for _, dose_row in individual_doses.iterrows():
    yil = int(dose_row['Yıl'])
    yıllık_hp10 = float(dose_row['Yıllık Toplam Hp(10) (mSv)'])
    yıllık_hp007 = float(dose_row['Yıllık Toplam Hp(0.07)'])
    yıllık_hp3 = float(dose_row['Yıllık Toplam Hp(3)'])

    doz_periyotlari = {}
    for i in range(1, 7):
        doz_periyotlari[f'Doz {i} (2 aylık)'] = float(dose_row[f'Doz {i} (2 aylık)'])

    add_dose_entry(tc_no, yil, doz_periyotlari, yıllık_hp10, yıllık_hp007, yıllık_hp3)
    print(f' - {yil} yılı için doz girişi eklendi/güncellendi: {yillik_hp10} mSv")

print("\nVeriler başarıyla veritabanına aktarıldı!")
input("Devam etmek için Enter'a basın...")

except FileNotFoundError:
    print(f"\nHata: Rapor dosyası '{output_filename}' bulunamadı. Lütfen önce raporu
        oluşturduğunuzdan emin olun.")
    input("Devam etmek için Enter'a basın...")
except Exception as e:
    print(f"\nVeri aktarılırken bir hata oluştu: {e}")
    input("Devam etmek için Enter'a basın...")

```

institution_operations.py

```
from data_operations import add_institution, get_institution_by_name,
get_all_institutions, update_institution, delete_institution
```

```
def add_institution_menu():
    """Yeni kurum ekleme menüsü."""
    print("\n--- Yeni Kurum Ekle ---")
    name = input("Kurum Adı: ")
    if add_institution(name):
        print("Kurum başarıyla eklendi.")
    else:
        print("Kurum eklenirken bir hata oluştu veya bu isimde bir kurum zaten mevcut.")
    input("Devam etmek için Enter'a basın...")
```

```
def update_institution_menu():
    """Kurum bilgilerini güncelleme menüsü."""
    print("\n--- Kurum Bilgilerini Güncelle ---")
    old_name = input("Güncellenecek kurumun mevcut adı: ")
    institution = get_institution_by_name(old_name)
    if not institution:
        print("Bu isimde bir kurum bulunamadı.")
    input("Devam etmek için Enter'a basın...")
    return
```

```
new_name = input(f"Yeni Kurum Adı ( {institution['name']} ): ")
if update_institution(old_name, new_name):
    print("Kurum adı başarıyla güncellendi.")
else:
    print("Kurum adı güncellenirken bir hata oluştu veya yeni isim zaten mevcut.")
    input("Devam etmek için Enter'a basın...")
```

```
def list_all_institutions_menu():
    """Tüm kurumları listeleme menüsü."""
    print("\n--- Tüm Kurumlar ---")
    institutions = get_all_institutions()
    if not institutions:
        print("Kayıtlı kurum bulunmamaktadır.")
    else:
        for inst in institutions:
            print(f"ID: {inst['id']}, Adı: {inst['name']}")
        input("Devam etmek için Enter'a basın...")
```

```
def delete_institution_menu():
    """Kurum silme menüsü."""
    print("\n--- Kurum Sil ---")
    name = input("Silinecek kurumun adı: ")
    institution = get_institution_by_name(name)
    if not institution:
```

```
print("Bu isimde bir kurum bulunamadı.")
input("Devam etmek için Enter'a basın...")
return
```

```
confirm = input(f'{institution['name']}' adlı kurumu silmek istediğinizden emin
misiniz? (Evet/Hayır): ')
ifconfirm.lower() == 'evet':
ifdelete_institution(name):
print("Kurum başarıyla silindi.")
else:
print("Kurum silinirken bir hata oluştu.")
else:
print("Silme işlemi iptal edildi.")
input("Devam etmek için Enter'a basın...")
```

individual_operations.py

```
fromdata_operationsimportadd_individual, get_individual_by_tc_no,
get_all_individuals, update_individual, delete_individual
```

```
defnew_individual_menu():
    """Yeni birey ekleme menüsü."""
    print("\n--- Yeni Birey Ekle ---")
    tc_no = input("TC Kimlik No: ")
    ifget_individual_by_tc_no(tc_no):
        print("Bu TC Kimlik Numarasına sahip bir birey zaten mevcut.")
        input("Devam etmek için Enter'a basın...")
        return
```

```
ad = input("Adı: ")
soyad = input("Soyadı: ")
cinsiyet = input("Cinsiyet (Erkek/Kadın): ")
hamilelik_durumu = input("Hamilelik Durumu (Evet/Hayır) (Kadınlar için): ")
kurum_adi = input("Kurum Adı: ")
departman = input("Departman: ")
calisma_kosulu = input("Çalışma Koşulu (A/B/C): ")
```

```
ifadd_individual(tc_no, ad, soyad, cinsiyet, hamilelik_durumu, kurum_adi, departman,
calisma_kosulu):
    print("Birey başarıyla eklendi.")
else:
    print("Birey eklenirken bir hata oluştu.")
    input("Devam etmek için Enter'a basın...")
```

```

defupdate_individual_menu():
    """Birey bilgilerini güncelleme menüsü."""
    print("\n--- Birey Bilgilerini Güncelle ---")
    tc_no = input("Güncellenecek bireyin TC Kimlik No: ")
    individual = get_individual_by_tc_no(tc_no)
    if not individual:
        print("Bu TC Kimlik Numarasına sahip birey bulunamadı.")
        input("Devam etmek için Enter'a basın...")
        return

    print(f'Mevcut Bilgiler: Adı: {individual['ad']}, Soyadı: {individual['soyad']}, Kurum: {individual['kurum_adi']}')

    new_ad = input(f'Yeni Adı ({individual['ad']}): ') or individual['ad']
    new_soyad = input(f'Yeni Soyadı ({individual['soyad']}): ') or individual['soyad']
    new_cinsiyet = input(f'Yeni Cinsiyet ({individual['cinsiyet']}): ') or individual['cinsiyet']
    new_hamilelik_durumu = input(f'Yeni Hamilelik Durumu ({individual['hamilelik_durumu']}): ') or individual['hamilelik_durumu']
    new_kurum_adi = input(f'Yeni Kurum Adı ({individual['kurum_adi']}): ') or individual['kurum_adi']
    new_departman = input(f'Yeni Departman ({individual['departman']}): ') or individual['departman']
    new_calisma_kosulu = input(f'Yeni Çalışma Koşulu ({individual['calisma_kosulu']}): ') or individual['calisma_kosulu']

    ifupdate_individual(tc_no, new_ad, new_soyad, new_cinsiyet, new_hamilelik_durumu, new_kurum_adi, new_departman, new_calisma_kosulu):
        print("Birey bilgileri başarıyla güncellendi.")
    else:
        print("Birey bilgileri güncellenirken bir hata oluştu.")
        input("Devam etmek için Enter'a basın...")

defquery_individual_menu():
    """Birey sorgulama menüsü."""
    print("\n--- Birey Sorgula ---")
    tc_no = input("Sorgulanacak bireyin TC Kimlik No: ")
    individual = get_individual_by_tc_no(tc_no)
    ifindividual:
        print("\n--- Birey Bilgileri ---")
        print(f'TC Kimlik No: {individual['tc_no']}')
        print(f'Adı Soyadı: {individual['ad']} {individual['soyad']}')
        print(f'Cinsiyet: {individual['cinsiyet']}')
        print(f'Hamilelik Durumu: {individual['hamilelik_durumu']}')
        print(f'Kurum Adı: {individual['kurum_adi']}')
        print(f'Departman: {individual['departman']}')
        print(f'Çalışma Koşulu: {individual['calisma_kosulu']}')
    else:

```

```

print("Bu TC Kimlik Numarasına sahip birey bulunamadı.")
input("Devam etmek için Enter'a basın...")

deflist_all_individuals_menu():
    """Tüm bireyleri listeleme menüsü."""
    print("\n--- Tüm Bireyler ---")
    individuals = get_all_individuals()
    if not individuals:
        print("Kayıtlı birey bulunmamaktadır.")
    else:
        for ind in individuals:
            print(f"TC: {ind['tc_no']}, Adı: {ind['ad']}, Soyadı: {ind['soyad']}, Kurum: {ind['kurum_adi']}, Departman: {ind['departman']}")
        input("Devam etmek için Enter'a basın...")

defdelete_individual_menu():
    """Birey silme menüsü."""
    print("\n--- Birey Sil ---")
    tc_no = input("Silinecek bireyin TC Kimlik No: ")
    individual = get_individual_by_tc_no(tc_no)
    if not individual:
        print("Bu TC Kimlik Numarasına sahip birey bulunamadı.")
        input("Devam etmek için Enter'a basın...")
        return

    confirm = input(f"{individual['ad']} {individual['soyad']} adlı bireyi ve tüm doz kayıtlarını silmek istediğinizden emin misiniz? (Evet/Hayır): ")
    if confirm.lower() == 'evet':
        ifdelete_individual(tc_no):
            print("Birey başarıyla silindi.")
        else:
            print("Birey silinirken bir hata oluştu.")
        else:
            print("Silme işlemi iptal edildi.")
        input("Devam etmek için Enter'a basın...")

```

dose_operations.py

```

from data_operations import get_individual_by_tc_no, add_dose_entry,
get_dose_entries_by_tc_no
import random # Yıllık hp007 ve hp3 değerleri için

defmanual_dose_entry_menu():
    """Manuel olarak doz girişi yapma menüsü."""

```

```

print("\n--- Manuel Doz Girişi ---")
tc_no = input("Doz girişi yapılacak bireyin TC Kimlik No: ")
individual = get_individual_by_tc_no(tc_no)
if not individual:
    print("Bu TC Kimlik Numarasına sahip birey bulunamadı. Lütfen önce bireyi ekleyin.")
    input("Devam etmek için Enter'a basın...")
    return

yil = int(input("Dozun girileceği yıl (örn: 2023): "))

doz_peryotlari = {}
total_hp10_yearly = 0.0
for i in range(1, 7):
    while True:
        try:
            doz_value = float(input(f"Doz {i} (2 aylık, mSv): "))
            if doz_value < 0:
                print("Doz değeri negatif olamaz. Lütfen pozitif bir değer girin.")
                continue
            doz_peryotlari[f"Doz {i} (2 aylık)"] = doz_value
            total_hp10_yearly += doz_value
            break
        except ValueError:
            print("Geçersiz giriş. Lütfen bir sayı girin.")

    # Hp(0.07) ve Hp(3) değerlerini Hp(10) ile akademik oranlarla hesapla
    # Bu oranlar, report.py dosyasındaki mantıkla tutarlı hale getirildi.
    if total_hp10_yearly > 0:
        hp007_multiplier = random.uniform(1.2, 2.5) # Hp(0.07) genellikle Hp(10)'dan daha
        yüksek olabilir (cilt dozu)
        hp3_multiplier = random.uniform(1.0, 1.8) # Hp(3) Hp(10)'a yakın veya biraz yüksek
        olabilir (göz merceği dozu)

    yıllık_hp007 = total_hp10_yearly * hp007_multiplier
    yıllık_hp3 = total_hp10_yearly * hp3_multiplier

    # Çok düşük Hp(10) değerleri için Hp(0.07) ve Hp(3) değerlerinin sıfır olmamasını
    veya
    # Hp(10)'dan çok düşük olmamasını sağla.
    yıllık_hp007 = max(yıllık_hp007, total_hp10_yearly * 0.8)
    yıllık_hp3 = max(yıllık_hp3, total_hp10_yearly * 0.8)
    else:
        yıllık_hp007 = 0.0
        yıllık_hp3 = 0.0

ifadd_dose_entry(tc_no, yil, doz_peryotlari, total_hp10_yearly, yıllık_hp007,
yıllık_hp3):

```

```

print(f'{yil} yılı için doz girişi başarıyla eklendi/güncellendi.')
else:
print("Doz girişi yapılırken bir hata oluştu.")
input("Devam etmek için Enter'a basın...")

defquery_dose_menu():
    """Bireyin doz kayıtlarını sorgulama menüsü."""
    print("\n--- Doz Kayıtlarını Sorgula ---")
    tc_no = input("Doz kayıtları sorgulanacak bireyin TC Kimlik No: ")
    individual = get_individual_by_tc_no(tc_no)
    if not individual:
        print("Bu TC Kimlik Numarasına sahip birey bulunamadı.")
        input("Devam etmek için Enter'a basın...")
        return

    doses = get_dose_entries_by_tc_no(tc_no)
    if not doses:
        print(f'{individual['ad']} {individual['soyad']} için kayıtlı doz bulunmamaktadır.')
    else:
        print(f"\n--- {individual['ad']} {individual['soyad']} ({individual['tc_no']}) için Doz Kayıtları ---")
        for dose in doses:
            print(f'Yıl: {dose['yil']}")
            print(f' Doz 1 (2 aylık): {dose['doz_1_2aylik']} mSv")
            print(f' Doz 2 (2 aylık): {dose['doz_2_2aylik']} mSv")
            print(f' Doz 3 (2 aylık): {dose['doz_3_2aylik']} mSv")
            print(f' Doz 4 (2 aylık): {dose['doz_4_2aylik']} mSv")
            print(f' Doz 5 (2 aylık): {dose['doz_5_2aylik']} mSv")
            print(f' Doz 6 (2 aylık): {dose['doz_6_2aylik']} mSv")
            print(f' Yıllık Toplam Hp(10): {dose['yillik_toplam_hp10']} mSv")
            print(f' YıllıkHp(0.07): {dose['yillik_hp007']} mSv")
            print(f' YıllıkHp(3): {dose['yillik_hp3']} mSv")
            print("-" * 30)
        input("Devam etmek için Enter'a basın...")

```

report.py

```

import pandas as pd
import random
from datetime import datetime, timedelta
import os
import openpyxl # Excel dosyalarını düzenlemek için eklendi
from openpyxl.styles import PatternFill, Font, Alignment # Koşullu biçimlendirme ve stil için eklendi

```

fromopenpyxl.formatting.ruleimportRule # Koşullu biçimlendirme için eklendi
 fromopenpyxl.styles.differentialimportDifferentialStyle # Koşullu biçimlendirme için eklendi

Sabitler

NUM_INDIVIDUALS = 50

NUM_YEARS = 5

DOSE_PERIODS_PER_YEAR = 6 # 2 aylık periyotlar

CUMULATIVE_DOSE_LIMIT_5_YEARS = 100 # mSv

ALARA Limitleri (Tez belgesinden alınmıştır)

ALARA_LIMITS = {

'A': 20, #mSv/yıl

'B': 5, # mSv/yıl

'C': 0.5 #mSv/yıl

}

ALARA_LIMIT_PREGNANT = 1 # mSv/yıl (Hamilelik dönemi için)

Hp(0.07) ve Hp(3) için akademik olarak temsil edici sabit çarpanlar

Bu çarpanlar, özellikle düşük enerjili radyasyon türlerinde cilt ve göz merceği

dozlarının derin dozdan daha yüksek olabileceği akademik bilgilere dayanır.

HP007_ACADEMIC_MULTIPLIER = 1.5

HP3_ACADEMIC_MULTIPLIER = 1.2

Sağlık Bakanlığı Sağlık İzni Tablosu (Çalışma Süresi (Saat) -> Sağlık İzni (Gün))

(min_saat, max_saat, izin_gün)

HEALTH_LEAVE_TABLE = [

(0, 50, 1),

(51, 100, 2),

(101, 150, 3),

(151, 200, 4),

(201, 250, 5),

(251, 300, 6),

(301, 350, 7),

(351, 400, 8),

(401, 450, 9),

(451, 500, 10),

(501, 550, 11),

(551, 600, 12),

(601, 650, 13),

(651, 700, 14),

(701, 750, 15),

(751, 800, 16),

(801, 850, 17),

(851, 900, 18),

(901, 950, 19),

(951, 1000, 20),

(1001, 1050, 21),

```
(1051, 1100, 22),
(1101, 1150, 23),
(1151, 1200, 24),
(1201, 1250, 25),
(1251, 1300, 26),
(1301, 1350, 27),
(1351, 1400, 28),
(1401, 1450, 29),
(1451, 1500, 30)
```

```
]
```

```
# Çalışma Süresi (Saat) simülasyonu için aralıklar (doz senaryosuna göre)
```

```
WORKING_HOURS_RANGES = {
```

```
    'normal': (500, 1000),    # Düşük dozlar için daha az çalışma saati
```

```
    'borderline': (900, 1300), # Orta dozlar için orta çalışma saati
```

```
    'excess': (1200, 1500)    # Yüksek dozlar için daha fazla çalışma saati
```

```
}
```

```
# Örnek isimler ve departmanlar (Cinsiyete göre ayrıldı)
```

```
MALE_NAMES = ["Mehmet", "Ali", "Can", "Burak", "Mert"]
```

```
FEMALE_NAMES = ["Ayşe", "Fatma", "Zeynep", "Elif", "Deniz"] # Deniz hem erkek  
hem kadın ismi olabilir, burada kadın olarak varsayıldı
```

```
LAST_NAMES = ["Yılmaz", "Demir", "Kara", "Aslan", "Öztürk", "Kaya", "Yıldırım",  
"Güneş", "Aksoy", "Tunç"]
```

```
DEPARTMENTS = ["Radyoloji", "Nükleer Tıp", "Radyoterapi", "PET/CT",  
"Laboratuvar", "Onkoloji"]
```

```
# Çalışma koşulları güncellendi: Sadece 'A' ve 'B'
```

```
WORK_CONDITIONS = ['A', 'B']
```

```
# Örnek kurum adları (Akdeniz Üniversitesi Hastanesi eklendi)
```

```
INSTITUTIONS = [
```

```
    "Akdeniz Üniversitesi Hastanesi",
```

```
    "Ankara Şehir Hastanesi",
```

```
    "İstanbul Üniversitesi Hastanesi",
```

```
    "Ege Üniversitesi Hastanesi",
```

```
    "Medipol Mega Üniversite Hastanesi",
```

```
    "Dokuz Eylül Üniversitesi Hastanesi",
```

```
    "Hacettepe Üniversitesi Hastanesi"
```

```
]
```

```
defgenerate_tc_kimlik_no():
```

```
    """Rastgele bir TC Kimlik Numarası oluşturur."""
```

```
    return ".join([str(random.randint(0, 9)) for _ in range(10)]) + str(random.randint(0, 9))
```

```
defcalculate_yearly_alara(work_condition, is_pregnant, gender):
```

```
    """Yıllık ALARA limitini çalışma koşulu ve hamilelik durumuna göre hesaplar."""
```

```
    ifgender == 'Kadın' andis_pregnant:
```

```

return ALARA_LIMIT_PREGNANT
return ALARA_LIMITS.get(work_condition, 20) # Varsayılan olarak A koşulu limiti

def calculate_extra_leave(dose_excess_amount):
    """Aşım miktarına göre ek izin süresini hesaplar.
    Bu fonksiyon, aşım miktarına göre kademeli olarak farklı ek izin günleri döndürür.
    """
    if dose_excess_amount <= 0:
        return 0
    elif dose_excess_amount < 10:
        return 0 # Küçük aşım için ek izin yok
    elif 10 <= dose_excess_amount < 25:
        return 1
    elif 25 <= dose_excess_amount < 50:
        return 2
    elif 50 <= dose_excess_amount < 75:
        return 3
    else: # dose_excess_amount >= 75
        # Daha büyük aşım için, 4 gün taban izin + her ek 25 mSv için 1 gün daha
        return 4 + int((dose_excess_amount - 75) / 25)

def calculate_health_leave(working_hours):
    """Çalışma süresine göre sağlık izni süresini hesaplar (Sağlık Bakanlığı Tablosuna göre)."""
    for min_h, max_h, leave_days in HEALTH_LEAVE_TABLE:
        if min_h <= working_hours <= max_h:
            return leave_days
    # Eğer çalışma süresi tablodaki aralıkların dışındaysa
    if working_hours < HEALTH_LEAVE_TABLE[0][0]:
        return 0 # Minimumdan az ise izin yok
    elif working_hours > HEALTH_LEAVE_TABLE[-1][1]:
        return HEALTH_LEAVE_TABLE[-1][2] # Maksimumdan fazla ise maksimum izin
    return 0 # Varsayılan olarak 0 gün

def generate_individual_data_for_excel(individual_id):
    """Tek bir birey için 5 yıllık doz verilerini ve ilgili bilgileri oluşturur."""
    tc_no = generate_tc_kimlik_no()
    last_name = random.choice(LAST_NAMES)
    gender = random.choice(["Erkek", "Kadın"])

    # Cinsiyete göre isim seçimi
    if gender == 'Erkek':
        first_name = random.choice(MALE_NAMES)
    else:
        first_name = random.choice(FEMALE_NAMES)

    department = random.choice(DEPARTMENTS)

```

```
work_condition = random.choice(WORK_CONDITIONS) # Sadece 'A' ve 'B' arasından
seçim yapılacaktır
```

```
# Kurum adı, birey başına bir kez belirlenir
institution_name = random.choice(INSTITUTIONS)
```

```
# Hamilelik durumunu rastgele belirle (sadece kadınlar için)
is_pregnant = False
ifgender == 'Kadın' and random.random() < 0.15: # %15 ihtimalle hamile
is_pregnant = True
```

```
# Yaş bilgisi eklendi
age = random.randint(22, 65) # Gerçekçi bir yaş aralığı
```

```
# Doz senaryosu belirleme (Doz Aşımı, Sınırdaki, Limit Aşılması)
scenario_type = random.choices(
    ['excess', 'borderline', 'normal'],
    weights=[0.4, 0.3, 0.3], # Oranlar: %40 aşım, %30 sınırdaki, %30 normal (daha fazla aşım
ve sınırdaki durumu için ayarlandı)
k=1
)[0]
```

```
yearly_data = []
cumulative_hp10_5_years = 0 # Hp(10) için kümülatif toplam
# Hp(0.07) ve Hp(3) için Python'da kümülatif toplamaları takip etmek için değişkenler
cumulative_hp007_5_years = 0
cumulative_hp3_5_years = 0
```

```
# Yıllık bazda aşım olup olmadığını kontrol etmek için bir bayrak
any_yearly_excess_occurred = False
total_working_hours_5_years_individual = 0 # Bireyin 5 yıllık toplam çalışma süresi
```

```
for year_offset in range(NUM_YEARS):
current_year = 2019 + year_offset
```

```
# Yıllık ALARA limitini hesapla
yearly_alara_limit = calculate_yearly_alara(work_condition, is_pregnant, gender)
```

```
# Yıllık toplam dozu senaryoya göre belirle
if scenario_type == 'excess':
avg_yearly_dose = random.uniform(21, 45)
elif scenario_type == 'borderline':
avg_yearly_dose = random.uniform(18, 22)
else: # normal
avg_yearly_dose = random.uniform(1, 18)
```

```
# Hamilelik durumunda yıllık dozu düşür
if is_pregnant and gender == 'Kadın':
```

```
avg_yearly_dose = min(avg_yearly_dose, yearly_alara_limit + random.uniform(0, 1.0))
avg_yearly_dose = min(avg_yearly_dose, 5.0)
```

```
# 2 aylık periyotlar için dozları dağıt
bi_monthly_doses_hp10 = []
bi_monthly_doses_hp007 = []
bi_monthly_doses_hp3 = []
```

```
for _ in range(DOSE_PERIODS_PER_YEAR):
    # Hp(10) dozunu simüle et
    hp10_dose_period = max(0, random.gauss(avg_yearly_dose /
    DOSE_PERIODS_PER_YEAR, (avg_yearly_dose / DOSE_PERIODS_PER_YEAR) *
    0.2))
    bi_monthly_doses_hp10.append(hp10_dose_period)
```

```
# Hp(0.07) dozunu Hp(10) ile ilişkili olarak ve varyasyonla simüle et
# Hp(10) sıfırda Hp(0.07) de sıfır olsun, aksi takdirde çarpan ve varyasyon
uygula
hp007_dose_period = hp10_dose_period * HP007_ACADEMIC_MULTIPLIER *
random.uniform(0.8, 1.2) if hp10_dose_period > 0 else 0
bi_monthly_doses_hp007.append(hp007_dose_period)
```

```
# Hp(3) dozunu Hp(10) ile ilişkili olarak ve varyasyonla simüle et
# Hp(10) sıfırda Hp(3) de sıfır olsun, aksi takdirde çarpan ve varyasyon uygula
hp3_dose_period = hp10_dose_period * HP3_ACADEMIC_MULTIPLIER *
random.uniform(0.8, 1.2) if hp10_dose_period > 0 else 0
bi_monthly_doses_hp3.append(hp3_dose_period)
```

```
# Yıllık toplam Hp(10) dozunu hesapla
total_hp10_yearly = sum(bi_monthly_doses_hp10)
# Yıllık toplam Hp(0.07) dozunu hesapla
total_hp007_yearly = sum(bi_monthly_doses_hp007)
# Yıllık toplam Hp(3) dozunu hesapla
total_hp3_yearly = sum(bi_monthly_doses_hp3)
```

```
# Çalışma süresi simülasyonu (doz senaryosu ile ilişkili)
min_hours, max_hours = WORKING_HOURS_RANGES[scenario_type]
total_working_hours_yearly = random.randint(min_hours, max_hours)
total_working_hours_5_years_individual += total_working_hours_yearly # 5 yıllık
toplam çalışma süresini topla
```

```
# Yıllık aşım durumu kontrolü (Hp(10) baz alınarak)
yearly_excess_status = 'Aşım Var' if total_hp10_yearly > yearly_alara_limit else 'Aşım
Yok'
ifyearly_excess_status == 'Aşım Var':
    any_yearly_excess_occurred = True # Yıllık aşım varsa bayrağı ayarla
```

```

# Yıllık aşım miktarı hesaplaması (Bu değer Excel'e yazılmayacak ama
hesaplamada kullanılabilir)
yearly_excess_amount = max(0, total_hp10_yearly - yearly_alara_limit)

cumulative_hp10_5_years += total_hp10_yearly
cumulative_hp007_5_years += total_hp007_yearly # Python'da kümülatif Hp(0.07)
toplamı
cumulative_hp3_5_years += total_hp3_yearly # Python'da kümülatif Hp(3) toplamı

yearly_data.append({
    'Yıl': current_year,
    'TC Kimlik No': tc_no,
    'Adı': first_name,
    'Soyadı': last_name,
    'Cinsiyet': gender,
    'Yaş': age, # Yaş bilgisi eklendi
    'Hamilelik Durumu': 'Evet' if is_pregnant else 'Hayır',
    'Kurum Adı': institution_name,
    'Departman': department,
    'Çalışma Koşulu': work_condition,
    'Yıllık ALARA Limiti (mSv)': yearly_alara_limit,
    **{f'Doz {i+1} Hp(10) (2 aylık)': round(d, 2) for i, d in
enumerate(bi_monthly_doses_hp10)},
    **{f'Doz {i+1} Hp(0.07) (2 aylık)': round(d, 2) for i, d in
enumerate(bi_monthly_doses_hp007)},
    **{f'Doz {i+1} Hp(3) (2 aylık)': round(d, 2) for i, d in
enumerate(bi_monthly_doses_hp3)},
    'Yıllık Toplam Hp(10) (mSv)': round(total_hp10_yearly, 2),
    'Yıllık Aşım Durumu': yearly_excess_status,
    # 'Yıllık Aşım Miktarı (mSv)': round(yearly_excess_amount, 2), # Bu sütun
kaldırıldı
    'Yıllık Toplam Hp(0.07)': round(total_hp007_yearly, 2),
    'Yıllık Toplam Hp(3)': round(total_hp3_yearly, 2),
    'Çalışma Süresi (Saat)': total_working_hours_yearly,
    '5 Yıllık Hp(10) Toplamı (mSv)': 0,
    '5 Yıllık Hp(10) Ortalaması (mSv)': 0,
    'Aşım Durumu': '',
    # 'Aşım Miktarı (mSv)': 0, # Bu sütun kaldırıldı
    'Önerilen Sağlık İzni (Gün)': 0
})

# 5 yıllık kümülatif doz, aşım durumu ve izinleri hesapla
dose_excess_5_years = cumulative_hp10_5_years -
CUMULATIVE_DOSE_LIMIT_5_YEARS

# Ortalama yıllık dozu hesapla
average_annual_dose = cumulative_hp10_5_years / NUM_YEARS

```

```

# Yeni aşım durumu mantığı:
# 5 yıllık toplam Hp(10) 100 mSv'yi aşarsa VEYA
# 5 yıllık ortalama yıllık Hp(10) 20 mSv'yi aşarsa VEYA
# Herhangi bir yıl içinde yıllık Hp(10) ALARA limitini aşarsa 'Aşım Var' olur.
if cumulative_hp10_5_years > CUMULATIVE_DOSE_LIMIT_5_YEARS or \
average_annual_dose > 20 or \
any_yearly_excess_occurred:
    excess_status_5_years = 'Aşım Var'
    # Önerilen Sağlık İzni artık 5 yıllık ortalama yıllık çalışma süresine göre
    hesaplanacak
    average_annual_working_hours_for_leave = total_working_hours_5_years_individual /
    NUM_YEARS
    health_leave = calculate_health_leave(average_annual_working_hours_for_leave)
    extra_leave = calculate_extra_leave(dose_excess_5_years)
else:
    excess_status_5_years = 'Aşım Yok'
    health_leave = 0
    extra_leave = 0

# Her yılın verisine kümülatif doz ve aşım bilgilerini ekle
for entry in yearly_data:
    entry['5 Yıllık Hp(10) Toplamı (mSv)'] = round(cumulative_hp10_5_years, 2)
    entry['5 Yıllık Hp(10) Ortalaması (mSv)'] = round(average_annual_dose, 2) # Hp(10)
    ortalaması
    entry['Aşım Durumu'] = excess_status_5_years
    # 'Aşım Miktarı (mSv)' sütunu kaldırıldığı için buradan da kaldırıldı
    entry['Önerilen Sağlık İzni (Gün)'] = health_leave
    entry['Önerilen Ek İzni (Gün)'] = extra_leave
    # Diğer kümülatif Hp(0.07) ve Hp(3) değerleri yıllık detayda kalacak, özet
    tablodan kaldırıldı.
    # entry['5 Yıllık Kümülatif Hp(0.07) (mSv)'] = round(cumulative_hp007_5_years,
    2) # Bu sütun kaldırıldı
    # entry['5 Yıllık Kümülatif Hp(3) (mSv)'] = round(cumulative_hp3_5_years, 2) #
    Bu sütun kaldırıldı

return yearly_data

def generate_excel_report_file():
    """
    50 kişilik örnek veri ile Excel raporu oluşturur ve 'reports' klasörüne kaydeder.
    """
    all_data = []
    for i in range(NUM_INDIVIDUALS):
        all_data.extend(generate_individual_data_for_excel(i + 1))

    df_yearly = pd.DataFrame(all_data)

    # Doz 1 (2 aylık) ... Doz 6 (2 aylık) sütunlarını oluştur

```

```

dose_columns_hp10 = [f'Doz {i+1} Hp(10) (2 aylık)' for i in
range(DOSE_PERIODS_PER_YEAR)]
dose_columns_hp007 = [f'Doz {i+1} Hp(0.07) (2 aylık)' for i in
range(DOSE_PERIODS_PER_YEAR)]
dose_columns_hp3 = [f'Doz {i+1} Hp(3) (2 aylık)' for i in
range(DOSE_PERIODS_PER_YEAR)]

# Yıllık veriler için sütunları istenen sıraya göre düzenle
ordered_yearly_columns = [
    'Yıl', 'TC Kimlik No', 'Adı', 'Soyadı', 'Cinsiyet', 'Yaş', 'Hamilelik Durumu', # 'Yaş'
    sütunu eklendi
    'Kurum Adı', 'Departman', 'Çalışma Koşulu', 'Yıllık ALARA Limiti (mSv)',
    *dose_columns_hp10,
    *dose_columns_hp007,
    *dose_columns_hp3,
    'Yıllık Toplam Hp(10) (mSv)', 'Yıllık Aşım Durumu',
    # 'Yıllık Aşım Miktarı (mSv)', # Bu sütun kaldırıldı
    'Yıllık Toplam Hp(0.07)', 'Yıllık Toplam Hp(3)', 'Çalışma Süresi (Saat)',
    '5 Yıllık Hp(10) Toplamı (mSv)',
    # '5 Yıllık Kümülatif Hp(0.07) (mSv)', # Bu sütun kaldırıldı
    # '5 Yıllık Kümülatif Hp(3) (mSv)', # Bu sütun kaldırıldı
    '5 Yıllık Hp(10) Ortalaması (mSv)',
    'Aşım Durumu',
    # 'Aşım Miktarı (mSv)', # Bu sütun kaldırıldı
    'Önerilen Sağlık İzni (Gün)', 'Önerilen Ek İzni (Gün)'
]

df_yearly = df_yearly[ordered_yearly_columns]

# 5 yıllık toplamlar için özet DataFrame oluştur
df_summary = df_yearly.drop_duplicates(subset=['TC Kimlik No']).copy()

# Ortalama Yıllık Çalışma Süresi hesaplaması eklendi
df_summary['Ortalama Yıllık Çalışma Süresi (Saat)'] = round(df_summary['Çalışma
Süresi (Saat)'] / NUM_YEARS, 2)
# 5 Yıllık Toplam Çalışma Süresi (Saat) sütununu ekleyelim
df_summary['5 Yıllık Toplam Çalışma Süresi (Saat)'] = df_yearly.groupby('TC Kimlik
No')['Çalışma Süresi (Saat)'].transform('sum')

# Özet raporda sadece ilgili sütunları tut ve isimlerini güncelle
ordered_summary_columns = [
    'TC Kimlik No', 'Adı', 'Soyadı', 'Cinsiyet', 'Yaş', 'Hamilelik Durumu', # 'Yaş' sütunu
    eklendi
    'Kurum Adı', 'Departman', 'Çalışma Koşulu',
    '5 Yıllık Hp(10) Toplamı (mSv)',
    '5 Yıllık Hp(10) Ortalaması (mSv)',
    'Ortalama Yıllık Çalışma Süresi (Saat)', '5 Yıllık Toplam Çalışma Süresi (Saat)',

```

```

'Aşım Durumu',
# 'Aşım Miktarı (mSv)', # Bu sütun kaldırıldı
'Önerilen Sağlık İzni (Gün)', 'Önerilen Ek İzni (Gün)'
]

# İstenmeyen sütunları kaldır ve sırayı ayarla
df_summary = df_summary[ordered_summary_columns]

# Rapor dosyasının kaydedileceği klasörü belirle
script_dir = os.path.dirname(os.path.abspath(__file__))
reports_dir = os.path.join(script_dir, 'reports')

if not os.path.exists(reports_dir):
    os.makedirs(reports_dir)

output_filename = os.path.join(reports_dir, 'radyasyon_doz_raporu.xlsx')

# KULLANICIYA UYARI MESAJI VE BEKLEME KOMUTU BURAYA
TAŞINDI
print(f'\nExcel raporu '{output_filename}' oluşturuluyor veya güncelleniyor.")
print("Eğer dosya açıksa, lütfen kapatın.")
input("Dosyayı kapattıktan sonra devam etmek için Enter'a basın...") # Kullanıcıdan
onay bekle

# ExcelWriter kullanarak farklı sekmelere yaz
try:
    withpd.ExcelWriter(output_filename, engine='openpyxl') as writer:
        df_yearly.to_excel(writer, sheet_name='Yıllık Doz Detayı', index=False)
        df_summary.to_excel(writer, sheet_name='5 Yıllık Toplamlar', index=False)

    # Excel dosyasını openpyxl ile tekrar açarak formülleri ekle
    wb = openpyxl.load_workbook(output_filename)
    ws_yearly = wb['Yıllık Doz Detayı']
    ws_summary = wb['5 Yıllık Toplamlar']

    # Yıllık Doz Detayı sekmesi için sütun indekslerini bulalım
    header_yearly = [cell.value for cell in ws_yearly[1]]
    try:
        # Yeni 2 aylık doz sütunlarının başlangıç indekslerini bul
        doz1_hp10_col_idx = header_yearly.index('Doz 1 Hp(10) (2 aylık)') + 1
        doz1_hp007_col_idx = header_yearly.index('Doz 1 Hp(0.07) (2 aylık)') + 1
        doz1_hp3_col_idx = header_yearly.index('Doz 1 Hp(3) (2 aylık)') + 1

        yıllık_hp10_col_idx = header_yearly.index('Yıllık Toplam Hp(10) (mSv)') + 1
        yıllık_hp007_col_idx = header_yearly.index('Yıllık Toplam Hp(0.07)') + 1
        yıllık_hp3_col_idx = header_yearly.index('Yıllık Toplam Hp(3)') + 1
    except ValueError as e:

```

```
print(f'Hata: 'Yıllık Doz Detayı' sekmesinde gerekli sütun başlıklarından biri bulunamadı: {e}')
return
```

```
# Yıllık Doz Detayı sekmesindeki veri satırları için döngü (başlık satırından sonra başlar)
for row_idx in range(2, ws_yearly.max_row + 1):
    # Yıllık Toplam Hp(10) (mSv) hücresine SUM formülü yaz
    start_cell_hp10 = openpyxl.utils.get_column_letter(doz1_hp10_col_idx) + str(row_idx)
    end_cell_hp10 = openpyxl.utils.get_column_letter(doz1_hp10_col_idx + DOSE_PERIODS_PER_YEAR - 1) + str(row_idx)
    ws_yearly.cell(row=row_idx, column=yillik_hp10_col_idx).value = f=SUM({start_cell_hp10}:{end_cell_hp10})'
```

```
# Yıllık Toplam Hp(0.07) hücresine SUM formülü yaz
start_cell_hp007 = openpyxl.utils.get_column_letter(doz1_hp007_col_idx) + str(row_idx)
end_cell_hp007 = openpyxl.utils.get_column_letter(doz1_hp007_col_idx + DOSE_PERIODS_PER_YEAR - 1) + str(row_idx)
ws_yearly.cell(row=row_idx, column=yillik_hp007_col_idx).value = f=SUM({start_cell_hp007}:{end_cell_hp007})'
```

```
# Yıllık Toplam Hp(3) hücresine SUM formülü yaz
start_cell_hp3 = openpyxl.utils.get_column_letter(doz1_hp3_col_idx) + str(row_idx)
end_cell_hp3 = openpyxl.utils.get_column_letter(doz1_hp3_col_idx + DOSE_PERIODS_PER_YEAR - 1) + str(row_idx)
ws_yearly.cell(row=row_idx, column=yillik_hp3_col_idx).value = f=SUM({start_cell_hp3}:{end_cell_hp3})'
```

```
# 5 Yıllık Toplamlar sekmesi için sütun indekslerini bulalım
header_summary = [cell.value for cell in ws_summary[1]] # Headerlar hala 1. satırda try:
tc_no_col_summary_idx = header_summary.index('TC Kimlik No') + 1
cumulative_hp10_total_col_idx = header_summary.index('5 Yıllık Hp(10) Toplamı (mSv)') + 1
cumulative_hp10_average_col_idx = header_summary.index('5 Yıllık Hp(10) Ortalaması (mSv)') + 1
avg_working_hours_col_idx = header_summary.index('Ortalama Yıllık Çalışma Süresi (Saat)') + 1
total_working_hours_col_idx = header_summary.index('5 Yıllık Toplam Çalışma Süresi (Saat)') + 1
```

```
except ValueError as e:
    print(f'Hata: '5 Yıllık Toplamlar' sekmesinde gerekli sütun başlıklarından biri bulunamadı: {e}.')
return
```

```

# TC Kimlik No'ların Yıllık Doz Detayı'ndaki sütun indeksini bulalım
tc_no_yearly_col_idx = header_yearly.index('TC Kimlik No') + 1
# Yıllık Çalışma Süresi (Saat) sütununun indeksini bulalım
yearly_working_hours_col_idx = header_yearly.index('Çalışma Süresi (Saat)') + 1

# Yeni bir üst başlık satırı ekle
ws_summary.insert_rows(1)

# Üst başlık için hücreleri birleştir ve metin ekle
start_merge_col = cumulative_hp10_total_col_idx
end_merge_col = cumulative_hp10_average_col_idx
ws_summary.merge_cells(start_row=1, start_column=start_merge_col, end_row=1,
end_column=end_merge_col)
merged_cell = ws_summary.cell(row=1, column=start_merge_col)
merged_cell.value = '5 Yıllık Hp(10) Dozları'
merged_cell.alignment = Alignment(horizontal='center', vertical='center')
merged_cell.font = Font(bold=True)

# Veri satırları artık 3. satırdan başlayacak
for row_idx_summary in range(3, ws_summary.max_row + 1):
    tc_no_value = ws_summary.cell(row=row_idx_summary,
column=tc_no_col_summary_idx).value

# SUMIF formülü ile 5 yıllık kümülatif Hp(10) toplamını hesapla
ws_summary.cell(row=row_idx_summary,
column=cumulative_hp10_total_col_idx).value = \
f=SUMIF('Yıllık Doz
Detayı'!\{openpyxl.utils.get_column_letter(tc_no_yearly_col_idx)}:$'\
f\{openpyxl.utils.get_column_letter(tc_no_yearly_col_idx)},''\
f$\{openpyxl.utils.get_column_letter(tc_no_col_summary_idx)}\{row_idx_summary},''\
f\Yıllık Doz Detayı'!\{openpyxl.utils.get_column_letter(yillik_hp10_col_idx)}:$'\
f\{openpyxl.utils.get_column_letter(yillik_hp10_col_idx)})'

# 5 Yıllık Hp(10) Ortalaması için formül (Toplam / NUM_YEARS)
ws_summary.cell(row=row_idx_summary,
column=cumulative_hp10_average_col_idx).value = \
f={openpyxl.utils.get_column_letter(cumulative_hp10_total_col_idx)}\{row_idx_summ
ary}/{NUM_YEARS}'

# 5 Yıllık Toplam Çalışma Süresi (Saat) için SUMIF formülü
ws_summary.cell(row=row_idx_summary,
column=total_working_hours_col_idx).value = \
f=SUMIF('Yıllık Doz
Detayı'!\{openpyxl.utils.get_column_letter(tc_no_yearly_col_idx)}:$'\
f\{openpyxl.utils.get_column_letter(tc_no_yearly_col_idx)},''\
f$\{openpyxl.utils.get_column_letter(tc_no_col_summary_idx)}\{row_idx_summary},''\
f\Yıllık Doz
Detayı'!\{openpyxl.utils.get_column_letter(yearly_working_hours_col_idx)}:$'\

```

```

f'{openpyxl.utils.get_column_letter(yearly_working_hours_col_idx)})'

    # Ortalama Yıllık Çalışma Süresi (Saat) için formül
    # Toplam çalışma süresi / NUM_YEARS (5 yıl)
ws_summary.cell(row=row_idx_summary, column=avg_working_hours_col_idx).value
= \
f'{openpyxl.utils.get_column_letter(total_working_hours_col_idx)}{row_idx_summar
y}/{NUM_YEARS}'

    # Koşullu biçimlendirme: 5 Yıllık Kümülatif Doz 100 mSv üzerinde ise satırı
renklendir
    # '5 Yıllık Hp(10) Toplamı (mSv)' sütununun indeksini kullanacağız
col_letter = openpyxl.utils.get_column_letter(cumulative_hp10_total_col_idx)

red_fill = PatternFill(start_color="FFC7CE", end_color="FFC7CE", fill_type="solid")
# Açık kırmızı dolgu
dxf = DifferentialStyle(fill=red_fill)

    # Koşullu biçimlendirme kuralı: Eğer ilgili sütundaki değer 100'den büyükse
    # Kural, tüm veri aralığına uygulanır, ancak formül o sütunu referans alır.
    # Data range ve formula'daki satır referansları yeni durumlarına göre ayarlandı (3.
satırdan başlıyor)
data_range =
f'A3:{openpyxl.utils.get_column_letter(ws_summary.max_column)}{ws_summary.ma
x_row}'
rule = Rule(type="expression", dxf=dxf,
formula=[f'${col_letter}3>{CUMULATIVE_DOSE_LIMIT_5_YEARS}'])

ws_summary.conditional_formatting.add(data_range, rule)

wb.save(output_filename)
print(f'Excel raporu '{output_filename}' başarıyla güncellendi (tüm formüller ve
koşullu biçimlendirme eklendi).")

except Exception as e:
print(f'Excel dosyasında formül veya koşullu biçimlendirme eklenirken bir hata oluştu:
{e}')
print("Rapor dosyası değerlerle oluşturuldu, ancak formüllerin/koşullu
biçimlendirmelerin tamamı eklenemedi.")

input("Devam etmek için Enter'a basın...")

def display_excel_report_summary():
    """
    Oluşturulan radyasyon doz raporunun 5 yıllık özetini terminalde görüntüler.
    """

```

```
script_dir = os.path.dirname(os.path.abspath(__file__))
reports_dir = os.path.join(script_dir, 'reports')
output_filename = os.path.join(reports_dir, 'radyasyon_doz_raporu.xlsx')

if not os.path.exists(output_filename):
    print("\n--- Hata: Radyasyon Doz Raporu Bulunamadı ---")
    print(f'Rapor dosyası '{output_filename}' bulunamadı. Lütfen önce '1. Örnek Excel Raporu Oluştur' seçeneğini kullanarak raporu oluşturduğunuzdan emin olun.')
    input("Devam etmek için Enter'a basın...")
    return

try:
    df_summary = pd.read_excel(output_filename, sheet_name='5 Yıllık Toplamlar')

    print("\n--- 5 Yıllık Doz Toplamları ve Aşım Durumları (Excel Raporundan) ---")
    print(df_summary.to_string(index=False))
    print("\n")
    input("Devam etmek için Enter'a basın...")

except FileNotFoundError:
    print(f'\nHata: Rapor dosyası '{output_filename}' bulunamadı. Lütfen önce raporu oluşturduğunuzdan emin olun.')
    input("Devam etmek için Enter'a basın...")
except Exception as e:
    print(f'\nRapor görüntülenirken bir hata oluştu: {e}')
    input("Devam etmek için Enter'a basın...")
```

populate_data.py

```
print("DEBUG: populate_data.py başlatılıyor - Versiyon 4")
```

```
import sqlite3
import datetime
import random
import data_operations
importalara
import db_setup # DATABASE_NAME için
```

```
def populate_sample_data():
    """Uygulamayı test etmek için örnek veriler ekler."""
    print("DEBUG: Örnek veriler veritabanına ekleniyor...")
```

```
conn = sqlite3.connect(db_setup.DATABASE_NAME)
cursor = conn.cursor()
```

```
# Tabloların boş olup olmadığını kontrol et (sadece ilk çalıştırmada eklemek için)
cursor.execute("SELECT COUNT(*) FROM kurumlar")
if cursor.fetchone()[0] > 0:
    print("DEBUG: Veritabanında zaten veri bulunmaktadır, örnek veriler eklenmedi.")
conn.close()
return
conn.close()
```

```
# Kurumlar
institution1_data = {'kurum_adi': 'ABC Hastanesi', 'sehir': 'Antalya', 'iletisim':
'5551112233 / abc@hastane.com'}
institution2_data = {'kurum_adi': 'XYZ Araştırma Merkezi', 'sehir': 'Ankara', 'iletisim':
'5554445566 / xyz@arastirma.com'}
```

```
kurum1_id = data_operations.insert_institution(institution1_data)
kurum2_id = data_operations.insert_institution(institution2_data)
```

```
if kurum1_id:
    print(f"DEBUG: 'ABC Hastanesi' eklendi, ID: {kurum1_id}")
if kurum2_id:
    print(f"DEBUG: 'XYZ Araştırma Merkezi' eklendi, ID: {kurum2_id}")
```

```
# Bireyler
# ALARA limitleri calculate_alara_limit fonksiyonu ile hesaplanacak
individual1_data = {
    'tc_kimlik_no': '11111111111', 'ad': 'Ayşe', 'soyad': 'Yılmaz',
    'cinsiyet': 'Kadın', 'yas': 35, 'kurum_id': kurum1_id,
    'calisma_kosulu': 'B', 'departman': 'Radyoloji', 'hamilelik_durumu': False,
    'radyasyon_maruz_kalma_suresi': 800, 'izin_sureleri': 'Yıllık İzin: 15 gün'
}
individual1_data['alara_limiti'] = alara.calculate_alara_limit(
    individual1_data['cinsiyet'], individual1_data['hamilelik_durumu'],
    individual1_data['calisma_kosulu'], individual1_data['yas']
)
```

```
individual2_data = {
    'tc_kimlik_no': '22222222222', 'ad': 'Mehmet', 'soyad': 'Demir',
    'cinsiyet': 'Erkek', 'yas': 42, 'kurum_id': kurum1_id,
    'calisma_kosulu': 'A', 'departman': 'Nükleer Tıp', 'hamilelik_durumu': False,
    'radyasyon_maruz_kalma_suresi': 1200, 'izin_sureleri': 'Yıllık İzin: 24 gün'
}
individual2_data['alara_limiti'] = alara.calculate_alara_limit(
    individual2_data['cinsiyet'], individual2_data['hamilelik_durumu'],
    individual2_data['calisma_kosulu'], individual2_data['yas']
)
```

```
individual3_data = {
```

```

        'tc_kimlik_no': '33333333333', 'ad': 'Zeynep', 'soyad': 'Can',
        'cinsiyet': 'Kadın', 'yas': 28, 'kurum_id': kurum2_id,
        'calisma_kosulu': 'C', 'departman': 'İdari', 'hamilelik_durumu': False,
        'radyasyon_maruz_kalma_suresi': 100, 'izin_sureleri': 'Yıllık İzin: 1 gün'
    }
    individual3_data['alara_limiti'] = alara.calculate_alara_limit(
        individual3_data['cinsiyet'], individual3_data['hamilelik_durumu'],
        individual3_data['calisma_kosulu'], individual3_data['yas']
    )

    individual4_data = { # Hamilelik durumu testi için
        'tc_kimlik_no': '44444444444', 'ad': 'Elif', 'soyad': 'Kara',
        'cinsiyet': 'Kadın', 'yas': 30, 'kurum_id': kurum1_id,
        'calisma_kosulu': 'A', 'departman': 'Radyasyon Onkolojisi', 'hamilelik_durumu':
        True,
        'radyasyon_maruz_kalma_suresi': 600, 'izin_sureleri': 'Yıllık İzin: 11 gün'
    }
    individual4_data['alara_limiti'] = alara.calculate_alara_limit(
        individual4_data['cinsiyet'], individual4_data['hamilelik_durumu'],
        individual4_data['calisma_kosulu'], individual4_data['yas']
    )

    individual5_data = { # Doz aşımı için kullanılacak birey
        'tc_kimlik_no': '55555555555', 'ad': 'Fatma', 'soyad': 'Kurt',
        'cinsiyet': 'Kadın', 'yas': 38, 'kurum_id': kurum1_id,
        'calisma_kosulu': 'A', 'departman': 'Nükleer Tıp', 'hamilelik_durumu': False,
        'radyasyon_maruz_kalma_suresi': 1500, 'izin_sureleri': 'Yıllık İzin: 29 gün'
    }
    individual5_data['alara_limiti'] = alara.calculate_alara_limit(
        individual5_data['cinsiyet'], individual5_data['hamilelik_durumu'],
        individual5_data['calisma_kosulu'], individual5_data['yas']
    )

    data_operations.insert_individual(individual1_data)
    data_operations.insert_individual(individual2_data)
    data_operations.insert_individual(individual3_data)
    data_operations.insert_individual(individual4_data)
    data_operations.insert_individual(individual5_data)

    print(f"DEBUG: 'Ayşe Yılmaz' eklendi, ALARA: {individual1_data['alara_limiti']}")
    print(f"DEBUG: 'Mehmet Demir' eklendi, ALARA: {individual2_data['alara_limiti']}")
    print(f"DEBUG: 'Zeynep Can' eklendi, ALARA: {individual3_data['alara_limiti']}")
    print(f"DEBUG: 'Elif Kara' eklendi, ALARA: {individual4_data['alara_limiti']}")
    print(f"DEBUG: 'Fatma Kurt' eklendi, ALARA: {individual5_data['alara_limiti']}")

    # Doz Kayıtları

```

```
today = datetime.date.today()
current_year = today.year

# Ayşe Yılmaz için 2023 ve 2024 dozları
data_operations.insert_dose_record({
    'tc_kimlik_no': '11111111111', 'doz_tarihi': '2023-01-10', 'hp10': 0.5, 'hp007': 0.7,
    'hp3': 0.3, 'doz_tipi': 'Rutin'
})
data_operations.insert_dose_record({
    'tc_kimlik_no': '11111111111', 'doz_tarihi': '2023-04-20', 'hp10': 1.2, 'hp007': 1.5,
    'hp3': 0.8, 'doz_tipi': 'Rutin'
})
data_operations.insert_dose_record({
    'tc_kimlik_no': '11111111111', 'doz_tarihi': f'{current_year}-01-15', 'hp10': 0.8,
    'hp007': 1.0, 'hp3': 0.5, 'doz_tipi': 'Rutin'
})
data_operations.insert_dose_record({
    'tc_kimlik_no': '11111111111', 'doz_tarihi': f'{current_year}-03-05', 'hp10': 0.7,
    'hp007': 0.9, 'hp3': 0.4, 'doz_tipi': 'Rutin'
})
print(f"DEBUG: Ayşe Yılmaz için doz kayıtları eklendi.")

# Mehmet Demir için 2024 dozları
data_operations.insert_dose_record({
    'tc_kimlik_no': '22222222222', 'doz_tarihi': f'{current_year}-02-01', 'hp10': 1.5,
    'hp007': 1.8, 'hp3': 1.0, 'doz_tipi': 'Rutin'
})
data_operations.insert_dose_record({
    'tc_kimlik_no': '22222222222', 'doz_tarihi': f'{current_year}-05-10', 'hp10': 0.9,
    'hp007': 1.1, 'hp3': 0.6, 'doz_tipi': 'Rutin'
})
print(f"DEBUG: Mehmet Demir için doz kayıtları eklendi.")

# Zeynep Can için 2024 dozları
data_operations.insert_dose_record({
    'tc_kimlik_no': '33333333333', 'doz_tarihi': f'{current_year}-06-01', 'hp10': 0.2,
    'hp007': 0.3, 'hp3': 0.1, 'doz_tipi': 'Rutin'
})
print(f"DEBUG: Zeynep Can için doz kayıtları eklendi.")

# Elif Kara için (hamile) 2024 dozları - düşük limitli
data_operations.insert_dose_record({
    'tc_kimlik_no': '44444444444', 'doz_tarihi': f'{current_year}-01-01', 'hp10': 0.3,
    'hp007': 0.4, 'hp3': 0.2, 'doz_tipi': 'Rutin'
})
data_operations.insert_dose_record({
    'tc_kimlik_no': '44444444444', 'doz_tarihi': f'{current_year}-02-01', 'hp10': 0.4,
    'hp007': 0.5, 'hp3': 0.3, 'doz_tipi': 'Rutin'
})
```

```

    })
    print(f"DEBUG: Elif Kara için doz kayıtları eklendi.")

    # Fatma Kurt için yüksek dozlar (aşım testi için)
    data_operations.insert_dose_record({
        'tc_kimlik_no': '55555555555', 'doz_tarihi': f'{current_year}-01-01', 'hp10': 4.0,
        'hp007': 5.0, 'hp3': 3.0, 'doz_tipi': 'Rutin'
    })
    data_operations.insert_dose_record({
        'tc_kimlik_no': '55555555555', 'doz_tarihi': f'{current_year}-03-01', 'hp10': 6.0,
        'hp007': 7.0, 'hp3': 4.0, 'doz_tipi': 'Rutin'
    })
    data_operations.insert_dose_record({
        'tc_kimlik_no': '55555555555', 'doz_tarihi': f'{current_year}-05-01', 'hp10': 8.0,
        'hp007': 9.0, 'hp3': 5.0, 'doz_tipi': 'Rutin'
    })
    print(f"DEBUG: Fatma Kurt için yüksek doz kayıtları eklendi (aşım testi için).")

    print("DEBUG: Örnek veri ekleme tamamlandı.")

    if __name__ == "__main__":
        db_setup.setup_database() # Veritabanını kur
        populate_sample_data() # Örnek verileri ekle

```

main.py

```

from db_setup import setup_database
from individual_operations import new_individual_menu, list_all_individuals_menu,
update_individual_menu, query_individual_menu, delete_individual_menu
from institution_operations import add_institution_menu, update_institution_menu,
list_all_institutions_menu, delete_institution_menu
from dose_operations import manual_dose_entry_menu, query_dose_menu
from data_operations import import_excel_data_to_db
from report import generate_excel_report_file, display_excel_report_summary

def main_menu():
    """Ana menüyü gösterir ve kullanıcı seçimlerini işler."""
    while True:
        print("\n--- Radyasyon Doz Takip Sistemi ---")
        print("1. Kurum İşlemleri")
        print("2. Birey İşlemleri")
        print("3. Doz Girişi ve Sorgulama")

```

```
print("4. Raporlar ve Bildirimler")
print("0. Çıkış")

choice = input("Seçiminizi yapın: ")

if choice == '1':
    institution_operations_menu()
elif choice == '2':
    individual_operations_menu()
elif choice == '3':
    dose_entry_and_query_menu()
elif choice == '4':
    reports_and_notifications_menu()
elif choice == '0':
    print("Uygulamadan çıkılıyor...")
    break
else:
    print("Geçersiz seçim. Lütfen tekrar deneyin.")

def individual_operations_menu():
    """Birey İşlemleri menüsünü gösterir."""
    while True:
        print("\n--- Birey İşlemleri ---")
        print("1. Yeni Birey Ekle")
        print("2. Birey Bilgilerini Güncelle")
        print("3. Birey Sorgula (TC Kimlik No ile)")
        print("4. Tüm Bireyleri Listele")
        print("5. Birey Sil")
        print("0. Geri Dön")

        choice = input("Seçiminizi yapın: ")

        if choice == '1':
            new_individual_menu()
        elif choice == '2':
            update_individual_menu()
        elif choice == '3':
            query_individual_menu()
        elif choice == '4':
            list_all_individuals_menu()
        elif choice == '5':
            delete_individual_menu()
        elif choice == '0':
            break
        else:
            print("Geçersiz seçim. Lütfen tekrar deneyin.")

    def institution_operations_menu():
```

```
    """Kurum İşlemleri menüsünü gösterir."""
while True:
    print("\n--- Kurum İşlemleri ---")
    print("1. Yeni Kurum Ekle")
    print("2. Kurum Bilgilerini Güncelle")
    print("3. Tüm Kurumları Listele")
    print("4. Kurum Sil")
    print("0. Geri Dön")

    choice = input("Seçiminizi yapın: ")

    if choice == '1':
        add_institution_menu()
    elif choice == '2':
        update_institution_menu()
    elif choice == '3':
        list_all_institutions_menu()
    elif choice == '4':
        delete_institution_menu()
    elif choice == '0':
        break
    else:
        print("Geçersiz seçim. Lütfen tekrar deneyin.")

def dose_entry_and_query_menu():
    """Doz Girişi ve Sorgulama menüsünü gösterir."""
    while True:
        print("\n--- Doz Girişi ve Sorgulama ---")
        print("1. Manuel Doz Girişi")
        print("2. Doz Kayıtlarını Sorgula")
        print("0. Geri Dön")

        choice = input("Seçiminizi yapın: ")

        if choice == '1':
            manual_dose_entry_menu()
        elif choice == '2':
            query_dose_menu()
        elif choice == '0':
            break
        else:
            print("Geçersiz seçim. Lütfen tekrar deneyin.")

def reports_and_notifications_menu():
    """Raporlar ve Bildirimler menüsünü gösterir."""
    while True:
        print("\n--- Raporlar ve Bildirimler ---")
        print("1. Örnek Excel Raporu Oluştur")
```

```
print("2. Excel Raporundaki Verileri Veritabanına Aktar")
print("3. Excel Raporu Özetini Terminalde Görüntüle")
print("0. Geri Dön")
```

```
choice = input("Seçiminizi yapın: ")
```

```
if choice == '1':
    generate_excel_report_file()
elif choice == '2':
    import_excel_data_to_db()
elif choice == '3':
    display_excel_report_summary()
elif choice == '0':
    break
else:
    print("Geçersiz seçim. Lütfen tekrar deneyin.")
```

```
if __name__ == "__main__":
    setup_database() # Uygulama başlatıldığında veritabanını kur
    main_menu()
```

radyasyon_doz_raporu_olusturucu.py

```
import pandas as pd
```

```
import random
```

```
from datetime import datetime, timedelta
```

```
import os
```

```
# Sabitler
```

```
NUM_INDIVIDUALS = 50
```

```
NUM_YEARS = 5
```

```
DOSE_PERIODS_PER_YEAR = 6 # 2 aylık periyotlar
```

```
CUMULATIVE_DOSE_LIMIT_5_YEARS = 100 # mSv
```

```
# Sağlık izni kuralı (varsayımsal)
```

```
HEALTH_LEAVE_FOR_EXCESS = 30 # Doz aşımı durumunda verilen sağlık izni (gün)
```

```
# ALARA Limitleri (Tez belgesinden alınmıştır)
```

```
ALARA_LIMITS = {
```

```
    'A': 20, #mSv/yıl
```

```
    'B': 5, # mSv/yıl
```

```
    'C': 0.5 #mSv/yıl
```

```
}
```

```
ALARA_LIMIT_PREGNANT = 1 # mSv/yıl (Hamilelik dönemi için)
```

```

# Örnek isimler ve departmanlar
FIRST_NAMES = ["Ayşe", "Fatma", "Mehmet", "Ali", "Zeynep", "Can", "Elif",
"Deniz", "Burak", "Mert"]
LAST_NAMES = ["Yılmaz", "Demir", "Kara", "Aslan", "Öztürk", "Kaya", "Yıldırım",
"Güneş", "Aksoy", "Tunç"]
DEPARTMENTS = ["Radyoloji", "Nükleer Tıp", "Radyoterapi", "PET/CT",
"Laboratuvar", "Onkoloji"]
WORK_CONDITIONS = ['A', 'B', 'C']

# Örnek kurum adları (Akdeniz Üniversitesi Hastanesi eklendi)
INSTITUTIONS = [
    "Akdeniz Üniversitesi Hastanesi",
    "Ankara Şehir Hastanesi",
    "İstanbul Üniversitesi Hastanesi",
    "Ege Üniversitesi Hastanesi",
    "Medipol Mega Üniversite Hastanesi",
    "Dokuz Eylül Üniversitesi Hastanesi",
    "Hacettepe Üniversitesi Hastanesi"
]

defgenerate_tc_kimlik_no():
    """Rastgele bir TC Kimlik Numarası oluşturur."""
    return ".join([str(random.randint(0, 9)) for _ in range(10)]) + str(random.randint(0, 9))

defcalculate_yearly_alara(work_condition, is_pregnant, gender):
    """Yıllık ALARA limitini çalışma koşulu ve hamilelik durumuna göre hesaplar."""
    ifgender == 'Kadın' andis_pregnant:
        return ALARA_LIMIT_PREGNANT
    returnALARA_LIMITS.get(work_condition, 20) # Varsayılan olarak A koşulu limiti

defcalculate_extra_leave(dose_excess_amount):
    """Aşım miktarına göre ek izin süresini hesaplar.
    Bu fonksiyon, aşım miktarına göre kademeli olarak farklı ek izin günleri döndürür.
    """
    ifdose_excess_amount<= 0:
        return 0
    elifdose_excess_amount< 10:
        return 0 # Küçük aşım için ek izin yok
    elif 10 <= dose_excess_amount< 25:
        return 1
    elif 25 <= dose_excess_amount< 50:
        return 2
    elif 50 <= dose_excess_amount< 75:
        return 3
    else: # dose_excess_amount>= 75
        # Daha büyük aşım için, 4 gün taban izin + her ek 25 mSv için 1 gün daha
        return 4 + int((dose_excess_amount - 75) / 25)

```

```

defgenerate_individual_data(individual_id):
    """Tek bir birey için 5 yıllık doz verilerini ve ilgili bilgileri oluşturur."""
    tc_no = generate_tc_kimlik_no()
    first_name = random.choice(FIRST_NAMES)
    last_name = random.choice(LAST_NAMES)
    gender = random.choice(["Erkek", "Kadın"])
    department = random.choice(DEPARTMENTS)
    work_condition = random.choice(WORK_CONDITIONS)

    # Hamilelik durumunu rastgele belirle (sadece kadınlar için)
    is_pregnant = False
    ifgender == 'Kadın' andrandom.random() < 0.15: # %15 ihtimalle hamile
        is_pregnant = True

    # Doz senaryosu belirleme (Doz Aşımı, Sınırdaki, Limit Aşılması)
    scenario_type = random.choices(
        ['excess', 'borderline', 'normal'],
        weights=[0.4, 0.3, 0.3], # Oranlar: %40 aşım, %30 sınırdaki, %30 normal (daha fazla aşım
        ve sınırdaki durumu için ayarlandı)
        k=1
    )[0]

    yearly_data = []
    cumulative_dose_5_years = 0

    foryear_offset in range(NUM_YEARS):
        current_year = 2019 + year_offset

        # Yıllık ALARA limitini hesapla
        yearly_alara_limit = calculate_yearly_alara(work_condition, is_pregnant, gender)

        # Yıllık toplam dozu senaryoya göre belirle
        ifscenario_type == 'excess':
            # 5 yıllık toplamın 100 mSv'yi geçmesini sağlayacak yıllık ortalama (örn: 21-40
            mSv/yıl)
            avg_yearly_dose = random.uniform(21, 40) # Daha geniş aralık, daha çeşitli aşım
            miktarları için
        elifscenario_type == 'borderline':
            # 5 yıllık toplamın 100 mSv civarında olmasını sağlayacak yıllık ortalama (örn:
            18-22 mSv/yıl)
            avg_yearly_dose = random.uniform(18, 22) # Daha geniş aralık, sınırdaki daha fazla
            çeşitlilik için
        else: # normal
            # 5 yıllık toplamın 100 mSv altında kalmasını sağlayacak yıllık ortalama (örn: 1-
            19 mSv/yıl)
            avg_yearly_dose = random.uniform(1, 19) # Daha geniş aralık, normalde daha fazla
            çeşitlilik için

```

```

# Hamilelik durumunda yıllık dozu düşür
ifis_pregnantandgender == 'Kadın':
    # Hamilelik limitine yakın veya hafif üstü, ancak çok aşmayacak şekilde
    avg_yearly_dose = min(avg_yearly_dose, yearly_alara_limit + random.uniform(0, 1.0))
    # Hamilelik durumunda yıllık dozun 1 mSv'yi çok aşmamasını sağla
    avg_yearly_dose = min(avg_yearly_dose, 5.0) # Maksimum 5 mSv olarak sınırla,
    gerçekçi olması için

# 2 aylık periyotlar için dozları dağıt
bi_monthly_doses_hp10 = [max(0, random.gauss(avg_yearly_dose /
DOSE_PERIODS_PER_YEAR, (avg_yearly_dose / DOSE_PERIODS_PER_YEAR) *
0.2)) for _ in range(DOSE_PERIODS_PER_YEAR)]

# Yıllık toplam dozları hesapla
total_hp10_yearly = sum(bi_monthly_doses_hp10)
total_hp007_yearly = sum([d * random.uniform(0.2, 0.5) for d in
bi_monthly_doses_hp10])
total_hp3_yearly = sum([d * random.uniform(0.1, 0.4) for d in
bi_monthly_doses_hp10])

# Yıllık aşım durumu kontrolü
yearly_excess_status = 'Aşım Var' if total_hp10_yearly > yearly_alara_limit else 'Aşım
Yok'

# Yıllık aşım miktarı hesaplaması
yearly_excess_amount = max(0, total_hp10_yearly - yearly_alara_limit)

cumulative_dose_5_years += total_hp10_yearly

yearly_data.append({
    'Yıl': current_year,
    'TC Kimlik No': tc_no,
    'Adı': first_name,
    'Soyadı': last_name,
    'Cinsiyet': gender,
    'Hamilelik Durumu': 'Evet' ifis_pregnant else 'Hayır',
    'Kurum Adı': random.choice(INSTITUTIONS), # Kurum adları listesinden
    rastgele seçildi
    'Departman': department,
    'Çalışma Koşulu': work_condition,
    'Yıllık ALARA Limiti (mSv)': yearly_alara_limit,
    **{f'Doz {i+1} (2 aylık)': round(d, 2) for i, d in
enumerate(bi_monthly_doses_hp10)},
    'Yıllık Toplam Hp(10) (mSv)': round(total_hp10_yearly, 2),
    'Yıllık Aşım Durumu': yearly_excess_status,
    'Yıllık Aşım Miktarı (mSv)': round(yearly_excess_amount, 2), # Yeni eklenen
sütun

```

```

'Yıllık Toplam Hp(0.07)': round(total_hp007_yearly, 2),
'Yıllık Toplam Hp(3)': round(total_hp3_yearly, 2),
'5 Yıllık Kümülatif Doz (mSv)': 0, # Bu değer en son doldurulacak
'Aşım Durumu': "", # Bu değer en son doldurulacak (5 yıllık)
'Aşım Miktarı (mSv)': 0, # Bu değer en son doldurulacak (5 yıllık)
'Önerilen Sağlık İzni (Gün)': 0, # Bu değer en son doldurulacak
'Önerilen Ek İzni (Gün)': 0 # Bu değer en son doldurulacak
})

# 5 yıllık kümülatif doz, aşım durumu ve izinleri hesapla
dose_excess_5_years = cumulative_dose_5_years -
CUMULATIVE_DOSE_LIMIT_5_YEARS

# Ortalama yıllık dozu hesapla
average_annual_dose = cumulative_dose_5_years / NUM_YEARS

# Yeni aşım durumu mantığı: 5 yıllık toplam 100 mSv'yi aşarsa VEYA yıllık
ortalama 20 mSv'yi aşarsa 'Aşım Var'
if cumulative_dose_5_years > CUMULATIVE_DOSE_LIMIT_5_YEARS
or average_annual_dose > 20:
    excess_status_5_years = 'Aşım Var'
    health_leave = HEALTH_LEAVE_FOR_EXCESS
    extra_leave = calculate_extra_leave(dose_excess_5_years)
else:
    excess_status_5_years = 'Aşım Yok'
    health_leave = 0
    extra_leave = 0

# Her yılın verisine kümülatif doz ve aşım bilgilerini ekle
for entry in yearly_data:
    entry['5 Yıllık Kümülatif Doz (mSv)'] = round(cumulative_dose_5_years, 2)
    entry['Aşım Durumu'] = excess_status_5_years # 5 yıllık aşım durumu
    entry['Aşım Miktarı (mSv)'] = round(max(0, dose_excess_5_years), 2)
    entry['Önerilen Sağlık İzni (Gün)'] = health_leave
    entry['Önerilen Ek İzni (Gün)'] = extra_leave

return yearly_data

def create_excel_report():
    """50 kişilik örnek veri ile Excel raporu oluştur."""
    all_data = []
    for i in range(NUM_INDIVIDUALS):
        all_data.extend(generate_individual_data(i + 1))

    df_yearly = pd.DataFrame(all_data)

    # Doz 1 (2 aylık) ... Doz 6 (2 aylık) sütunlarını oluştur
    dose_columns = [f'Doz {i+1} (2 aylık)' for i in range(DOSE_PERIODS_PER_YEAR)]

```

```

# Yıllık veriler için sütunları istenen sıraya göre düzenle
ordered_yearly_columns = [
    'Yıl', 'TC Kimlik No', 'Adı', 'Soyadı', 'Cinsiyet', 'Hamilelik Durumu',
    'Kurum Adı', 'Departman', 'Çalışma Koşulu', 'Yıllık ALARA Limiti (mSv)',
    *dose_columns,
    'Yıllık Toplam Hp(10) (mSv)', 'Yıllık Aşım Durumu', 'Yıllık Aşım Miktarı (mSv)',
# Yeni sütun eklendi
    'Yıllık Toplam Hp(0.07)', 'Yıllık Toplam Hp(3)',
    '5 Yıllık Kümülatif Doz (mSv)', 'Aşım Durumu', 'Aşım Miktarı (mSv)',
    'Önerilen Sağlık İzni (Gün)', 'Önerilen Ek İzni (Gün)'
]

df_yearly = df_yearly[ordered_yearly_columns]

# 5 yıllık toplamlar için özet DataFrame oluştur
# Her birey için tek bir satır alıyoruz, çünkü 5 yıllık kümülatif bilgiler her yıl için
aynı tekrar ediyor
df_summary = df_yearly.drop_duplicates(subset=['TC Kimlik No']).copy()

# Ortalama Yıllık Doz hesaplaması eklendi
df_summary['Ortalama Yıllık Doz (mSv)'] = round(df_summary['5 Yıllık Kümülatif
Doz (mSv)'] / NUM_YEARS, 2)

# Özet raporda sadece ilgili sütunları tut
ordered_summary_columns = [
    'TC Kimlik No', 'Adı', 'Soyadı', 'Cinsiyet', 'Hamilelik Durumu',
    'Kurum Adı', 'Departman', 'Çalışma Koşulu',
    '5 Yıllık Kümülatif Doz (mSv)', 'Ortalama Yıllık Doz (mSv)',
    'Aşım Durumu', 'Aşım Miktarı (mSv)',
    'Önerilen Sağlık İzni (Gün)', 'Önerilen Ek İzni (Gün)'
]
df_summary = df_summary[ordered_summary_columns]

# Rapor dosyasının kaydedileceği klasörü belirle
# Bu betiğin bulunduğu dizini al
script_dir = os.path.dirname(os.path.abspath(__file__))
# 'reports' klasörünün yolunu oluştur
reports_dir = os.path.join(script_dir, 'reports')

# 'reports' klasörü yoksa oluştur
if not os.path.exists(reports_dir):
    os.makedirs(reports_dir)

# Excel dosyasının tam yolunu oluştur
output_filename = os.path.join(reports_dir, 'radyasyon_doz_raporu.xlsx')

```

```
# ExcelWriter kullanarak farklı sekmelere yaz
withpd.ExcelWriter(output_filename, engine='openpyxl') as writer:
    df_yearly.to_excel(writer, sheet_name='Yıllık Doz Detayı', index=False)
    df_summary.to_excel(writer, sheet_name='5 Yıllık Toplamlar', index=False)

print(f'Excel raporu '{output_filename}' başarıyla oluşturuldu.")

if __name__ == "__main__":
    create_excel_report()
```



ÖZGEÇMİŞ

Gökhan AYÇİÇEK

ÖĞRENİM BİLGİLERİ

Yüksek Lisans	Akdeniz Üniversitesi
2022-2025	Fen Bilimleri Enstitüsü, Fizik Anabilim Dalı, Antalya
Lisans	Süleyman Demirel Üniversitesi
2010-2015	Fen - Edebiyat Fakültesi, Fizik Bölümü, Isparta