**T.C.**

**FIRAT ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ**

**YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ**

# Proje Dokümantasyonu

**Tez Duzenleyicisi**

**Proje Ekibi**

**Yusuf İNEL**

**Muhsin Polat UÇAR**

**Mehmet BEKTAŞ**

**23.01.2021**

|  |
| --- |
|  |
| **1.GİRİŞ** |
| * 1. Projenin Amacı   2. Projenin Kapsamı   3. Tanımlamalar ve Kısaltmalar |
| 1. **PROJE PLANI** |
| * 1. Giriş   2. Projenin Plan Kapsamı   3. Proje Zaman-İş Planı   4. Proje Ekip Yapısı   5. Önerilen Sistemin Teknik Tanımları   6. Kullanılan Özel Geliştirme Araçları ve Ortamları   7. Proje Standartları, Yöntem ve Metodolojiler   8. Kalite Sağlama Planı   9. Konfigürasyon Yönetim Planı   10. Kaynak Yönetim Planı   11. Eğitim Planı   12. Test Planı   13. Bakım Planı |
| 1. **SİSTEM ÇÖZÜMLEME** |
| * 1. **Mevcut Sistem İncelemesi**      1. Örgüt Yapısı      2. İşlevsel Model      3. Veri Modeli      4. Var olan Yazılım/Donanım Kaynakları      5. Var olan Sistemin Değerlendirilmesi   2. **Gereksenen Sistemin Mantıksal Modeli**      1. Giriş      2. İşlevsel Model      3. Genel Bakış      4. Bilgi Sistemleri/Nesneler      5. Veri Sözlüğü      6. İşlevlerin Sıradüzeni      7. Başarım Gerekleri   3. **Ara yüz (Modül) Gerekleri**      1. Yazılım Ara yüzü   4. **Belgeleme Gerekleri**      1. Geliştirme Sürecinin Belgelenmesi      2. Eğitim Belgeleri |
| 1. **SİSTEM TASARIMI** |
| * 1. **Genel Tasarım Bilgileri**       1. Genel Sistem Tanımı      2. Varsayımlar ve Kısıtlamalar      3. Sistem Mimarisi      4. Dış Arabirimler         1. Kullanıcı Arabirimleri         2. Veri Arabirimleri         3. Diğer Sistemlerle Arabirimler      5. Veri Modeli      6. Testler      7. Performans   2. **Veri Tasarımı**      1. Tablo tanımları      2. Tablo- İlişki Şemaları      3. Veri Tanımları      4. Değer Kümesi Tanımları   3. **Süreç Tasarımı**      1. Genel Tasarım      2. Modüller         1. XXX Modülü            1. İşlev            2. Kullanıcı Arabirimi            3. Modül Tanımı            4. Modül iç Tasarımı         2. YYY Modülü      3. Kullanıcı Profilleri      4. Entegrasyon ve Test Gereksinimleri   4. **Ortak Alt Sistemlerin Tasarımı**      1. Ortak Alt Sistemler      2. Modüller arası Ortak Veriler      3. Ortak Veriler İçin Veri Giriş ve Raporlama Modülleri      4. Güvenlik Alt sistemi      5. Veri Dağıtım Alt sistemi      6. Yedekleme ve Arşivleme İşlemleri |
| 1. **SİSTEM GERÇEKLEŞTİRİMİ** |
| * 1. Giriş   2. Yazılım Geliştirme Ortamları   3. Kodlama Stili      1. Açıklama Satırları      2. Kod Biçimlemesi      3. Anlamlı İsimlendirme      4. Yapısal Programlama Yapıları   4. Program Karmaşıklığı      1. Programın Çizge Biçimine Dönüştürülmesi      2. McCabe Karmaşıklık Ölçütü Hesaplama   5. Olağan Dışı Durum Çözümleme   6. Kod Gözden Geçirme |

# 1 .GİRİŞ

### 1.1 Projenin Amacı

Bu Proje’nin amacı insanların yüksek lisans eğitiminde tez hazırlarken yazım ve mantık kurallarına uygun bir şekilde hazırlamasıdır. Bu şekilde yapılan tez dokümantasyonlarında daha anlaşılır bir şekilde yapılmasını hedefler.

### 1.2 Projenin Kapsamı

Projedeki Tez Düzenleyicisinin kapsadığı bazı uygulama alanları ;

**•** Ön Lisans,

•Lisans,

•Yüksek Lisans,

•Doktora.

### 1.3 Tanımlamalar ve kısaltmalar

Tez Düzenleyicisi: TD

# 2. PROJE PLANI

### 2.1 Giriş

Bu projede derleyici ve editör programlarında c# programlama dilini kullanılarak bir tez düzenleyicisi elde etmektir.

### 2.2 Projenin Plan Kapsamı

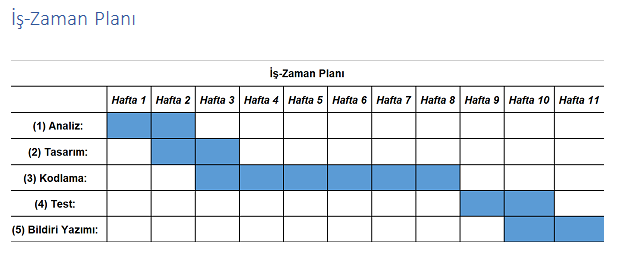
Projenin plan kapsamında genel olarak mevcut sistem, sistem gerekliliği ve bu sistem sayesinde yapılacak olan tezlerin daha anlaşılır ve okunaklı olması planlanmıştır.

#### TEZ DÜZENLEYİCİ HAYATIMIZDA NEDEN GEREKLİ?

•ilk olarak Tez’in ne anlama geldiğin bilirsek hayatımızda neden gerekli olduğunu anlayabiliriz. Tez lisans düzeyi ve üst seviyelerdeki kişilerin araştırma yapma yeteneğinin kazanması ve yaptığı araştırma ile katkı sağlamaktır. Bu yüzden tez hazırlayan kişinin daha okunaklı ve anlaşılır bir şekilde karşı tarafa istediklerini anlatmak isteriz bu yüzden TD sayesinde bu koşulları yerine getirmiş oluruz.

### 2.3 Projenin Zaman-İş Planı

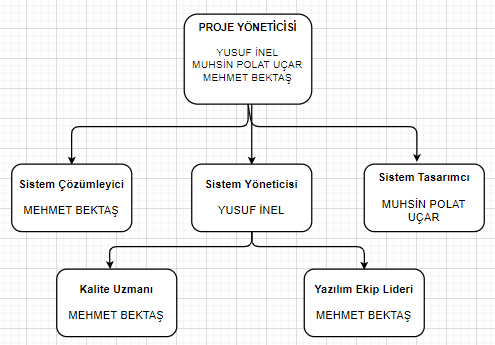
Proje’nin zaman ve iş planı şekil 2.3.1 ‘de gösterilmiştir.



(şekil 2.3.1)

### 2.4 Projenin Ekip Yapısı

Bu projede 3 kişi tarafından yürütülmektedir. Kişiler belgeleme ve kodlama olmak üzere 2 yapıya bölünmüştür. Zaman zaman projedeki kişilerin birbirine yardımı olmuştur.



(Şekil 2.4.1)

### 2.5 Önerilen Sistemin Teknik Tanımları

### 2.6 Kullanılan Özel Geliştirme Araçları ve Ortamları

Bu projede;

•Tasarım Araçları

Visio 2016,Draw io,Word,Excel ,Visual Studio 2019,Visual Studio code

•Programlama Araçları

C#

•Sınama Araçları

Microsoft Windows 7,Microsoft Windows 10

•Destek Araçları

Mozilla Firefox,Google Chrome20 ,Safari,Yandex,

### 2.7 Proje Standartları, Yöntem ve Metodolojiler

TD’de spiral modeli kullanılmıştır. İlk bölümde planlama aşaması için kullanıcıdan. docx uzantılı dosya istenir. İkinci çeyrekte, risk çözümlemesi olarak kullanıcının girdiği . docx uzantılı hatalı yerler belirlenir. Üçüncü çeyrekte risk çözümlemesi olarak ikinci çeyrekte raporlanan hataların ortadan kaldırmak için kullanıcı uyarılır. Dördüncü çeyrekte hataların belirlenip ve kullanıcıya bildirildikten sonra sistem modeli tamamlanmış olur.Formun Üstü

### 2.8 Kalite Sağlama Planı

#### 2.8.1 Tamlık:

Projede herhangi bir açık olmamalı ve programda bulunan tüm butonlar textler vs. çalışır ve tamdır.

#### 2.8.2 Belgeleme

Bu belgeden de anlaşılacağı üzere tam anlamıyla sistemin özeti olacak bu doküman oluşturulmuştur.

#### 2.8.3 Etkinlik

Kullanıcı sistemin basit ara yüzünden dolayı her alana hâkim olduğu için sistemi etkin bir biçimde kullanacak.

### 2.9 Konfigürasyon Yönetim Planı

### 2.10 Kaynak Yönetim Planı

Mevcut bir kaynak yönetimi bulunmamaktadır

### 2.11 Eğitim Planı

Projede kullanılacak dillerin ve projenin belgelemesi yapılabilmesi için bazı eğitimler alınması gereklidir.

Proje kapsamı alınacak olan eğitimler;

Visual studio 2019 eğitimi

Visio 2016 eğitimi

Word ve Excel eğitimi

Eğitimlerin zaman çizelgesi;

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Eğitimler | Başlama tarihi | Bitiş tarihi | Yusuf İnel | Muhsin Polat  Uçar | Mehmet  Bektaş |
| Visual studio 2019 eğitimi | Ekim 31 | Aralık 25 | • | • | • |
| Visio 2016  Eğitimi | Ekim 31 | Aralık 25 | • | • |  |
| Word eğitimi | Ekim 31 | Aralık 25 | • | • | • |
| Excel eğitimi | Ekim 31 | Aralık 25 |  |  | • |

Bu eğitimlerin verilmemesi durumunda kullanıcılar sistemi tam olarak kullanamayacaktır.

### 2.12 Test Planı

Tez çalışması yapacak son sınıf doktora, lisans, ön lisans öğrencilerinin tez çalışmalarını yazım doğruluğunu kontrol etmek amacıyla denenecektir.

### 2.13 Bakım Plan

Proje çalışabilir vaziyette iken hata alınmaksızın üzerine güncelleme geldiği hemen ardından bakım yapılacaktır.

# 3.SİSTEM ÇÖZÜMLEME

## 3.1 MEVCUT SİSTEMLERİN İNCELENMESİ

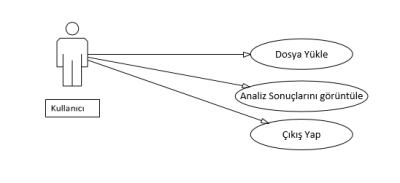
Mevcut sistem bulunamadığı için projenin geçerliliğini gözetleme yapılamayacaktır.

#### 3.1.1 Örgüt Yapısı

Örgüt yapısı olarak Fırat üniversitesi bünyesindeki bilgi işlem personellerinin ortak çalışmasıyla oluşan bir örgüt yapısı vardır.

#### 3.1.2 İşlevsel Model

Sistemin Use Case Diyagramı

[](https://cdn.discordapp.com/attachments/757335011685826670/795359769464012830/unknown.png)

#### 3.1.3 Veri Modeli

Bu projede VTYS(veri tabanı yönetim sistemleri) kullanılmadığı için bir ilişki söz konusu değildir.

#### 3.1.4 Var olan Yazılım/Donanım Kaynaklar

-Microsoft Word

-Microsoft Visual Studio

-Visual Studio code

#### 3.1.5 Var olan Sistemin Değerlendirilmesi

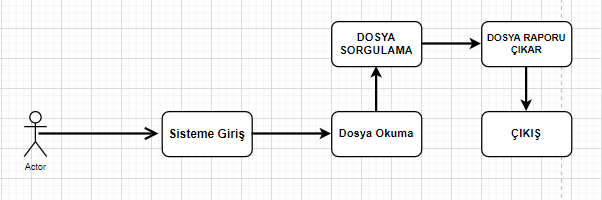
Var olan bir sistem bulunamadığı için sistem değerlendirilmesi yapılamamıştır.

## 3.2 Gereksenen Sistemin Mantıksal Modeli

### 3.2.1 Giriş

Mevcut sistemler olmadığından dolayı sonuca giden yolda epeyce bir eksikler ve resmi olmayan durumlar söz konusu artık bu sistemi Türkiye hukuk standartlarına uydurmak bize kalıyor.

### 3.2.2 İŞLEVSEL MODEL



Durum ismi Tez Okuma

Aktör başlatıyor Aktör

Giriş Durumu Üniversiteye bağlı Aktör giriş yapmış

Olayların akışı

1.Sisteme giriş 2.Dosya okuma 3.Dosya sorgula

Durum çıkışı Tez raporu çıkar

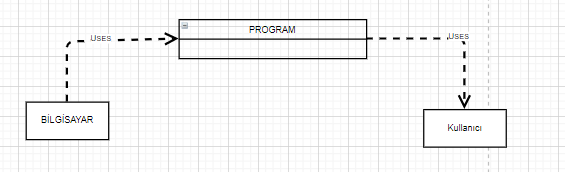
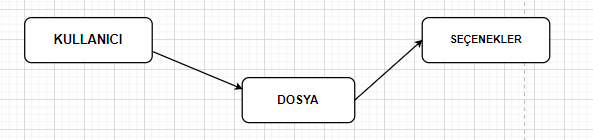
### 3.2.3 Genel Bakış

Projenin işlevselliği göz önünde bulundurularak başarım deneyi yapılarak ortaya çıkan sonuç ve feed-back’lere göre projenin ilerleyen zamanlarda neye doğru evirileceği düşünülecektir.

### 3.2.4 Bilgi Sistemleri/Nesneler

**Kullanıcı**: Sistemin tek nesnesidir. Program tarafından tüm yetkilere sahiptir.

[https://media.discordapp.net/attachments/757335011685826670/795377416137998346/unknown.png?width=400&height=44](https://cdn.discordapp.com/attachments/757335011685826670/795377416137998346/unknown.png)



### 3.2.5 Veri Sözlüğü

Dosya\_adı=\*String tipinde bir değerden oluşur\*

Dosya\_yolu=\*String tipinde bir değerden oluşur\*

Dosya\_biçimi=\*String tipinde bir değerden oluşur\*

Durum=\*boolean tipinde bir değerden oluşur. Tezin içerisindeki istenilen durumu gösterir.\*

### 3.2.6 İşlevlerin Sıra Düzeni

Kullanıcı tarafından programa giriş yapılır. Kullanıcı intikal analizini yapmak istediği dosyayı programa yükler. Program kullanıcının istediği dosya türünü değerlendirir. Kullanıcı yüklediği dosya formatı doğruysa program çalışmaya başlar ve değerlendirme tamamlandığı zaman program son çıktıları kullanıcıya iletir ve program sollanır.

**3.2.7 Başarım Gerekleri**

İlk ürün ortaya koyulduğunda kullanıcılara bir deneyim yaşatıldı ve başarım için aşağıdaki maddeler temel gereklilikler olarak tespit edildi.

•Tarafsızlık

•Kullanım Kolaylığı

•Zaman tasarrufunun sağlanması

•Hile hata ve yanlışlıkların en aza indirilmesi ve tespit edilmesi

## 3.3 Ara yüz Gerekleri

### 3.3.1 Yazılım Ara yüzü

Projenin çalışması esnasında böyle bir açık verilmemesine özen gösterildi. Gerekli olan her türlü değişiklik kaynak kodları üzerinden yapılıp tekrar derlenecek.

## 3.4 Belgeleme Gerekleri

### 3.4.1 Geliştirme Sürecinin Belgelenmesi

Geliştirme sürecinde genel olarak belgelendirilmesi hem ileriye dönük hem de şimdiki geliştirme sürecinde projenin tamamlanma yüzdesini nerede kalınıp nerelerde eksikler olduğunu genel hatlarıyla göstermesi amacıyla yapıldı.

### 3.4.2 Eğitim Belgeleri

Mevcut bir eğitim belgesi bulunmamaktadır.

# 4. SİSTEM TASARIMI

### 4.1 Genel Tasarım Bilgileri

**C:\Users\PC\AppData\Local\Microsoft\Windows\INetCache\Content.Word\SİSTEM TASAMI.PNG**

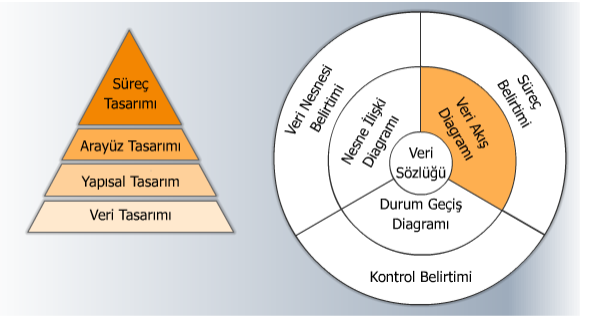
• **Gereksinimler**

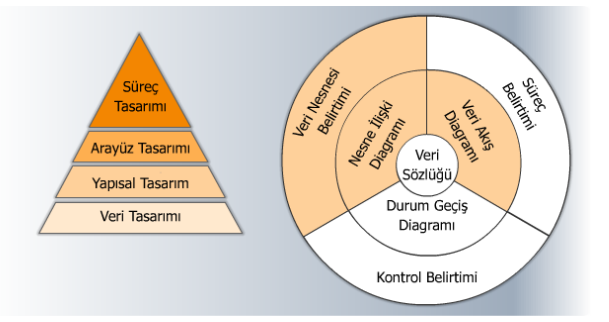
Lisans, ön lisans, doktora yapan üniversitedeki son sınıf öğrencilere programın kullanışı hakkında bir sunum ve anket yapılıp ona göre bir tasarım ortaya konulacaktır.

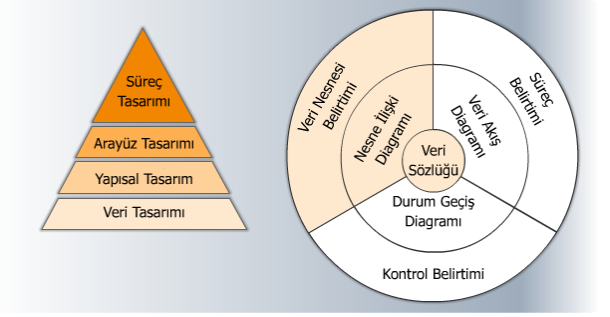
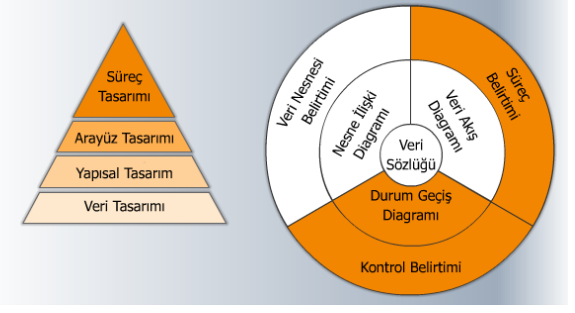
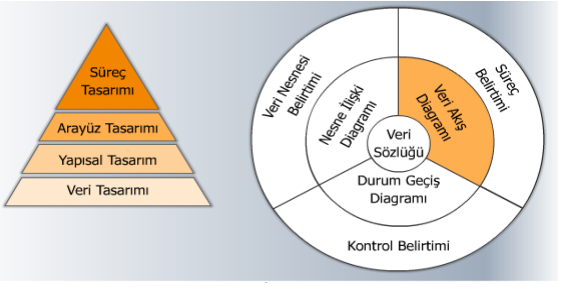
**• İşlevsel Belirtimler**

Bu program kullanıcılar tarafından yüksek tez örneklerinde yapılan hataların dikkatlice incelenerek bir sonuca varılarak sistemin güncellenmesi amaçlanmaktadır. İnternet üzerinden yüksek lisans tezleri veya lisans tezlerinde yapılan hata ve hileler tespit edilerek sistemin daha sabit ve doğru ölçüde sonuç üretilmesini amaçlamaktayız.

**• Tasarım**

Tasarım aşamasında neler olacağını grafiksel olarak aktarmak isterim.





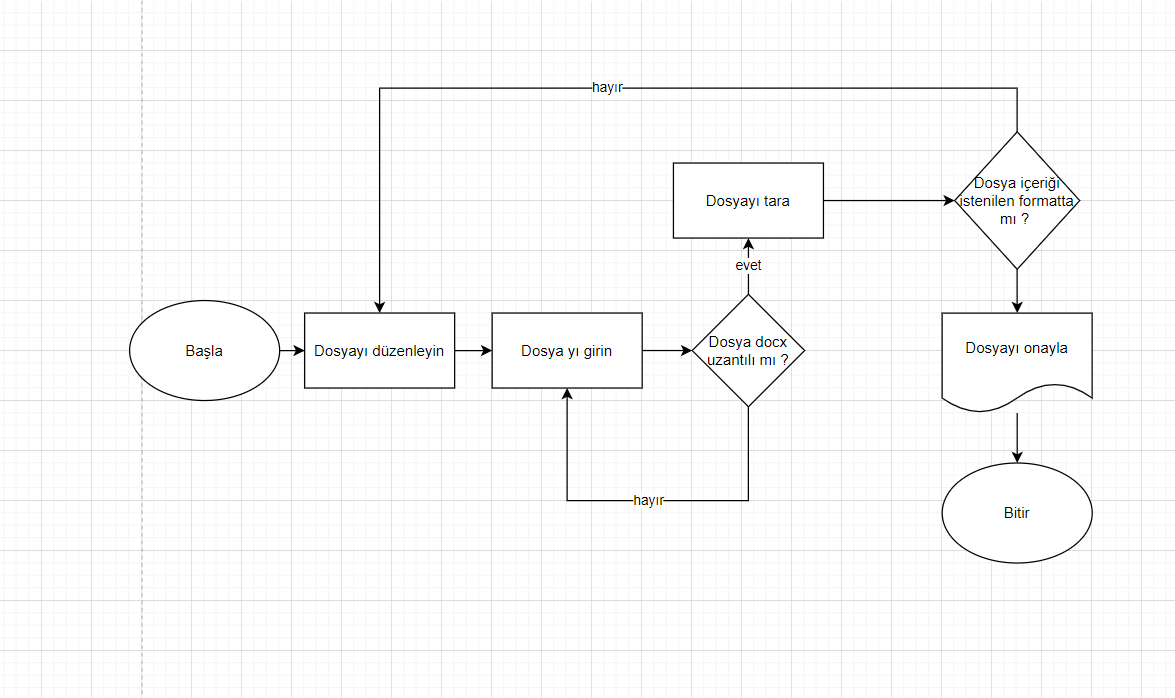
### 4.1.2 Varsayımlar ve Kısıtlamalar

Bu sistemde varsayılan değerler bulunmamakta bunun yanı sıra başlıca kısıtlamalar bulunmaktadır. Kısıtlamalar şunlardır:

•Üniversite Son sınıf bir öğrenci olması

•Sisteme yüklenecek olan dosyanın. docx uzantılı olmasıdır**.**

### 4.1.3 Sistem Mimarisi



### 4.1.4 Dış Arabirimler

Bu sistemde sadece tek bir aktör olarak kullanıcı tanımlanmıştır. Bundan dolayı ilk arabirimde kullanıcı direk olarak sisteme giriş yapılabilecektir ve istediği işlemleri gerçekleştirmek için sisteme yüklemek istediği dosyayı yükleyip sistemi kullanmış olacaktır.

### 4.1.5 Testler

Bu projenin testi 2 aşamada gerçekleşecektir.

**Alfa** **Aşaması**: Sistemin geliştirildiği yerde kullanıcıların gelerek katkıda bulunması sistemi test etmesi ile yapılacak.

**Beta Aşaması**: Kullanıcı, geliştirilen sistemi kendi yerleşkesinde, bir gözetmen eşliğinde yapılacak.

### 4.1.6 Performans

Sistemin performansını etkileyen faktörlerin test verileri değerlendirilecek Sistemin Tasarıma Uygunluk Performansı; Tasarımı yapılan sistemin stabilizesi ve işleyiş performansı değerlendirilecek. Veri Yapısının Sistemle Performansı; Veri yapısının sistemle stabilizesi ve çalışma zamanındaki uyumluluk düzeyindeki performansı değerlendirilecek.

## 4.2 Veri Tasarımı

### 4.2.1 Tablo tanımları

Sistem tek bir tablodan oluşmaktadır. Sistemi kullanacak her varlığın insan olduğunu varsayarsak sadece kullanıcı tablosu bulunmaktadır.

##### Tablo 4.1 Kullanıcı tablosu

|  |  |
| --- | --- |
| Kullanıcı Tablosu |  |
| Veri Adı | Veri Tipi |
| Dosya\_Adı | Varchar(60) |
| Dosya\_yolu | Varchar(60) |
| Dosya\_biçimi | Varchar(60) |
| Durum | Binary |

### 4.2.3 Veri Tanımları

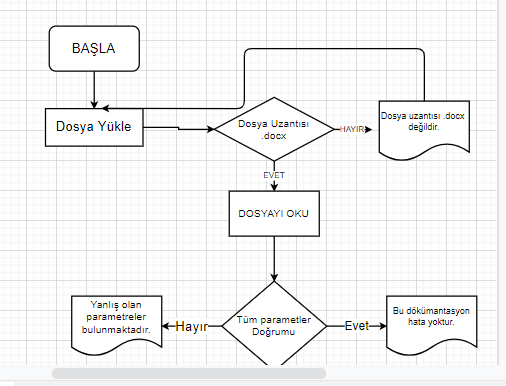
Veri tipi olarak varchar kullanılmasının amacı Veri adlarının karakter içermesidir. Binary veri tipi kullanılmasının amacı Durum adlı verinin kontrolü içindir. Binary yalnızca 2 durumdan oluşur bunlar 0 ve 1 dir.0 durumu dosyanın okunabilir özelliğinin kapalı olması ve 1 durumu ise dosyanın okunabilirliğinin açık olması anlamına gelir.

## 4.3 Süreç Tasarımı

### 4.3.1 Genel Tasarım

Genel olarak bu projede sadece kullanıcı Ara yüz modeli oluşturuldu. Kullanıcı Ara yüzü oluşturulmadan önce kullanıcı akış diyagramları oluşturulmuştur.

##### Tablo 4.2 Kullanıcı Akış Diyagramı



### 4.3.2 Modüller

#### 4.3.2.1 Giriş Modülü

#### 4.3.2.1.1 İşlev

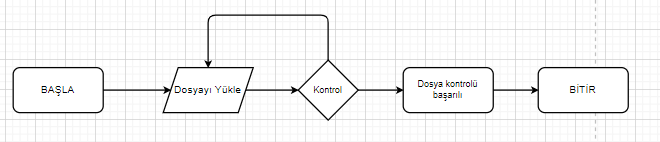
Kullanıcının sisteme müdahale edebileceği ekrana erişmesi için aşması gereken bir modüldür.

#### 4.3.2.1.2 Kullanıcı Arabirimi

#### 4.3.2.1.3 Modül Tanımı

Üniversite tezleri başta olmak üzere lisans, yüksek lisans ve doktora tezlerinde kopyacılık oranın otomatik hesabını tespit etmek için kullanılır ve sistem yüklenen dosyanın yazım kurallarına uygun olup olmadığını denetler.

#### 4.3.2.1.4 Modül iç Tasarımı



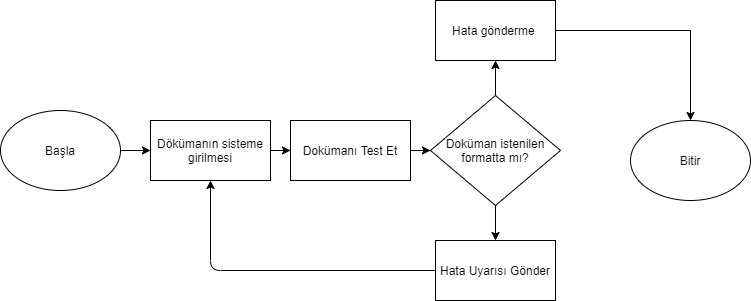
|  |  |
| --- | --- |
| METOD ADI | TANIM |
| Dosyaoku() | Dosyanın okuma işlemlerini gerçekleştirecek. |
| DosyaSeç() | Dosyanın Kullanıcı tarafından girilmesini sağlar |
| Rapor() | Sonuçların analiz edilip kullanıcı tarafından görüntülenmesini sağlar |

#### 4.3.3 Kullanıcı Profilleri

KULLANICI: Yapının tek nesnesidir. Tüm işlemlerde bulunur.

#### 4.3.4 Entegrasyon ve Test Gereksinimleri

Sistemimizin daha öncede belirttiğimiz gibi üniversitenin Tez kurallarına entegrasyon halinde olması gerekir.



# 5.SİSTEM GERÇEKLEŞTİRİMİ

## 5.1 GİRİŞ

Gerçekleştirim çalışması, tasarım sonucu ortaya çıkan sürecin fiziksel yapısını içerir. Fiziksel modelin bilgisayar ortamında çalışan yazılım biçimine dönüştürülmesi çalışmalarını içerir.

## 5.2 Yazılım Geliştirme Ortamları

Yazılım geliştirme ortamı, tasarım sonunda üretilen fiziksel modelin, bilgisayar ortamında çalıştırılabilmesi için gerekli olan:

• Programlama Dili

CASE Araçları belirlendi ve yazılım geliştirme ortamı hazırlandı.

### 5.2.1 Programlama Dilleri

Bu sistemde görsel programlamaya önem verildiğinden c# kullanılmıştır. C# barındırdığı

Kütüphane ve yapılabilirliği iyi olduğu için kullanılmıştır.

## 5.3 Kodlama Stili

Bu sistemin yazılım kısmı yapılırken herhangi bir kodlama stiline uyulmamıştır tamamen kendimize has kodlama stili ile yapılmıştır.

### 5.3.1 Açıklama Satırları

Yazılan kodun okunabilirliğini artırmak amacıyla Açıklama satırları her satırın sonunda yapıldı.

### 5.3.3 Anlamlı İsimlendirme

Sistem kodlamasının genel yapısında kullanılan değişkenlerin yaptığı işleme paralel isimler verildi.

### 5.3.4 Yapısal Programlama Yapıları

Genel olarak 3 başlıkta incelersek:

• Ardışık işlem yapıları: Bu tür yapılarda genellikle fonksiyon, altprogram ve buna benzer tekrarlı yapıları tek bir seferde çözdük.

• Koşullu işlem yapıları: Bu yapıları ise neredeyse programın tamamında kullandık karşılaştırma yapılan her yerde bunlara yer verildi.

• Döngü yapıları: Tıpkı ardışık işlemler gibi alt alta birkaç satır yazıcığımıza tek bir döngüyle bu sorunların üstesinden geldik.

## 5.4 Program Karmaşıklığı

Program karmaşıklığını ölçmek için birçok teorik model geliştirilmiştir. Bu modellerin en eskisi ve yol göstericisi McCabe karmaşıklık ölçütüdür. Bu bölümde bu ölçüt anlatılmaktadır. Söz konusu ölçüt 1976 yılında McCabe tarafından geliştirilmiştir. Bu konuda geliştirilen diğer ölçütlerin çoğu, bu ölçütten esinlenmiştir. McCabe ölçütü, bir programda kullanılan "koşul" deyimlerinin program karmaşıklığını etkileyen en önemli unsur olduğu esasına dayanır ve iki aşamada uygulanır:

### 5.4.1 McCabe Karmaşıklık Ölçütü Hesaplama

Kenar sayısı=7

Düğüm sayısı=7

Bileşen sayısı=1

V(G)=k-d+2p

V(g)=7-7+2\*1=2

## 5.5 Olağan Dışı Durum Çözümleme

Olağan dışı durum, bir programın çalışmasının, geçersiz ya da yanlış veri oluşumu ya da başka nedenlerle istenmeyen bir biçimde sonlanmasına neden olan durum olarak tanımlanmaktadır.

### 5.5.1 Olağandışı Durum Tanımları

Dosya türünün yanlış girilmesi tarafındaki sorunları try-catch blokları devreye girerek kullanıcıya mesaj gönderir.

## 5.6Kod Gözden Geçirme

Hiç kimse, önceki sürümlerini gözden geçirmeden ve incelemeden okunabilir bir program yazamaz. Hiçbir yazı editörün onayını almadan basılamayacağı gibi hiçbir program da incelenmeden, gözden geçirilmeden işletime alınmamalıdır. Kod gözden geçirme ile program sınama işlemlerini birbirinden ayırmak gerekir.

# 6.Doğrulama ve Geçerleme

## 6.1 Giriş

Geliştirmekte olduğumuz akademik tez yazma programının doğrulanması ve geçerleşmesi, üretim süreci boyunca süren etkinliklerden oluşmaktadır. Söz konusu etkinlikleri sıralayacak olursak:

1. Yazılım belirtimlerinin ve proje yaşam sürecindeki her bir etkinlik sonunda alınan çıktıların doğru, açık ve önceki belirtimleri tutarlı olarak betimler durumda olduğunun doğrulanması.
2. Proje süresince her bir etkinlik ürününün teknik yeterliliğinin değerlendirilmesi ve uygun çözüm elde edilene kadar aktivitenin tekrarına sebep olması.
3. Projenin bir aşaması süresince geliştirilen anahtar belirtimlerin önceki belirtimlerle karşılaştırılıp doğru sonuçlar elde edebilmesi.

Yazılım ürünlerinin tüm uygulanabilir gerekleri sağladığının gerçekleşmesi için sınamaların hazırlanıp yürütülmesi biçiminde özetlenebilir

Öncelikle iki aşamalı Doğrulama ve gerçekleme aşamalarını paylaşmak isteriz. Doğrulama aşamasında kendimize “Projenin gereksinimlerini karşılayacak doğru ürünü mü üretiyoruz?” sorusunu kendimize sorarız. Ürünü kullanacak akademik personel ve lisansüstü öğrencilerinin isteklerinin karşılanıp karşılanmaması sorgulanmaktadır.

İkinci aşamamız olan gerçekleme de ise “Üniversite tarafından yayınlanan lisans kurallarını doğru olarak kullanıcıya sunuyor muyuz?” sorusuna cevap aramaktayız. Eğer ürünün içsel niteliğine ilişkin izleme ve denetim etkinlikleri sağlanmadığı takdirde yazdığımız program bize çözümden çok sorun üreteceğini düşünüyoruz.

## 6.2 Sınama Kavramları

### Alt Sistem Sınama

Birimlerin birleşmesiyle modüller oluşturulup bunların kendi içinde sınaması yapıldı. Genel olarak ara yüzde oluşan mevcut eksiklikler giderildi.

### Sistem Sınama

Sistemin bütün olarak sınanması yapıldı ve programın belirli birkaç yerinde eksiklik olduğu onaylandı.

### Kabul Sınama

Sistem prototipten çıkartılıp dosya okuması için veriler girildi ve sorunsuz bir şekilde çalıştığı onaylandı.

## 6.3 Sınama Yöntemleri

Sınama yöntemlerinde geliştirmeyi izleyen bir düzeltme görevinden ziyade sistemin daha çok sistemin var olan eksiklikleri ve hataları hakkında bilgi sahibi olundu. Sisteme örnek tez dokümantasyonun dışında farklı bir dosya girildiği taktirde programımız halen çalışmasını sürdürmüştür. Bu açıdan bakıldığında sınama konusunda programımız eksiklikler olmasına rağmen gayet sağlıklı ve başarılı bir şekilde çalışmaktadır.

## 6.4 Sınama ve Bütünleştirme Stratejileri

Genellikle sınama stratejisi, bütünleştirme stratejisi ile birlikte değerlendirilir. Ancak bazı sınama stratejileri bütünleştirme dışındaki tasaları hedefleyebilir. Örneğin, yukarıdan aşağı ve aşağıdan yukarı stratejileri bütünleştirme yöntemine bağımlıdır. Ancak işlem yolu ve gerilim sınamaları, sistemin olaylar karşısında değişik işlem sıralandırılması sonucunda ulaşacağı sonuçların doğruluğunu ve normal şartların üstünde zorlandığında dayanıklılık sınırını ortaya çıkarır.

## 6.5 Sınama Planlaması

Mevcut sistemi bir tablo halinde kısaca özetlemek gerekirse şu şekilde özetlememiz daha doğru olacaktır.

Öncelikle test raporu hazırlanırken alt kısımda belirtilen özellikler daima ön planlarda tutulmalıdır. Bunların netliği de belirtilmelidir. Kısaca listeleyecek olursak;

Test Planı kimliği: Projeye başlamadan önce projenin adının belirtilmesi

Giriş: Test edilecek sistemdeki yazılım elemanlarının genel tanıtım özetlerini oluşturan kısımdır. Burada kısaltmalar ve terim açıklamaları açık şekilde bildirilmelidir.

Test edilecek sistem: Sistemde bileşenleri sürüm sayıları olarak sıralarız ve sistemin özelliklerini ve işletim sisteminin gereksinimlerini açıkça kullanıcıya bildiririz. Sistemin nasıl kullanılacağından, test edilmeyecek taraflara kadar her şey belirtilmelidir.

Test edilecek ana fonksiyonlar: Sistemin doğru çalışması için ana fonksiyonların kısa bir tanıtımı yapılır.

Test edilmeyecek ana fonksiyonlar: Sistemde test edilmeyecek fonksiyonlar ve bunların neden test edilmedikleri açıklanmaktadır.

Geçti/Kaldı Kriterleri: Bir test sonucunda sistemin geçmiş veya kalmış olduğunu belirten kriterler burada toplanır.

Test Dokümanı: Test süresince yapılan işlemlerin rapor halinde sunulmasını sağlamaktadır.

Sorumluluklar: Proje içerisinde bulunan kişilerin nelerden sorumlu olduğu, test bakım lideri bilgileri mutlaka raporda belirtilerek sunulmalıdır.

Riskler ve Önlemler: Test planında varsayılan ve olası yüksek riskli durumları belirtir ve bu durumların olması durumunda, etkilerinin en aza indirilebilmesi için alınması gereken önlemleri açıklar.

## 6.6 Sınama Belirtimleri

Sınama belirtimleri, bir sınama işleminin nasıl yapılacağına ilişkin bilgiler ve ayrıntılar içermektedir. Bu ayrıntılar içerisinde temel olarak birkaç unsurlar bulunur. Bunlar:

1. Sınanan program modülü veya modüllerin isimleri
2. Sınama yöntemi veya türü
3. Sınama verileri
4. Sınama senaryoları

Türünde bilgileri içermektedir.

Sınama verilerinin bildiğiniz üzere elle hazırlanması çoğu zaman kolay bir durum değildir. Tam aksine bu süreç yoğun zaman alıcı ve bir o kadar da zor olabilir. Bizim yapmış olduğumuz Tez düzenleyici programında gereksinimlere göre birkaç hatanın önüne geçilmesi ve hataların kullanıcı tarafından kolaylıkla algılanması amaçlanmaktadır.

# 7.Bakım

## 7.1 Giriş

Projemizin tasarımı bittikten sonra artık projemizin bakım evresine gelebiliriz. Tez düzenleyici programımızda genel nitelikler ve şartlar belirli olduğu için fazla bakıma ve gereksinime ihtiyaç duymayacaktır.

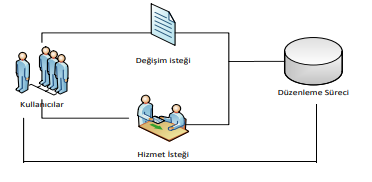
## 7.2 Kurulum

Tez düzenleyici programımızın kurulumu gayet basit olmakla birlikte kaynak kodlarını da github üzerinde bulabilirsiniz. Github üzerinden erişim sağlayarak dosyayı indirdikten sonra gerekli .NET Framework sisteminin kurulması durumunda programımız sorunsuz bir şekilde çalışacaktır. Kullanıcının yapması gereken tek şey yazmış olduğu Tez dokümanını program tarafından seçerek çalıştırmaktır.

## 7.3 Yerinde Destek Organizasyonu

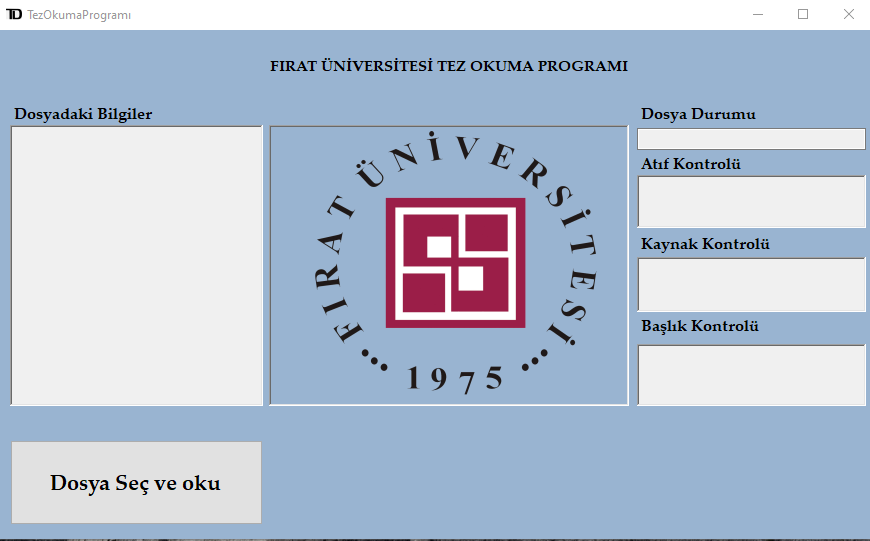
Programımız için yerinde destek organizasyonu bulunmamaktadır. Zaten programda meydana gelecek hasar ve hatalar bizlere geri besleme olarak döndüğü takdirde sorunun çözümü hakkında iyileştirmeler ve çözümler sağlanacaktır.

## 7.4 Yazılım Bakımı



Bakım Aşaması

## 7.5 Yazılım Ara yüzü



Yazılım Ara yüzünü gösteren resim

Yazılım ara yüzünü tasarlarken gayet basit ve anlaşılabilir bir sistem yapısı kurulmaya özen gösterilmeye çalışılmıştır. Kullanıcının yazılımı kolaylıkla yükleme sağlaması amacıyla kurulum talimatları ve yapması gerekenler basitçe tanımlanmıştır.

# 8.Sonuç

Sonuç olarak programdaki amacımız tez yazım aşamalarında meydana gelen hataları azaltarak bir nebze olsun kullanıcının hatalarını görmesini ve ön izlemesini amaçlamaktadır. Başlık kontrolü, kaynak kontrolü, atıf kontrolü ve dosya durumunu gösterebilir.

Kolay kurulum ve kullanıcı dostu bir yazılım projesi yapmak için yola çıktığımız bu yolda kullanıcının hiçbir bilgisayar bilgisi olmadığını varsayarak kurulum dosyası ve kurulum talimatlarını da programın içerisinde belirterek projemizi teslim ediyoruz.