FIRAT ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

YAZILIM MÜHENDİSLİĞİ – PROGRAMLAMA DİLLERİ DERSİ

YAZIM FORMAT DÜZENLEYİCİ

PROJE ÇALIŞMA GRUBU

16542014-MUHAMMED HAKAN ÇELİK

16542009-OLCAY KAYMAZ

16542017-FEHMİ DENİZ BİÇER

16542006-NİHAT ÇETİN

170541055-MUSA KARABULUT

1.GİRİŞ

1.1 Projenin Amacı

1.2 Projenin Kapsamı

2.PROJE PLANI

2.1 Giriş

2.2 Yazılım Plan Kapsamı

2.3 Zaman-İş Planı

2.4 Yazılım Ekip Yapısı

2.5 Maliyet Kestirimi

2.6 Önerilen Sistemin Teknik Tanımları

2.7 Kullanılan Özel Geliştirme Araçları ve Ortamları

2.8 Yazılım Standartları, Yöntem ve Metodolojiler

2.9 Kalite Sağlama Planı

2.10 Konfigürasyon Yönetim Planı

2.11 Kaynak Yönetim Planı

3.SİSTEM ÇÖZÜMLEME

3.1 Mevcut Sistem İncelemesi

3.1.1 Örgüt Yapısı

3.1.2 İşlevsel Model

3.1.3 Veri Modeli

3.1.4 Var olan Yazılım/Donanım Kaynakları

3.1.5 Var olan Sistemin Değerlendirilmesi

3.2 Gereksenen Sistemin Mantıksal Modeli

3.2.1 Giriş

3.2.2 İşlevsel Model

3.2.3 Genel Bakış

3.2.4 Bilgi Sistemleri/Nesneler

3.2.5 Veri Modeli

1.GİRİŞ

1.1 Projenin Amacı

Genel Amacı

Tez ve dokümantasyon hazırlarken birçok kural hatası yapılmaktadır. Bu sorunlar; başlık boyutunun yanlış ayarlanması, başlık yazılmaması, yazı tipinin uygun seçilmemesi gibi sorunları meydana getirmektedir.
Çözüm bulmak için otomata teorilerinden yararlanarak yazarın yükünü hafifletmeyi amaçlamaktadır.

Özel Amacı

Otomata teorilerini kullanarak;
Belediyeler,
Ulusal kamu kurum ve kuruluşları,
Üniversiteler,
gibi kurumsal ve kişisel kullanım için de istenilen formatta dokümantasyon yazmayı kolaylaştırmayı hedeflemektedir.

2.PROJE PLANI

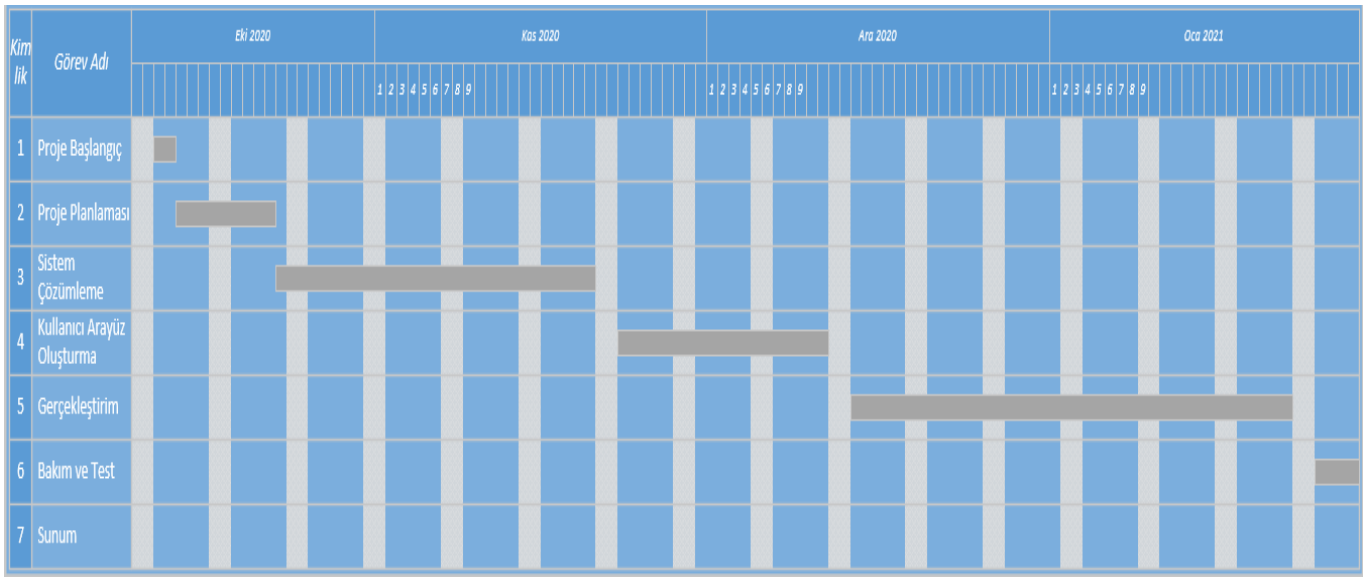
2.1 Giriş

İstenilen format dışında yazılan ve unutulan kısımlar dokümantasyon yazarken sıkça başımıza gelmektedir.

2.2 Yazılım Plan Kapsamı

Tez, dokümantasyon yazımı günden güne artmaktadır. Bu artış sürecinde bizler bazı kuralları gözden kaçırabiliyor ve fark etmekte sıkıntı yaşayabiliyoruz. Bu sıkıntılar otomata teorisi ile gelinen noktada çözülebilecek türdedir. Yazım format düzenleyici bağlığı altında ele almak istediğimiz bu sıkıntıların önüne geçmek ve daha doğru, uygun biçimde dokümantasyon yazabilmektir.

2.3 Zaman-İş Planı



2.4 Yazılım Ekip Yapısı

İnsan kaynaklarında belirttiğimiz kişiler projemizde yer alacaktır.

Projemizde yer alacak kişiler ve görevleri ;

- Proje yöneticisi → Projeyi hem teknik hem de iş modeli seviyesinde yönetebilecek nitelikteki kişilerdir.
- Programcı → Tasarım ve kodlama işlerini yapan kişidir.
- Test Uzmanı → Geliştirme safhasında tamamlanan ürün birimlerinin veya ürünün tamamının testinde görev alan kişidir.
- Yazılım Ekibi → Projenin teknik ve iş modeli kısımlarında, işlerini yapabilecekleri kadar iş modeli bilgisine sahip kişilerdir.
- Araştırma Ekibi → Proje için gerekli kaynakların belirlenmesine öncülük eden kişilerdir.

2.5 Maliyet Kestirimi

Ölçüm Parametresi	Sayı	Ağırlık	Toplam
Kullanıcı Çıktı Sayısı		4	
Kulanıcı Girdi Sayısı		3	
Kullanıcı Sorgu Sayısı		3	
Kütük Sayısı		7	
Dışsal Arayüz Sayısı		5	

Teknik karmaşıklık sorusu	**Puan**
1. Uygulama güvenilir yedekleme ve kurtarma gerekiyor mu?	2
2. Veri iletişimi gerekiyor mu?	5
3. Dağıtık işlevleri var mı?	4
4. Performans kritik mi?	5
5. Sistem mevcut ve ağır yükü olan bir işletim ortamında mı çalışacak?	5
6. Sistem çevrimiçi veri girişi gerektiriyor mu?	5
7. Çevrimiçi veri girişi, bir ara işlem için birden çok ekran gerektiriyor mu?	5
8. Ana kütükler çevrimiçi olarak mı güncelleniyor?	5
9. Girdiler, çıktılar, kütükler yada sorgular karmaşık mı?	3
10. İçsel işlemler karmaşık mı?	2
11. Tasarlanacak kod, yeniden kullanılabilir mi?	3
12. Dönüştürme ve kurulum tasarımda dikkate alınacak mı?	2
13. Sistem biden çok yerde yerleşik farklı kurumlar için mi geliştiriliyor?	5
14. Tasarlanan uygulama, kolay kullanılabilir ve kullanıcı tarafından kolayca değiştirilebilir mi olacak?	4
Toplam	55

0 : Etkisi yok 1 : Çok az etkili 2 : Etkisi var 3 : Ortalama etkisi var 4 : Önemli etkisi var

5 : Çok önemli etkisi var

İN : İşlev noktası
AİN : Ana işlev nokta sayısı
TKF : Teknik karmaşıklık faktörü

İN : AİN * (0,65 * 0,01 * TKF)
İN : 401 * (0,65 * 0,01 * 60)
İN : 156,39
Satır Sayısı : İN * 30 = 4691,7
Etkin maliyet modeli : COCOMO
Projemiz küçük ekip tarafından geliştirildiği için organik projeler sınıfına girdiğinden;
Organik proje : a= 2,4 b= 1,05 c= 2,5 d=0,38
Aylık kişi başı iş gücü : E = a * (KSS)b
Geliştirme süresi : D = c * (E)d
Eleman sayısı : E / D
KSS : kod satır sayısı

Aylık kişi başı iş gücü = 1,89
Geliştirme süresi = 4 ay
Eleman sayısı = 3,18 / 1,89 = 2 (Yaklaşık)

2.6 Önerilen Sistemin Teknik Tanımları

Dokümantasyon yazımı kapsamında birçok yerde kullanılması hedeflenmektedir. Şu anda projemin devamında staj dokümantasyonunu için pilot olarak belirledik.

Bu sorunu çözmek için bir uygulama geliştirmeyi öngörüyoruz. Önereceğimiz sistem offline olarak kullanılabilmektedir. Masaüstü sistemler için tasarlanacaktır. Sistem esnek yapıya sahip olan Java tabanlı olarak geliştirilecektir.

2.7 Kullanılan Özel Geliştirme Araçları ve Ortamları

Çözümleme ve Tasarım Araçları

- Gantt diyagramı için Visio kullanılmıştır

Programlama Araçları

- Programlama dili olarak JAVA seçilmiştir ve Intellij IDEA kullanılacaktır.

Sınama Araçları

- Windows

2.8 Yazılım Standartları, Yöntem ve Metodolojiler

Agile, yazılım projelerinde müşterinin ihtiyaçlarına daha çabuk ve geliştirilebilir şekilde cevap verebilmek için kullanılmaya başlamış olan yaklaşımdır. Agile yaklaşımının temel prensiblerinden biri yazılımın erken ve sürekli teslimini sağlayarak müşteriyi memnun etmektir. Çalışan yazılımı kısa zaman aralıklarında düzenli olarak müşteriye sunmaktadır.

2.9 Kalite Sağlama Planı

Sistemin kalite sağlama planında SPICE (ISO 15504) modeli referans alınacaktır. Bu modeli seçme nedenimize değinecek olursak ;

ISO 15504 yazılım süreç değerlendirme sistemi standardı bir belgelendirme amacı taşımamaktadır. Bu standart belgelendirmeye yönelik değil , teknoloji sektörünün kabul edilen yazılım çalışmalarında kaliteyi artırmaya yöneliktir.

Modelin önemli prensiplerinden biri de , yazılım ürünlerinin doğasından kaynaklanan güçlüklerin önüne geçmektir.

ISO 15504 yazılım süreç değerlendirme sisteminin tanımladığı süreçler şunlardır :

- Yazılım satın alma süreçleri
- Yazılım geliştirme süreçleri
- İşletim süreçleri
- Bakım ve destek süreçleri
- Planlama, yönetim,gerçekleştirim, denetim ve iyileştirme süreçlerini kapsamaktadır.

SPICE

ISO / IEC 15504

2.10 Konfigürasyon Yönetim Planı

Proje başlangıcında oluşturulan proje planında olası değişiklik yapmak söz konusudur. Değişimler eğer düzgün bir şekilde ele alınmazsa projenin teslim tarihi, projenin kalitesinin azalması gibi faktörlerin ortaya çıkmasına neden olur.

Bu yüzden bu değişimlerin kontrol edilmesi için konfigürasyon yönetim planı oluşturulmuştur.

Olası veri tabanına eklenecek bilgilerin veya silinecek bilgilerin kontrolü sağlanmıştır.

2.11 Kaynak Yönetim Planı

Bir yazılım projesi planlanırken, projede kullanılacak kaynaklar dikkatlice ele alınmalıdır. Proje kaynaklar, insan kaynakları, donanım kaynakları, yazılım kaynakları incelenmektedir.

2.12 Eğitim Planı

Projede kullanılacak dillerin ve projenin belgelemesi yapılabilmesi için bazı eğitimler alınması gereklidir.
Proje kapsamı alınacak olan eğitimler;

- Arayüz kullanım eğitimi

2.12 Test Planı

Tez, belgeleme, dokümantasyon çalışması yapacak kullanıcılarının çalışmalarının yazım kriterlerinin doğruluğunu kontrol etmek amacıyla denenecektir.

2.13 Bakım Plan

Çalışan projeye ilerleyen zamanlarda ek ihtiyaçlar doğrultusunda güncellemeler yapılacaktır.

2.14 Projede Kullanılan Yazılım/Donanım Araçlar

<u>Yazılım:</u>

- Python programlama dili
- İlerleyen zamanlarda web teknolojileri

<u>Donanım:</u>

- Kişisel bilgisayar(PC)

3. Sistem Çözümleme

3.1 Mevcut Sistemin İncelenmesi

Bu kısımdaki temel amacımız tez, dokümantasyon yazımının anlaşılması ve tanımlanmasıdır. Gereksinimlerin belirlenmesi için izlenecek en güzel yöntemlerden biri standart şablondaki kuralların bilinmesidir.

3.1.1 Örgüt Yapısı

Sınıf arkadaşlarının ortak çalışmalarıyla oluşturulan örgüt yapısı vardır.

3.1.2 İşlevsel Model ve Mimari Tasarım

Projemiz Masaüstü Ekranı'ndan oluşan tek ana modülden oluşmaktadır. Masaüstü uygulaması JAVA ile yazılacaktır.

Yüklenen dosya masaüstü uygulamasına yüklendikten sonra program çalıştırılır.

3.1.3 Veri Modeli

Veri tabanı bulunmamaktadır.

3.1.4 Var Olan Yazılım/Donanım Kaynakları

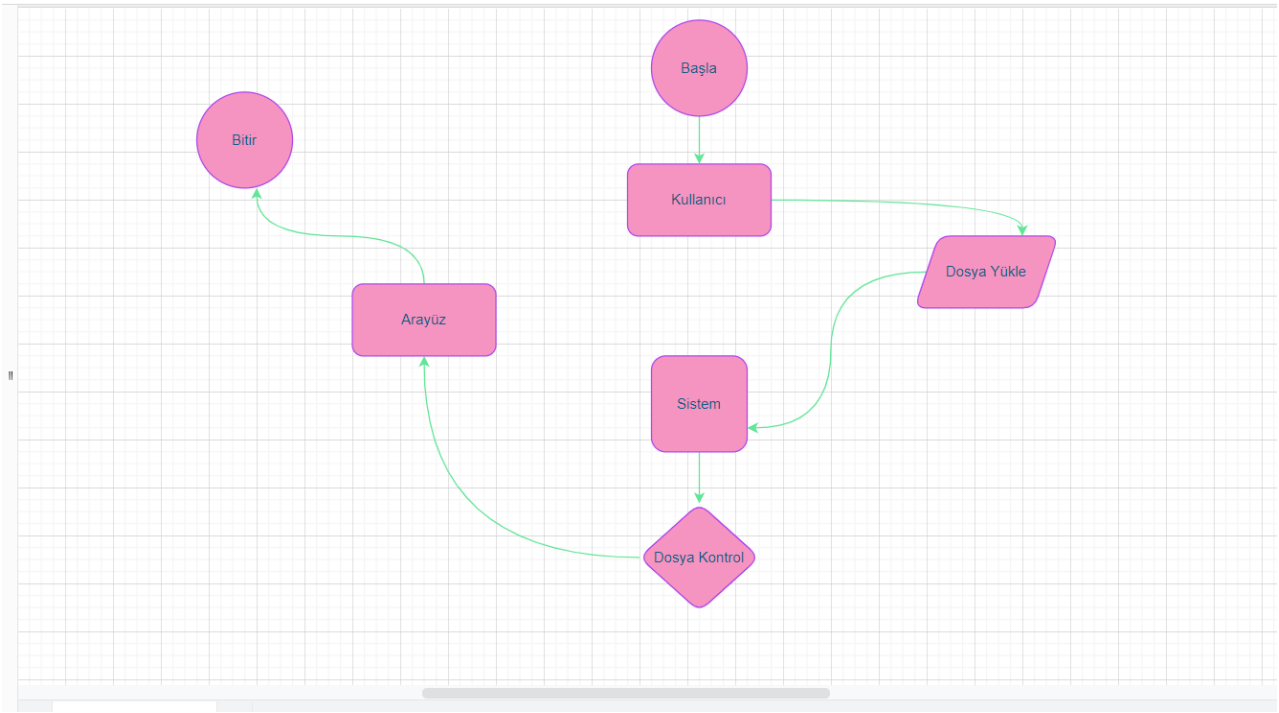
Donanım Kaynakları

- Dizüstü Bilgisayarlar

Yazılım Kaynakları

- Windows Server / Windows 10 Pro

3.2.7 İşlevlerin Sıradüzeni



Her yeni araç girişinde veri tabanına giden bilgiler güncellenecek ve yönetici masaüstü uygulaması sayesinde bu bilgilere erişebilecek.

3.2.8 Başarım Gerekleri

Mevcut sistem incelenmiş ve mevcut sistemin eksiklerinden yola çıkılarak, sistemin başarımı için:

- Sistemin sonuç üretim doğrulukları
- Tepki sürelerinin en aza indirilmesi
- Maliyetin en aza indirilmesi
- Hata risklerinin en aza indirilmesi
- Kullanım kolaylığı
- Anlaşılabilirlik

temel gereklilikler olarak tespit edilmiştir.

3.3 Arayüz(Modül) Gerekleri:

3.3.2 Kullanıcı Arayüzüne Genel Bakış

Uygulama daha çok yönetici odaklı dizayn edilmiştir. arayüzde almaktadır.

3.4 Belgeleme Gerekleri:

3.4.1 Geliştirme Sürecinin Belgelenmesi:

Geliştirme sürecini belgelemek ileriye dönük olarak projeye katılan geliştirme ekibinin ve kullanıcıların projeyi daha iyi anlaması açısından etkin rol oynamaktadır. Projenin geliştirilme sürecinde ileriye dönük olarak tasarımsal hataların tespitinde önemli bir yer almaktadır. Bunun yanı sıra proje ekibinin projeye olan hâkimiyeti açısından oldukça önemlidir.

3.4.2 Eğitim Belgeleri:

Proje çok karmaşık bir yapıdan oluşmadığı için projenin eğitim belgesi bulunmamaktadır.

3.4.3 Kullanıcı El Kitapları:

Projenin kullanımı oldukça basit olduğundan arayüzde kısa açıklamalardan oluşan kullanım kılavuzu adlı bölüm yayınlanacaktır.

4.SİSTEM TASARIMI

4.1 GENEL TASARIM BİLGİLERİ

4.1.1 GENEL SİSTEM TANIMI

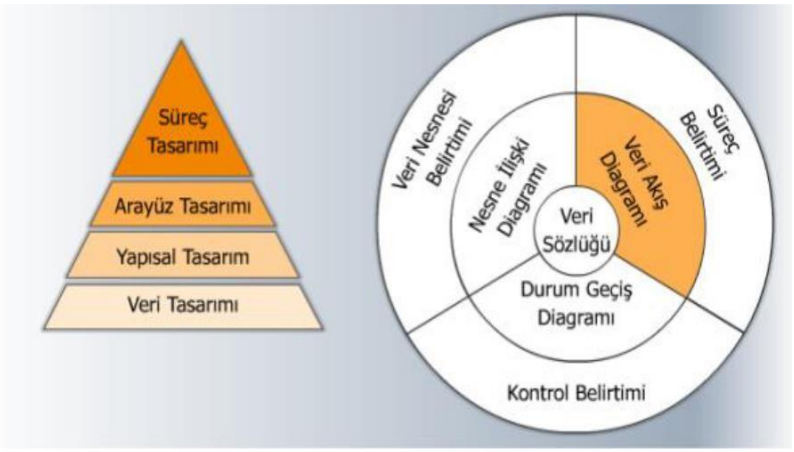
Bu kısımda sistemin genel yapısından bahsedeceğiz.

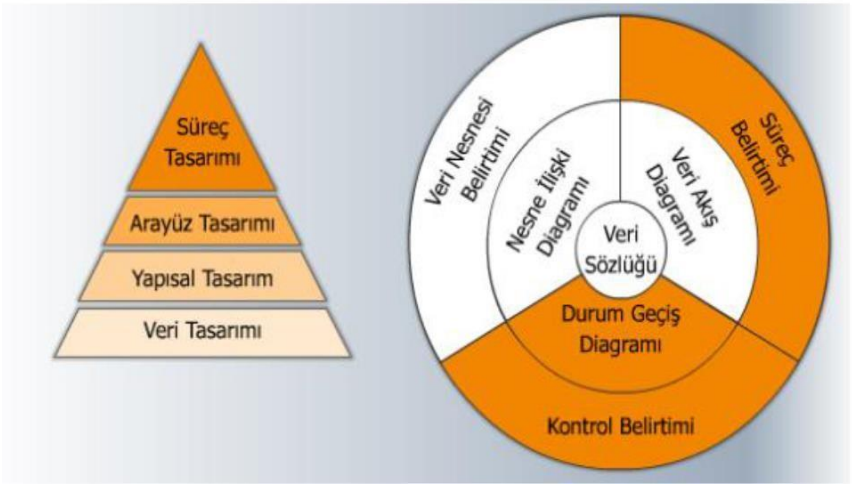
-Gereksinimler ve İşlevsellik

Öncelikle sistemin ne yapacağı hakkında bilgi verelim. Yazım Format Düzenleyici ile yaşanan yazım karmaşıklığın önüne geçilecek. Sistem birim arayüzü ile dokümantasyon istenilen formatın dışına çıkmayacak ve karmaşıklık yaşamadan sistem tarafından çözüme kavuşacak.

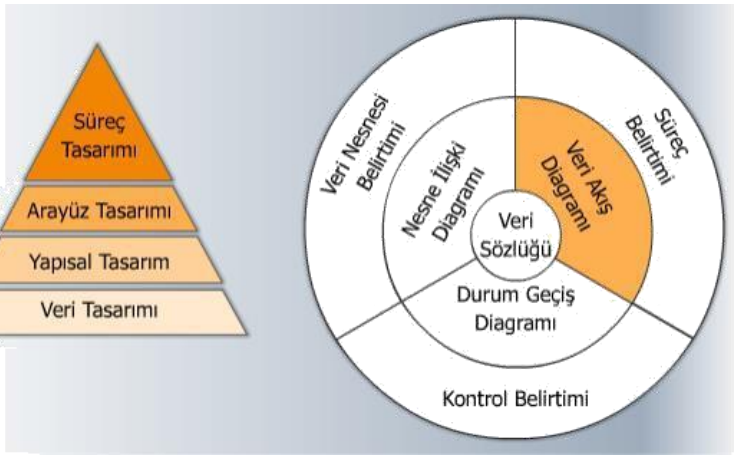
-Tasarım

1-İlk önce bir süreç tasarımı olacak ve aşağıdaki gibi olacak.

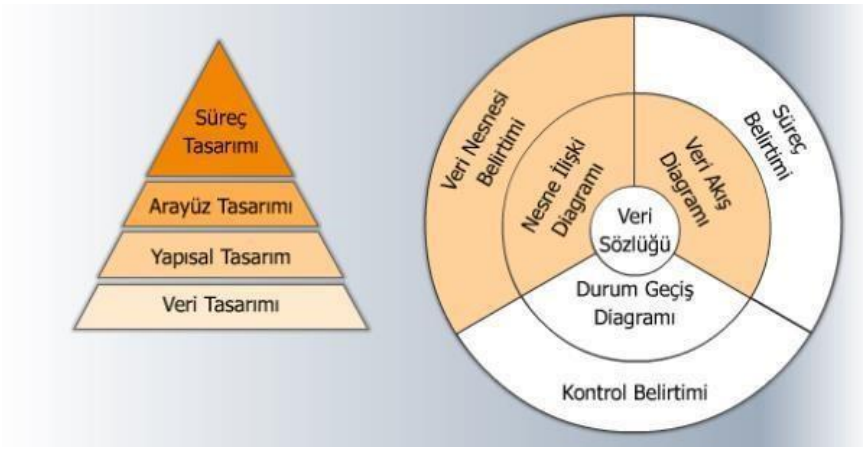




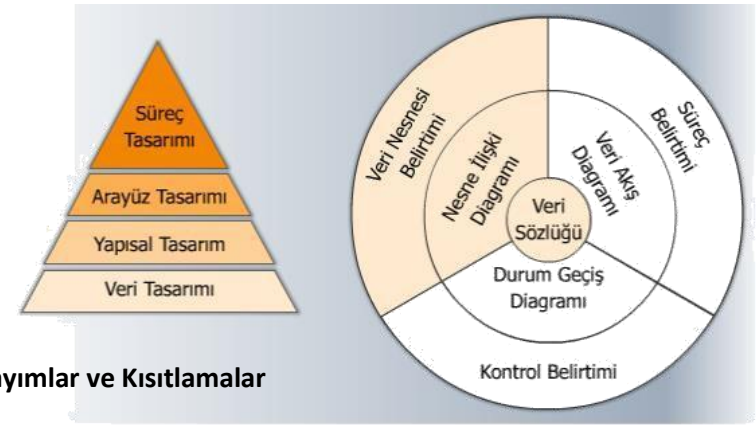
2.Süreç tasarımı bittikten sonra sıra Arayüz tasarımına geldi ve arayüz tasarımı aşağıda göründüğü adımlarla gerçekleştirildi.



3. Arayüz tasarımını aştıktan sonra sora yapısal tasarıma geldi. Yapısal tasarımda izlenen yollar aşağıdaki gibi oldu.



4.Yapısal tasarımı da yapıp Veri tasarımına geçildi ve şu adımlar izlendi.



4.1.2 Varsayımlar ve Kısıtlamalar

Sistemdeki varsayılan başlık, alt başlıklar, paragraf fontu ve büyüklüğü gibi bazı kısıtlamalar mevcuttur.

4.1.4 Dış Arabirimler

4.1.4.1 Kullanıcı Arabirimleri Kullanıcı ara yüzünde ilk olarak giriş ekranı gelir. Girişlerde yönetici ve personel girişleri aynıdır.

4.1.4.2 Veri Arabirimleri

Veri tabanı kullanımına ihtiyaç duyulmamaktadır.

4.1.6 Testler

Genel hatlarıyla testlerimiz iki aşamada gerçekleştirilecek. Bilinen adıyla pilot bölge uygulaması yapılacak.

Alfa Aşaması: Sistem geliştirildiği yerde farklı dokümantasyonlarla test edilecektir.

Beta Aşaması: Kullanıcı, geliştirilen sistemi grup eşliğinde gerekli testler yapılacaktır.

4.1.7 Performans

Test aşamasından sonra elde edilen kullanıcı tepkilerden, kullanıcı geri dönüşlerinden, işleyiş performansı, stabilizesi ve zaman temelli amaca uygunluğu performans değerlendirmesi için belirlenen faktörlerdir.

4.2 Veri Tasarımı

4.2.1 Tablo Tanımları

Projede veri tabanı ihtiyacı duyulmamaktadır.

4.2.3 Veri Tanımları Sistem kullanıcıları sistemi doğrudan kullanabilmektedir.

4.3 Süreç Tasarımı

4.3.1 Genel Tasarım Genel olarak tasarımda ilk önce kullanıcı ara yüzleri daha sonra proje modülleri oluşturuldu. En sonda modüller ilişkilendirildi.

4.3.2 Modüller

4.3.2.1 Giriş Modülü

4.3.2.1.1 İşlev

Kullanıcının sisteme erişimi için gereken giriş modülüdür.

4.3.2.1.3 Modül Tanımı Kullanıcı ilk karşılaştığı arayüzdür.

4.3.2.1.2 Kullanıcı Arabirimi

4.3.3 Kullanıcı Profilleri

Sistemimizde tek giriş profili mevcuttur.

4.4 Ortak Alt Sistemlerin Tasarımı

4.4.1 Ortak Alt Sistemler

Ortak olan alt sistem bulunmamaktadır.

4.4.2 Modüller Arası Ortak Veriler

Modüller arasında nesne yönelik ilişkisel veri modelinde olduğu üzere ortak veriler mevcuttur.

4.4.3 Güvenlik Alt Sistemi

Yazılım sistemlerinin güvenilirliğe ilişkin nicelikleri, kullanıcıların gereksinimlerini karşılayacak şekilde ortaya koymak ve güvenilirliğin hesaplanmasına yönelik verileri toplama, istatistiksel tahminlime, ölçütlerin tespiti, yazılıma ait mimari özelliklerin belirlenmesi, tasarım, geliştirme ve bunlara yönelik çalışma ortamının belirlenmesi ve modellenmesini kapsamaktadır.

1. Model seçme ve düzenlemeye yönelik faaliyetlerin temelinde uygun hedeflerin tespit edilmesi bulunmaktadır.
2. Hata ve aksaklıkların analiz edilmesi için uygun verilerin tanımlanması gerekmektedir. Örneğin, arıza veya hataları önemine göre sınıflandırmak, hatalar arası ortalama süreyi bulmak, hata nedenlerini araştırmak, hataları bulmaya yönelik test verilerine karar vermek.
3. Belirtilen hedeflere yönelik veriler modellenir.
4. Geçmişe yönelik verilerin zaman bilgilerini de içerecek şekilde elde edilerek yazılım geliştirme sürecine dâhil etmek.
5. Yazılım geliştirme sürecinin modellenmesi, hata ile karşılaşılıp, test sürecine başlamak ve model doğrulama işlemlerine gerçekleştirmek.
6. Güvenilirlik tahminleme modelinin seçilmesini sağlamak.
7. Güvenilirlik modeli tarafından kullanılacak olan parametrelerini tespit etmek.
8. Verilen bir noktayı kullanarak gelecekteki olası hatalar hakkında tahmin yapmak.

Tahmin edilen hata ve arıza oranları ile gerçekleşen değerleri kıyaslamak.

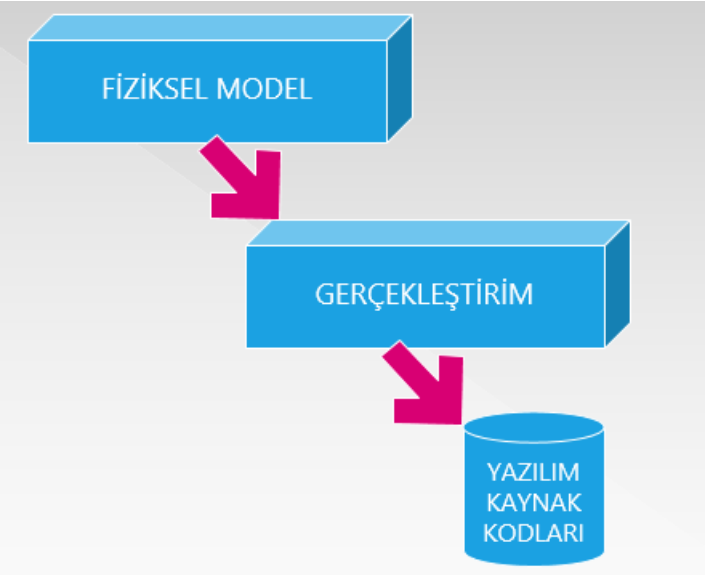
4.4.5 Veri Dağıtım Alt Sistemi

Sistem veri dağıtım ve veri alma üzerine kurulmuştur.

5. SİSTEM GERÇEKLEŞTİRİMİ

5.1 Giriş

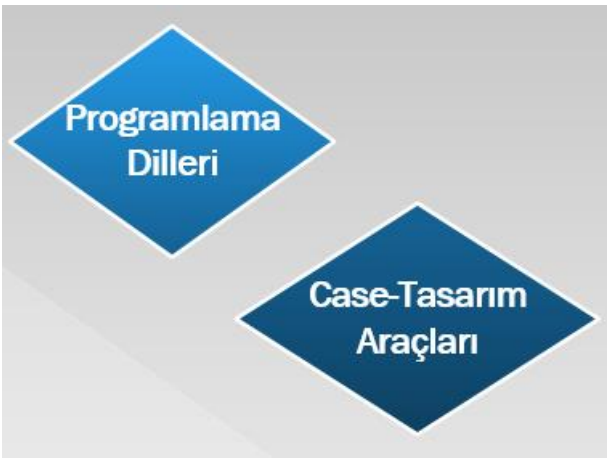
Gerçekleştirim bölümü, tasarım kısmında oluşturulan fiziksel modelin programlama dilleri ve araçları kullanılarak gerçeğe dönüştürülmesini içerir. Fiziksel modelin gerçekleştirimi için de en başta bu süreçlerin takibine yardımcı olacak bir yazılım geliştirme ortamı gerekmektedir.



5.2 Yazılım Geliştirme Ortamları

Yazılım geliştirme ortamı, tasarım kısmında oluşturulan fiziksel modelin gerçeğe dönüştürülmesinde kullanılacak olan:

- Programlama Dilleri
- Tasarım Araçları
- Veri Tabanı Yönetim Sistemi (VTYS) gibi araçların olduğu bir ortamdır. (Ancak projede database bulunmamaktadır.)



5.2.1 Programlama Dilleri

Projemiz masaüstü uygulaması olarak kullanılmaktadır. Kullanıcı tarafından Word ya da PDF formatında yüklenen dosya çeşitli işlemler sonucunda kontrol edilmektedir.

5.2.2 Case Araç ve Ortamları

Sistemin diyagramlarını ve tasarımlarını Microsoft Visio ile ile oluşturduk.

5.3 Kodlama Stili

Sistemin kodlama işleminin daha kolay ve anlaşılabilir olması için sisteme özel belirli bir kodlama stili olması gerekmektedir. Kodlama stili 4 aşamadan oluşur.

5.3.1 Açıklama Satırları

Program kod satırlarının arasına kodlarda düzenleme yapacak yetkili personellerin kullanması adına açıklama (yorum) satırları yerleştirilmiştir.

5.3.2 Kod Biçimlemesi

Programın okunabilirliğini artırmak amacıyla oluşturulmuş biçimlemedir.

Örnek:

```
if x == y:
        return x;
  else:
        return y;
```

5.3.3 Anlamlı İsimlendirme

Program kodlarındaki değişken isimleri Camel Case (degiskenAdi) tipinde ve değişkenin yaptığı işi açıklayacak biçimde verilmiştir.

Hangi değişkenlerin yalnızca ilgili kod içerisinde ara değişkenler olarak kullanıldıkları

5.3.4 Yapısal Programlama Yapıları

Program kodlarının, okunabilirlik, anlaşıla bilirlik, bakım kolaylığı gibi kalite etmenlerinin sağlanması ve program karmaşıklığının azaltılması amacıyla "yapısal programlama yapıları" kullanılarak yazılması önemlidir.

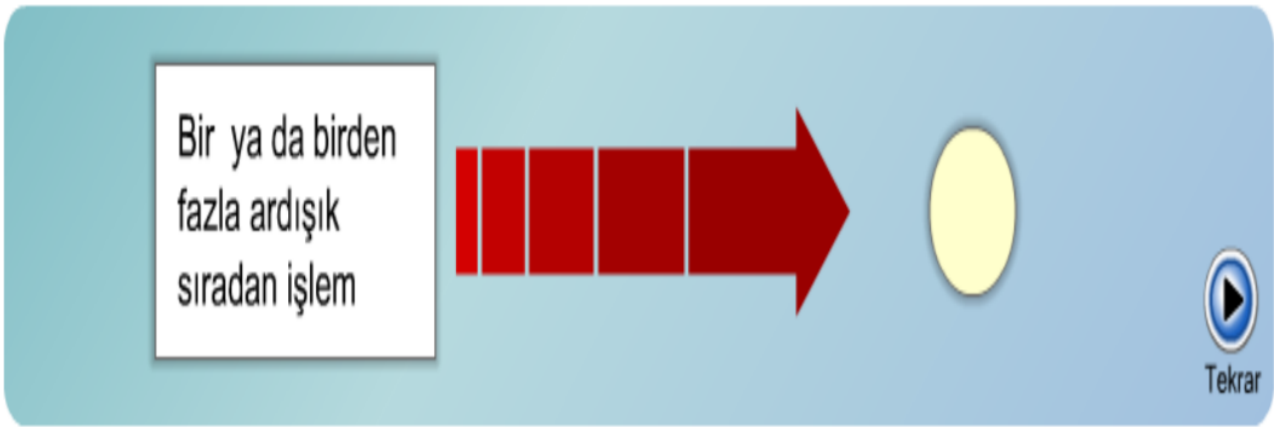
Yapısal programlama, büyük çaptaki projelerin daha küçük parçalara ayrılarak kodlanması ve sorunlarının bu sayede daha hızlı giderilmesi yöntemidir. Bu yöntem, üç tür yapılanma kullanır:

- Sıralı (Ardışık) Yürütme
- Mantıksal (Şartlı) Yürütme
- Döngü

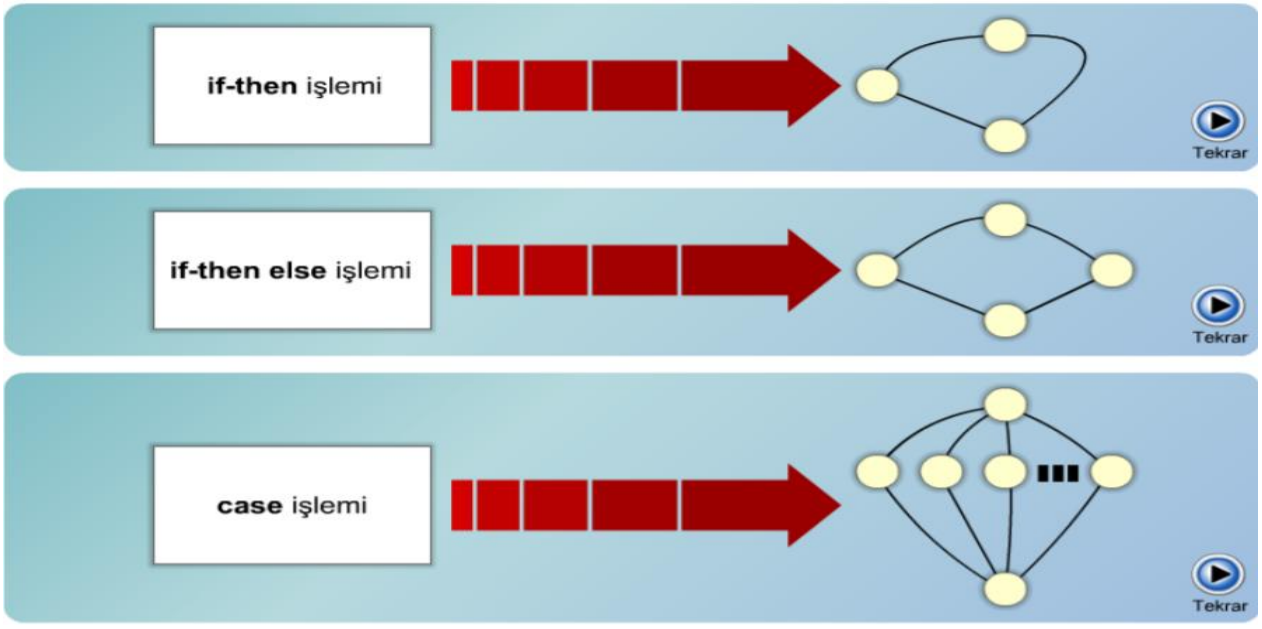
5.4 Program Karmaşıklığı

5.5.1 Programın Çizge Biçimine Dönüştürülmesi

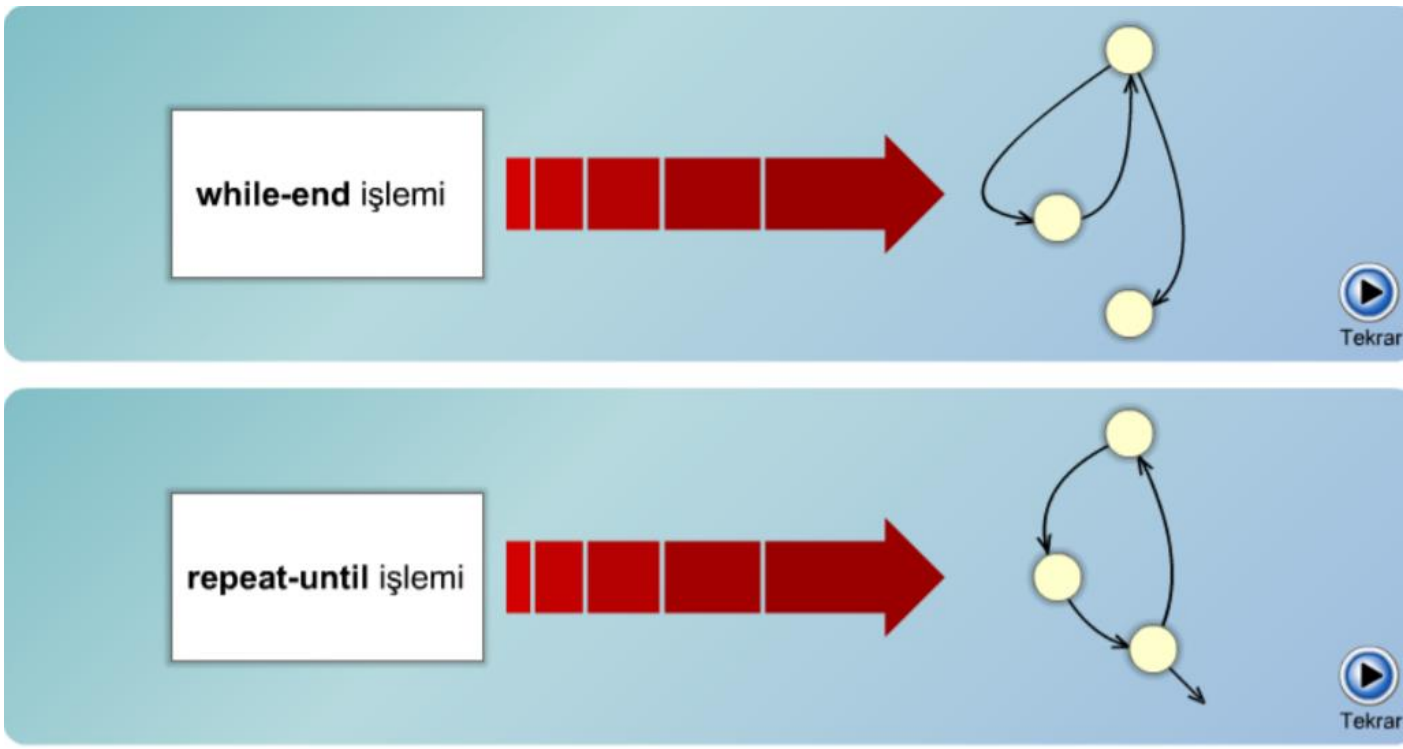
1. Sıradan İşlemler: Bir sıradan işlem ya da birden fazla ardışık sıradan işlem çizgede bir düğüme dönüştürülür.



2. Koşullu İşlemler:



3. Döngü işlemleri:



5.5 Olağan Dışı Durum Gözlemleme

Olağan dışı durum, program çalışırken oluşabilecek istenmeyen durumlardır. Burada önemli olan, sistemin bu durumlar karşısında nasıl davranacağı ve kullanıcıyı nasıl yönlendireceğidir.

5.5.1 Olağandışı Durum Tanımları

- Dosyanın bozuk olması.
- Dosyanın istenen formatta olmaması
- İşlemlerde zaman aşımı
- İnternet olmaması
- İnternet bağlantısının zayıf olması

5.5.2 Farklı Olağandışı Durum Çözümleme Yaklaşımları

Dosyanın bozuk olma durumda, “Dosya bozuk, lütfen kontrol ettikten sonra tekrar deneyiniz.” mesajı gösterilecek ve programdan çıkılacaktır.

Dosyanın istenen formatta olmadığında (pptx) vb. “dosya istenilen formatta değildir, lütfen docx veya pdf uzantılı dosya girin.” Mesajı gösterilecek ve yeniden dosya seçimi yapmaya yönlendirilecektir.

Herhangi bir işlemde zaman aşımı olduğunda kullanıcıya bekleme, tekrar deneme veya işlemi sonlandırma seçenekleri sunulacaktır.

İnternet olmadığında, "İnternet bağlantısı yok, lütfen daha sonra tekrar deneyin." mesajı gösterilecek ve programdan çıkılacaktır.

İnternet bağlantısı zayıf olduğunda, "İnternet bağlantısı zayıf, ne yapmak istersiniz?" diye bir soru sorulacak ve kullanıcılara bekleme veya sistemden çıkma seçenekleri sunulacaktır.

5.6 Kod Gözden Geçirme

5.6.1 Gözden Geçirme Sürecinin Düzenlenmesi

5.6.2 Gözden Geçirme Sırasında Kullanılacak Sorular

- Ürün eksiksiz olarak tamamlanmış mı?
- Planlama aşamasındaki problemin çözümünü sağlıyor mu?
- Ürün UML diyagramları tutarlı mı?
- Kullanıcı arayüzünde kullanım kolaylığı esas alındı mı?
- Program kodunda açıklama satırları anlamlı bir şekilde oluşturuldu mu?
- Program kodunda yapılan kısmi güncellemeler, kodun bütünlüğüne zarar verdi mi?
- Ürün orijinal dosyaya zarar veriyor mu?
- Program beklenmedik bir hata ile karşılaştığında nasıl üstesinden gelecek?

5.6.2.1 Öbek Arayüzü

Oluşturulan öbeklerin (kod bütünü) testi ve doğrulaması için sorulan belirli sorular vardır. Bunlar:

- Her öbek tek bir işlevsel amacı yerine getiriyor mudur?
- Öbek adı, işlevini açıklayacak biçimde anlamlı olarak verilmiş midir?
- Öbek tek girişli ve tek çıkışlı mıdır?

Öbek eğer bir işlev ise, parametrelerinin değerini değiştiriyor mudur?

5.6.2.2 Giriş Açıklamaları

Sistemin doğru anlaşılabilmesi için başta giriş açıklamaları oluşturmuştur. Bu açıklamaları test etmek ve doğrulamak için belli sorular sorulmuştur. Bunlar:

- Öbek, doğru biçimde giriş açıklama satırları içeriyor mudur?
- Giriş açıklama satırları, öbeğin amacını açıklıyor mudur?
- Giriş açıklama satırları, parametreleri, küresel değişkenleri içeren girdileri ve kütükleri tanıtıyor mudur?
- Giriş açıklama satırları, çıktıları ve hata iletilerini tanımlıyor mudur?

5.6.2.3 Veri Kullanımı

Oluşturulan veri kullanımlarının testi için belirli sorular sorulmuştur. Bunlar:

- İşlevsel olarak ilişkili bulunan veri elemanları uygun bir mantıksal veri yapısı içinde gruplanmış mıdır?
- Programda değişken isimleri belirtilen stilde yazıldı mı?
- Değişken adları, işlevlerini yansıtacak biçimde anlamlı mıdır?
- Değişkenlerin kullanımları arasındaki uzaklık anlamlı mıdır?
- Her değişken tek bir amaçla mı kullanılıyor?
- Dizin değişkenleri kullanıldıkları dizinin sınırları içerisinde mi tanımlanmış?
- Tanımlanan her gösterge değişkeni için bellek ataması yapılmış mıdır?

5.6.2.4 Öbeğin Düzenlenişi

Oluşturulan modellerin düzenlenişi için belirli sorular sorulmuştur. Bunlar:

- Modüllerin birleşimi uyumlu mudur?
- Modüller arası veri aktarımları sağlanıyor mudur?
- Bütün modüller birleştiğinde sistem çalışıyor mudur?

5.6.2.5 Sunuş

Sunuş kısmında ise şu sorular sorulmaktadır:

- Her satır, en fazla bir deyim mi içermektedir?

- Bir deyimin birden fazla satıra taşması durumunda, bölünme anlaşılabilirliği kolaylaştıracak biçimde anlamlı mıdır?
- Koşullu deyimlerde kullanılan mantıksal işlemler yalın mıdır?
- Bütün deyimlerde, karmaşıklığı azaltacak şekilde parantezler kullanılmış mıdır?
- Bütün deyimler, belirlenen program stiline uygun olarak yazılmış mıdır?

6.DOĞRULAMA VE GEÇERLEME

6.1 Giriş

Geliştirmiş olduğumuz uygulamada bir Word dosyası baştan sona kontrol ediliyor. Girdi ve çıktı sayıları kontrol ediliyor.

Geliştirmiş olduğumuz bu uygulama kullanıcı isteklerini tam olarak karşılıyor mu? Bu soruda doğru şeklinde geri dönüş almamız gerekiyor

Doğrulama: Doğru ürünü mü üretiyoruz. Ürünü kullanacak kişilerin isteklerinin karşılanıp karşılanmadığına dair etkinliklerden oluşur. Tasarım gereksinimleri uygulamak için yeterli mi?

Geçerleme: Ürünü doğru olarak mı üretiyoruz? Ürünün içsel niteliğine ilişkin izleme ve denetim etkinliklerinden oluşur. Yapılan bu uygulama kullanıcı gereksinimini karşılıyor mu?

Tüm nu işlemler başka kişi veya şirketler tarafından yapılabilir.

Avantajları	Dezavantajları
Uzman olan kişiler yapar	Daha pahalıdır
Kurum içi karmaşadan etkilenmez	Uygulamayı anlama/kavrama aşaması vardır
Hatalar erken çözülür.	Uygulama 3. Kişiler tarafından bilinmektedir bu da istenmeyen bir durumdur.

6.2 Test Doğrulama Planı

Test İçin Kullanılan Yöntemler

Uygulamaya çalıştığınız herhangi bir yazılım testi aşağıdaki boyutları içermektedir. Bunlar:

Testi Yapanlar: Yazılımı testi işlemini kim yapıyor?

Yapılan uygulama geliştiriciler tarafından test edilir. Uygulama test edildikten sonra kullanıcılar tarafından sistem kapsamlı bir teste sokulur. Kullanıcılar tüm fonksiyonları denerler ve hata varsa hataları tekrardan geliştiricilere bildirirler.

Potansiyel Problemler: Yazılım testi neden uygulanıyor?

Belirlenen fonksiyonların her biri için görevini yerine getirip getirilmediği kontrol edilir. Ortaya çıkabilecek hataları veya sorunları önceden belirleyip kullanıcıya yansıtmadan sorunların giderilmesi amaçlanır.

Doğrulama: Yazılım testinin başarılı olduğu nasıl belirlenecek?

Belirli kullanıcıların sistemi kullandıktan sonra geri bildirim olarak sistemdeki sorunları belirtirse bu uygulama tekrardan geliştiriciler tarafından düzeltilecektir. Eğer herhangi bir problemle karşılaşılmaz ise test başarılıdır. Aksi halde hataların giderilmesi amaçlanır.

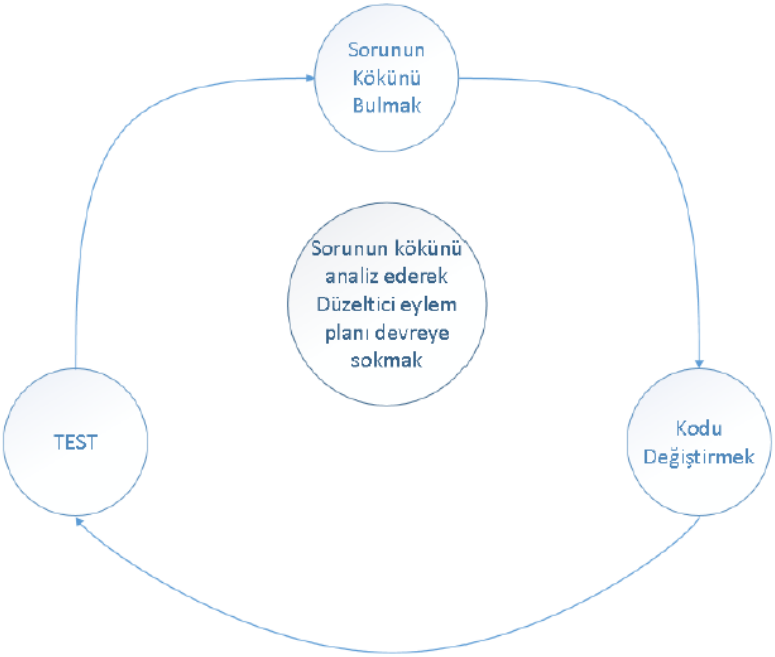
6.3 Kullanılacak Test Araçları

- **Python IDE**
- **Xcode**

7. BAKIM

7.1Giriş

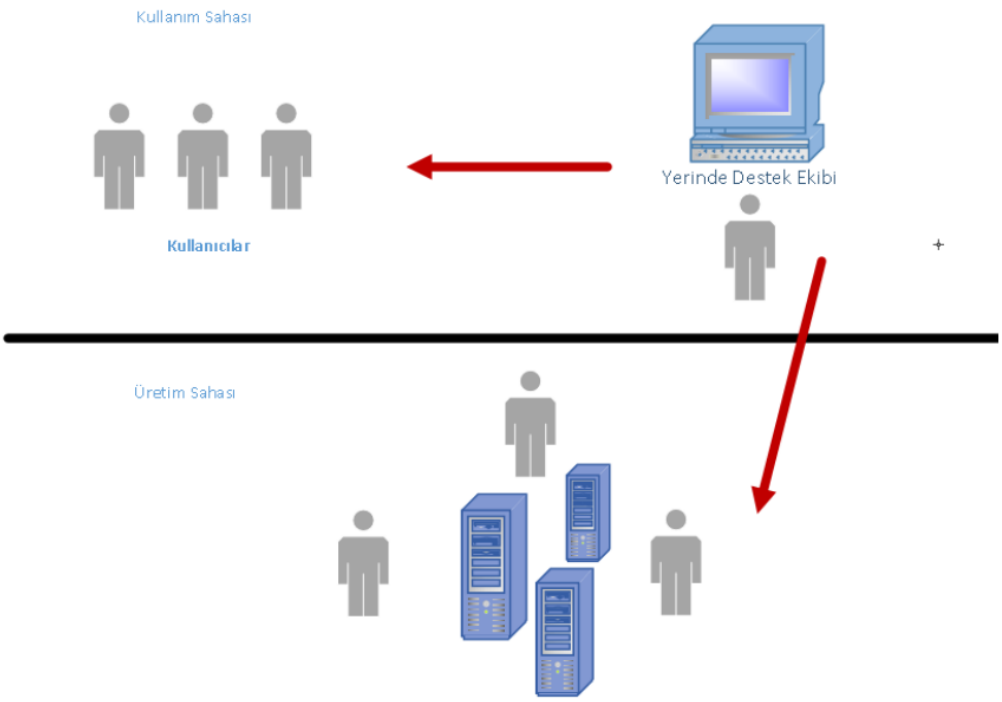
Sınama işlemleri bitirilen yazılımın kullanıcı alanına yüklenmesi ve uygulamanın başlatılması gerekmektedir. Yazılım kullanıma geçtikten sonra, yaşam döngüsünün en önemli ve hiç bitmeyecek aşaması olan "bakım" aşaması başlar. Bankacılık Denetleme ve Düzenleme Kuruluna uygun bir şekilde sürekli değişen kurallar çerçevesin de yapılacak hassas ve hata kabul etmeyecek bir sistemi düzenlemek ve bakımını gerçekleştirmek için yapacağımız açıklamalar da IEEE 1219-1998 standardı baz olarak alınmıştır.



Şekil 7.1 Bakım Planı

7.2Kurulum

Sistem kurulumu Github'tan ulaşabilir MAC/WİNDOWS işletim sistemlerinde rahatlıkla kullanılabilir.

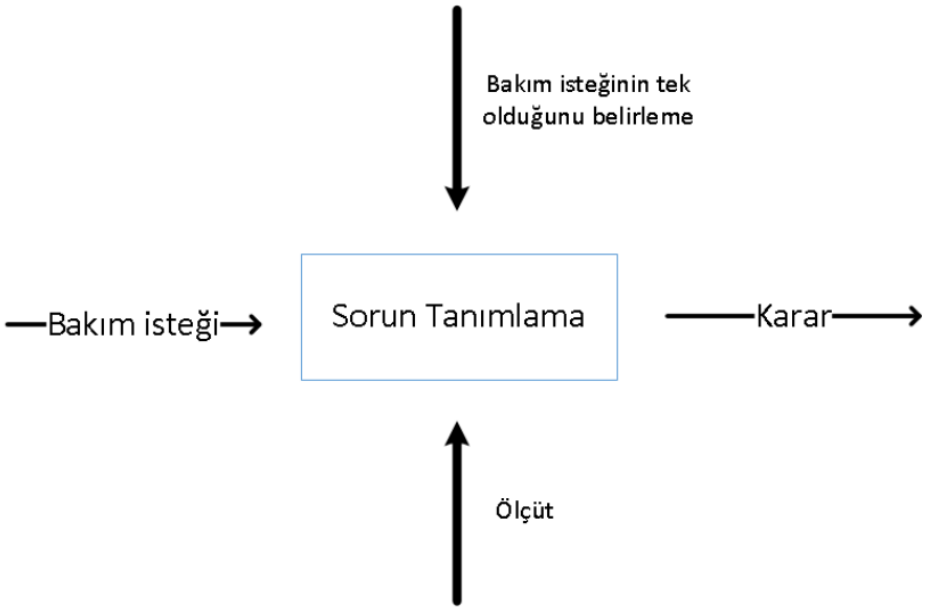


Şekil 7.2 Bakım Aşaması

7.3Yerinde Destek Organizasyonu

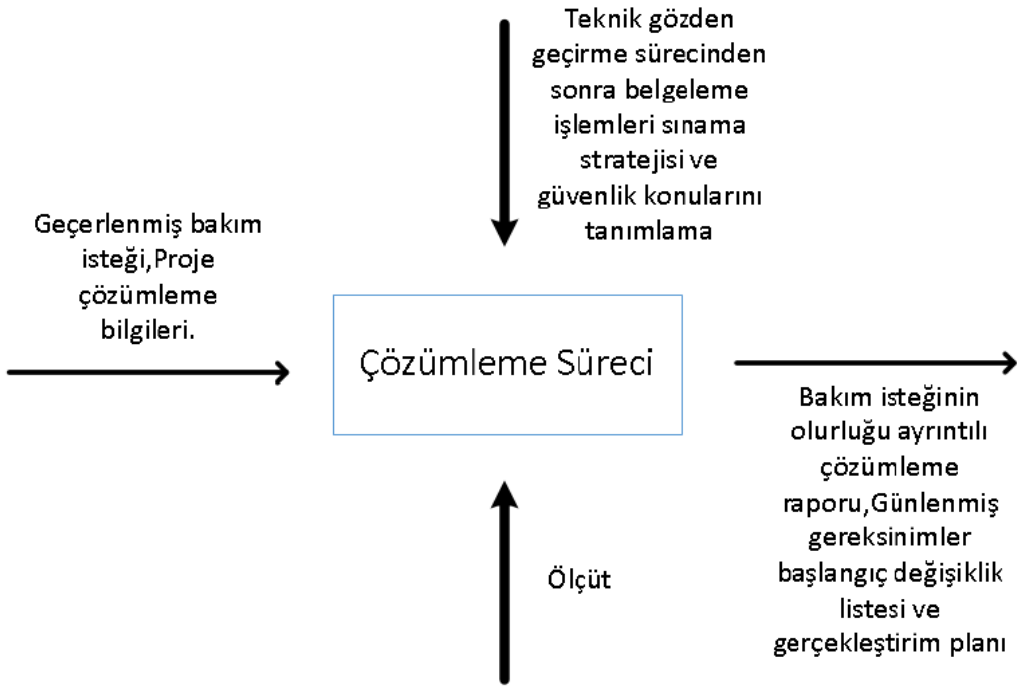
7.3.1 Bakım Süreç Modeli

1. Adım: Sorunu Tanımlama Süreci



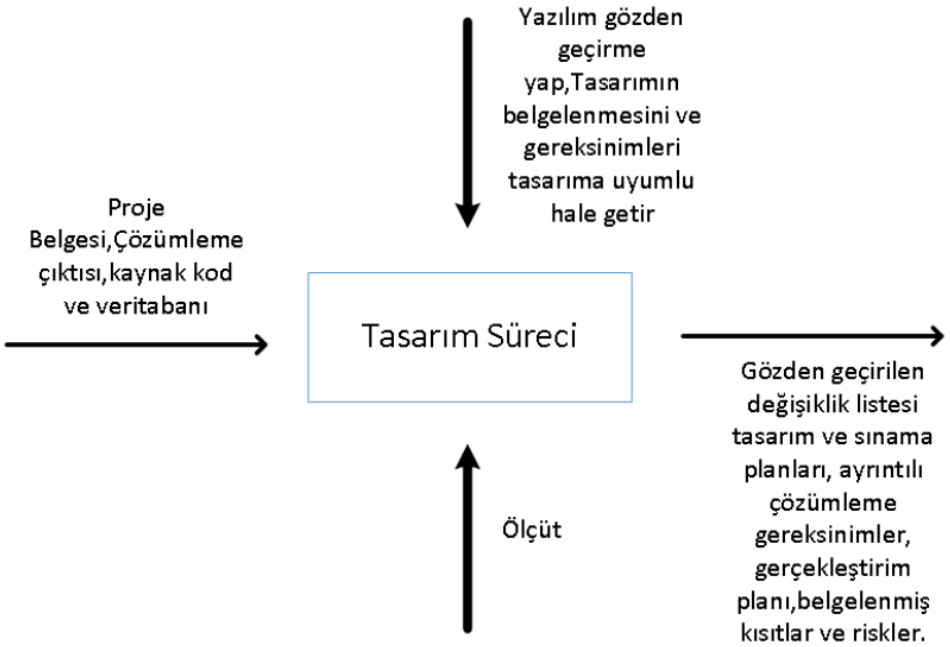
Şekil 7.3 Sorun Tanımlama

2. Adım: Çözümleme Süreci



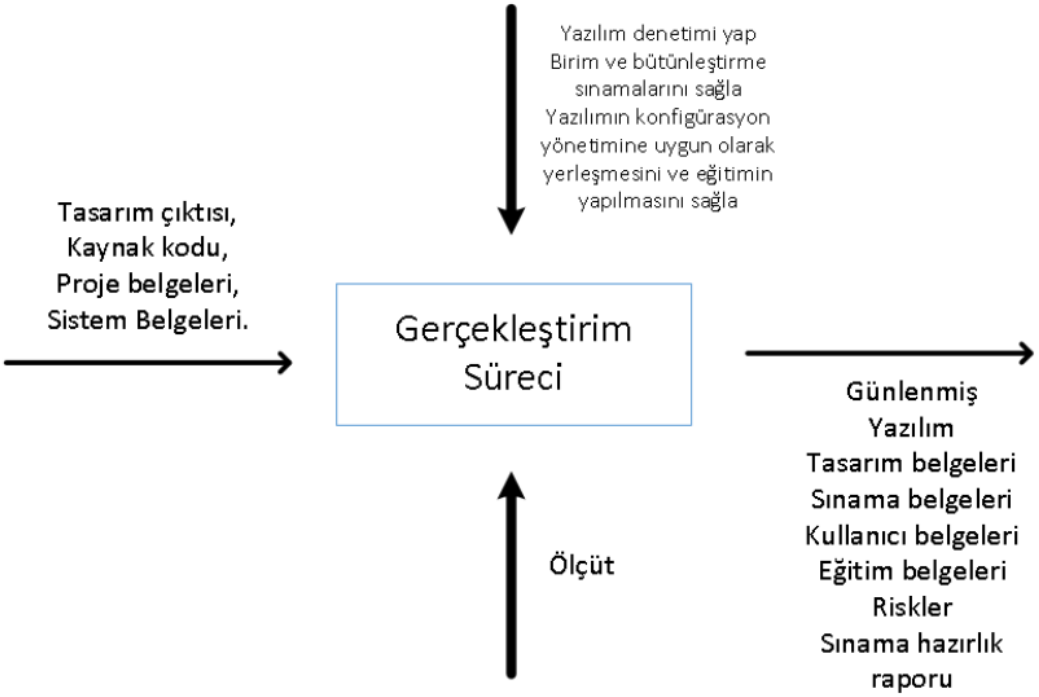
Şekil 7.4 Çözümleme Süreci

3. Adım: Tasarım Süreci



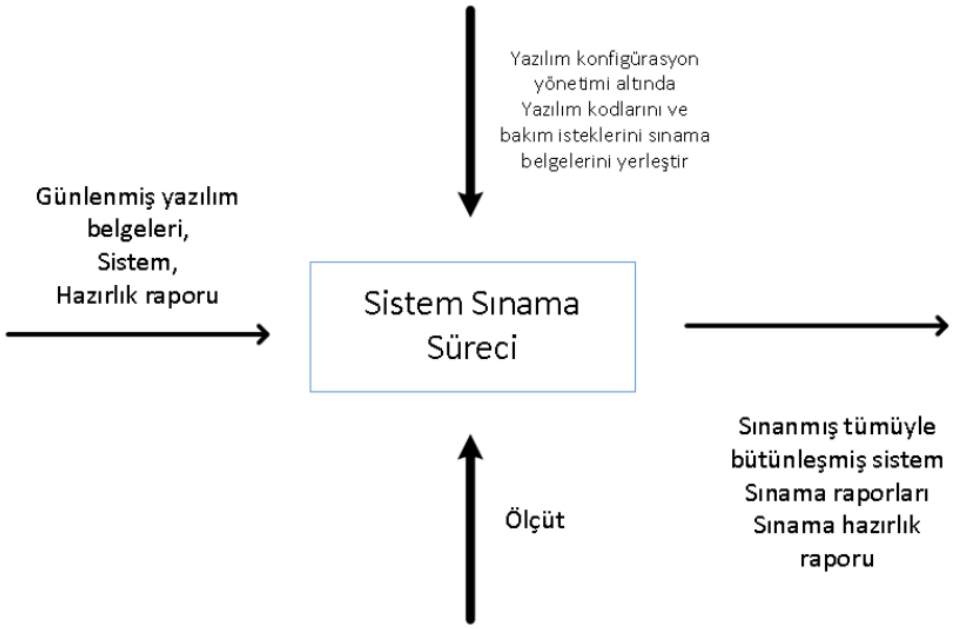
Şekil 7.5 Tasarım Süreci

4. Adım: Gerçekleştirim Süreci



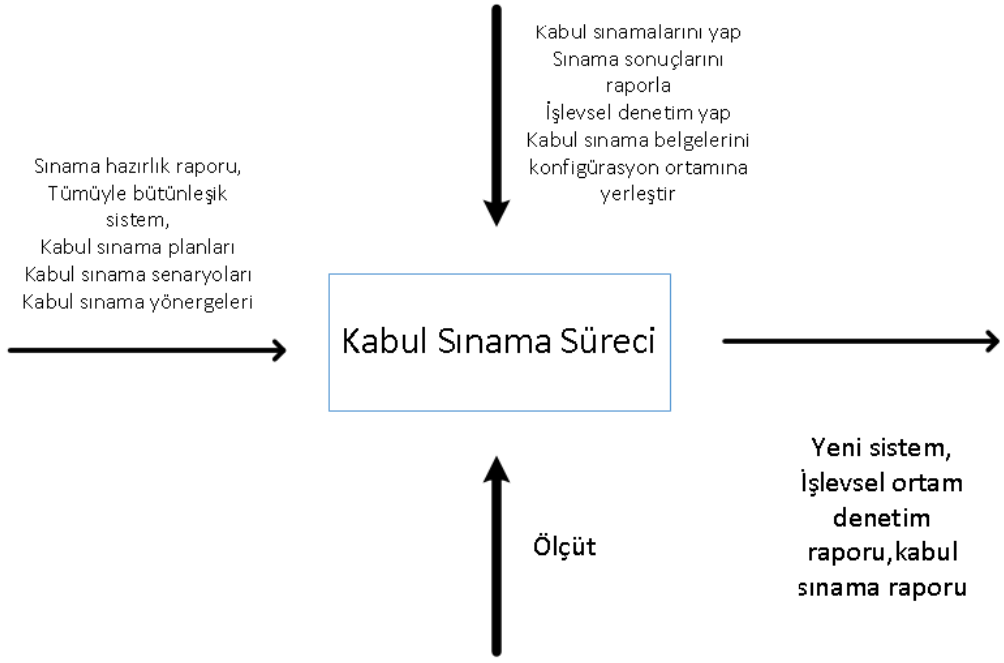
Şekil 7.6 Gerçekleştirim Süreci

5.Adım: Sistem Sınama Süreci



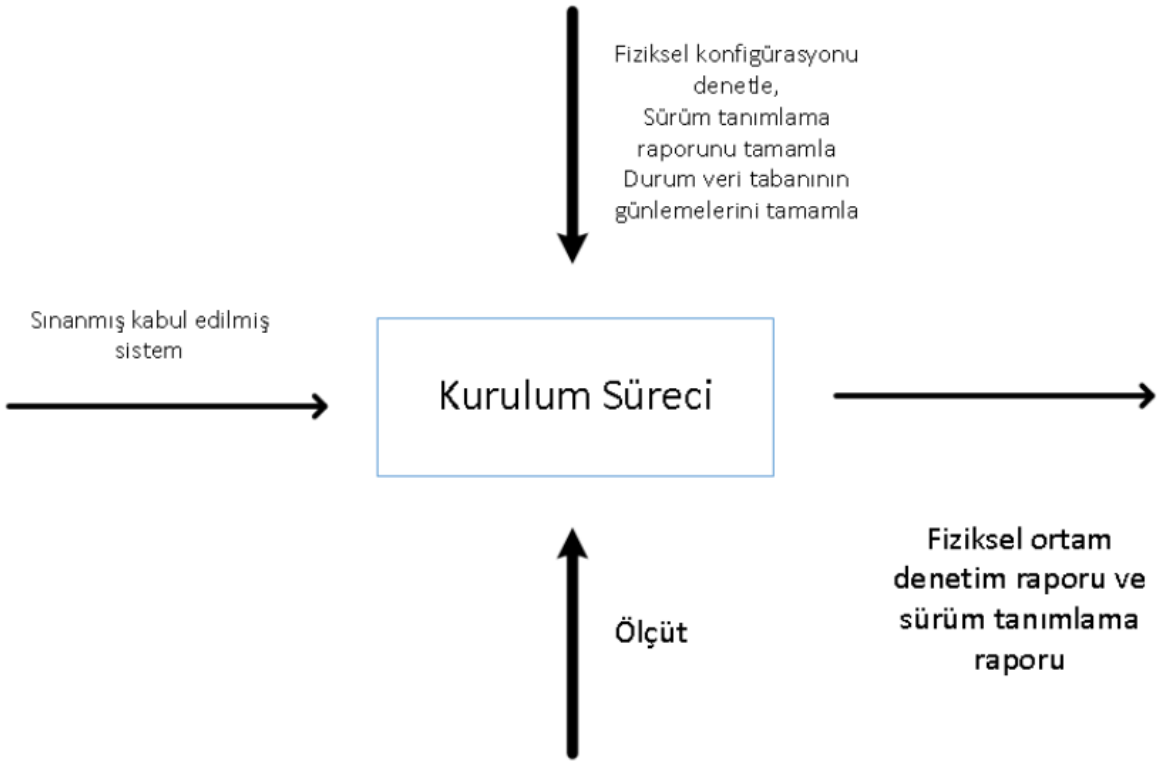
Şekil 7.7 Sistem Sınama Süreci

6.Adım: Kabul Sınaması Süreci



Şekil 7.8 Kabul Sınama Süreci

7.Adım: Kurulum Süreci



Şekil 7.9 Kurulum Süreci

8.SONUÇ

Kullanıcı programa girdi olarak verdiği Word belgesinin ayrıntılı ve detaylı şekilde incelenmiş raporunu yine Word belgesi olarak elde eder.

Kaynakça

https://www.youtube.com/watch?v=VvlEmuEK6VU&list=PLOAdLv-QeWyo-_pmprNd7J_mYd9Yd5vIH&index=2

https://python-docx.readthedocs.io/en/latest/user/documents.html

https://www.yazilimbilisim.net/python/python-tkinter-ornekleri/

https://python-istihza.yazbel.com/nesne_tabanli_programlama6.html

https://stackoverflow.com/questions/24805671/how-to-use-python-docx-to-replace-text-in-a-word-document-and-save