simge, sembol, logo, amblem, daire içeren bir resim

Açıklama otomatik olarak oluşturuldu

**KONYA TEKNİK ÜNİVERSİTESİ**

**YAZILIM MÜHENDİSLİĞİ DERSİ**

**FİNAL ÖDEVİ**

**Web Tabanlı Muhasebe Uygulaması**

**211213011**

**Berke TOPBAŞ**

**İçindekiler**

1. Projenin Tanımı--1
2. Projenin Amacı--1
3. Projenin Sahip Olduğu Nitelikler--1
4. Yazılım Geliştirme Yaşam Döngüsü --2
5. Artımsal Geliştirme Süreç Modeli--3
6. Artımsal Geliştirme Süreç Modeline Göre Projenin Gerçekleştirilmesi -4
7. Planlama--4
8. Proje Kapsamının Belirlenmesi--4
9. Proje Kaynaklarının Belirlenmesi—4

* Proje Ekibi--4
* Donanım ve Yazılım Kaynakları--5
* Proje Maliyetinin Belirlenmesi --5

1. Proje İş –Zaman Planı Oluşturulması--5
2. Sistem Çözümleme --12
3. Gereksinimlerin Belirlenmesi --12

* Fiziksel Ortam--12
* Arayüz--12
* Kullanıcılar--12
* İşlevsellik--12
* Veri ve Güvenlik--13
* Kalite ve Kaynak--13

1. Sistem Çözümleme Çalışması --13

* Mevcut Sistemin İncelenmesi ve Değerlendirilmesi--13

1. Önerilen Sistemin Mantıksal Modeli--14

* Use – Case Diyagramı--14
* Nesne İlişki Şeması--14
* Veri Sözlüğü--15
* İlişkiler ve Kardinaliteler--17

3. Tasarım--17

1. Genel Tasarım17
2. Veri Tasarımı18

* Tablo Tanımları--18

1. Veri Tabanı Diyagramı--20

4.Gerçekleştirim--21

1. Yazılım Geliştirme Ortamları--21
2. Kodlama Stili--22
3. Tasarlanan Sistemin Kodlanması--23
4. Kod Gözden Geçirme--24

4.Artımsal Geliştirme Süreç Modeline Göre Projenin Gerçekleştirilmesi

5. Yazılım Doğrulama ve Geçerleme--25

1. Sınama Kavramları--25
2. Sınama Yöntemi--26
3. Sınama ve Bütünleştirme Stratejisi--26

6. Kurulum ve Bakım--26

1. Hazırlık--26
2. Kurulum--27
3. Kullanıcı Eğitimi--27
4. Bakım ve Destek--27

**1. Projenin Tanımı**

**1.1 Projenin Amacı**

Bu projenin amacı, geleneksel muhasebe yöntemlerinden dijital platformlara geçişi destekleyerek işletmelerin zaman ve kaynak tasarrufu yapmalarına yardımcı olacaktır. Ayrıca, kağıt tabanlı işlemlerin yerini alarak çevre dostu bir çözüm sunacak ve veri yönetimini daha etkili hale getirecektir. Bu uygulama, kullanıcıların gelir-gider takibi yapmalarını, faturaları düzenlemelerini, stok takibi yapmalarını ve mali raporlar oluşturmalarını sağlayacak çeşitli özellikler içerecektir.

**1.2 Projenin Sahip Olduğu Nitelikler**

* **Kullanıcı Dostu Arayüz:** Kullanıcıların kolayca erişebileceği ve kullanabileceği bir arayüz tasarlanacaktır. İşletme sahipleri ve çalışanları, uygulamayı rahatlıkla kullanarak muhasebe işlemlerini gerçekleştirebileceklerdir.
* **Muhasebe Modülü:** Gelir ve giderlerin kaydedilmesini sağlayacak kapsamlı bir muhasebe modülü oluşturulacaktır. Kullanıcılar, gelir ve giderlerini kolayca kaydedebilecek, hesaplarını yönetebilecek ve mali durumlarını takip edebileceklerdir.
* **Fatura Düzenleme ve Takip:** Kullanıcılar, satış faturaları, alım faturaları ve diğer belgeleri düzenlemek ve takip etmek için özel bir modüle erişebileceklerdir. Faturaların düzenlenmesi, ödeme takibi ve tahsilatlar gibi işlemler bu modül üzerinden gerçekleştirilecektir.
* **Stok Takibi ve Envanter Yönetimi:** İşletmeler, stoklarını yönetmek için bir envanter modülü kullanabileceklerdir. Stok giriş-çıkışı, stok seviyelerinin takibi ve stok değerlemesi gibi işlemler bu modül üzerinden gerçekleştirilecektir.
* **Mali Raporlar:** Uygulama, kullanıcılara çeşitli mali raporlar oluşturma ve yönetme imkanı sunacaktır. Gelir-gider tabloları, bilançolar, nakit akış tabloları ve diğer raporlar, kullanıcıların işletmelerinin mali durumunu anlamalarına yardımcı olacaktır.
* **Güvenlik:** Uygulama, kullanıcı verilerinin güvenliğini sağlamak için gerekli güvenlik önlemlerini alacaktır. Veri şifreleme, kullanıcı yetkilendirme ve güvenli bağlantılar gibi güvenlik önlemleri uygulanacaktır.

**2. Yazılım Geliştirme Yaşam Döngüsü**

Yazılım Geliştirme Yaşam Döngüsü , bir yazılım projesinin başlangıcından sonuna kadar olan süreci ifade eder. Bu süreç, yazılımın ihtiyaç analizinden, tasarımına, kodlamasına, test edilmesine, dağıtımına ve bakımına kadar bir dizi aşamayı içerir. Yazılım Geliştirme Yaşam Döngüsü yazılım geliştirme sürecinin yönetilmesi ve organize edilmesi için bir çerçeve sağlar. Her aşama, projenin başarılı bir şekilde tamamlanması için önemlidir ve aşamalar genellikle birbirini takip eden ve önceki aşamalardan elde edilen çıktılar üzerine kuruludur. Bu sayede, yazılım projeleri sistematik bir şekilde yönetilebilir ve başarıyla tamamlanabilir.

1. **Planlama:** Projenin başlangıcında gerçekleşir ve projenin amaçları, kapsamı, kaynakları ve zaman çizelgesi gibi faktörlerin belirlenmesini içerir. Projenin genel stratejisi ve yönetim planı bu aşamada oluşturulur.
2. **Sistem Çözümleme:** Projenin gereksinimlerinin belirlendiği ve analiz edildiği aşamadır. Bu aşamada, kullanıcı ihtiyaçları ve iş süreçleri incelenir, gereksinimler belirlenir ve belgelendirilir. Sistem çözümleme, projenin gereksinimlerini anlamak ve proje kapsamını netleştirmek için önemlidir.
3. **Tasarım:** Tasarım aşaması, projenin detaylı olarak planlanması ve tasarlanması sürecidir. Bu aşamada, sistem mimarisi belirlenir, kullanıcı arayüzü tasarlanır, veri tabanı yapısı oluşturulur ve sistem bileşenleri tasarlanır. Tasarım aşaması, projenin teknik detaylarının belirlenmesi ve projenin fiziksel yapısının oluşturulması için önemlidir.
4. **Gerçekleştirme:** Gerçekleştirme aşaması, tasarımın kodlama ve yazılım geliştirme sürecidir. Bu aşamada, tasarlanan sistem bileşenleri kodlanır, yazılım modülleri oluşturulur ve entegre edilir. Yazılımın gerçekleştirilmesi, tasarımın hayata geçirilmesi ve yazılımın geliştirilmesi için kritik bir adımdır.
5. **Yazılım Doğrulama ve Geçerleme:** Yazılım doğrulama ve geçerleme aşaması, yazılımın test edilmesi ve doğrulanması sürecidir. Bu aşamada, yazılımın işlevselliği, performansı, güvenilirliği ve güvenliği test edilir. Hatalar ve kusurlar bulunur, düzeltilir ve yazılımın kalitesi artırılır.
6. **Kurulum ve Bakım:** Kurulum ve bakım aşaması, yazılımın kullanıma sunulması ve düzenli bakımının yapılması sürecidir. Bu aşamada, yazılım son kullanıcılar için dağıtılır, kurulur ve yapılandırılır. Yazılımın dağıtımından sonra, düzenli bakım yapılır ve hatalar düzeltilir, performans iyileştirmeleri yapılır ve yazılımın güncelliği sağlanır.

Projemizde bu adımları gerçekleyecek bir süreç modeli belirlemeliyiz. Projeyi modüllere böleceğimiz gerçeği göz önüne alınınca Artırımsal Süreç Modelini kullanacağız

**3. Artımsal Geliştirme Süreç Modeli**

Artırımsal model, yazılım geliştirme sürecini küçük artırımlar veya döngüler halinde ilerleyen bir yaklaşıma dayanır. Proje süresi, belirli bir zaman dilimine bölünür ve her bir zaman dilimi bir artırım veya döngü olarak adlandırılır. Her bir döngü, tasarım, kodlama, test ve entegrasyon gibi yazılım geliştirme süreçlerini içerir.

Yazılım geliştirme sürecinin her bir döngüsü, projeye yeni bir işlevsellik eklemek için kullanılır. Döngünün sonunda, planlanmış çıktılar elde edilir ve yazılımın yeni bir versiyonu tamamlanır. Bu süreç, yazılımın artırımsal olarak gelişmesini sağlar, yani her döngüde yazılımın işlevselliği biraz daha genişler.

Artırımsal modelin esnekliği, projenin farklı aşamalarının örtüşebilir olmasına izin verir. Bir döngü hala tamamlanmamışken, diğer bir döngünün tasarım veya kodlama süreci başlayabilir. Bu, projenin daha hızlı ilerlemesini sağlar ve zaman kazandırır.

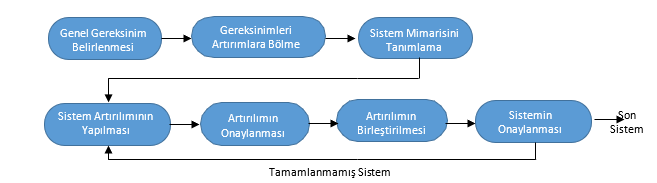
Her döngüde yeni bilgi ve tecrübeler edinilir. Bu bilgi ve tecrübeler, projenin ilerleyen aşamalarında değerli katkılar sağlar ve yazılımın kalitesini artırır.

Artırımsal modelin en büyük avantajlarından biri, projenin erken aşamalarında elde edilen çıktıların esnekliğidir. Projede değişiklikler yapılması gerektiğinde, bu değişikliklerin maliyeti ve etkisi daha az olabilir. Bu, projenin daha adapte ve müşteri odaklı olmasını sağlar.

Artırımsal model, özellikle gereksinimlerin başlangıçta tam olarak belirlenemediği veya değişebileceği projeler için uygundur. Yazılım projeleri, parçalara bölünmeye uygun olmalı ve her bir parça ayrı olarak ele alınmalıdır. Bu model, yazılım geliştirmenin doğasına daha uygun bir yaklaşım olarak görünmektedir ve başarıyla uygulanabilir.

metin, ekran görüntüsü, yazı tipi, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

****

**4. Artımsal Geliştirme Süreç Modeline Göre Projenin Gerçekleştirilmesi**

**4.1 Planlama**

**4.1.1 Proje Kapsamının Belirlenmesi**

Proje kapsamının belirlenmesi, projenin neyi hedeflediğini ve neyi hedeflemediğini netleştirmeyi içerir. Bu adım, projenin hedefleri, işlevleri ve sınırlarıyla ilgili bir anlayış oluşturmak için yapılır. Proje kapsam belirleme sürecinde, paydaşlarla yapılan görüşmeler ve gereksinim analizi gibi yöntemler kullanılabilir.

**4.1.2 Proje Kaynaklarının Belirlenmesi**

**1.Proje Ekibi**

Proje ekibi hem yazılım geliştirme sürecini hem de projenin iş gereksinimlerini etkili bir şekilde yönetebilir. Yazılım geliştiriciler, uygulamanın teknik yönlerini yönetirken, proje yöneticileri proje sürecini takip eder ve ekip üyeleri arasında koordinasyon sağlar. İş geliştirme yöneticisi ve iş analisti ise, iş gereksinimlerini analiz eder, kullanıcı ihtiyaçlarını belirler ve uygulamanın işlevselliğini tasarlar. Bu dengeli bir ekip yapısı, projenin başarılı bir şekilde tamamlanması ve hedeflenen sonuçların elde edilmesi için önemlidir.

* **Proje Yöneticisi :** Projenin genel yönetiminden sorumlu olan kişidir. Projede yer alan tüm birimlerin görevlerini belirler ve projenin her aşamasında denetleyici olarak görev yapar.
* **İş Analisti :** Proje sürecinin belirlenmesinden sorumludur. Kullanılacak sistemler ve modüller hakkında fizibilite çalışmaları yapar ve var olan sorunlara çözüm getirir ya da sistemleri geliştirir.
* **Sistem Tasarımcı :** Projenin genel sistem mimarisini oluşturur. İş analistiyle birlikte çalışarak projenin gereksinimlerini karşılayan bir sistem yapısı oluşturur.
* **Yazılım Geliştiriciler :** Projenin kodlama işlemlerinden sorumludur. Veri tabanı yöneticisi ve web tasarımcısıyla iş birliği içinde çalışarak uygulamanın geliştirilmesini sağlarlar.
* **Veri Tabanı Yöneticisi :** Veri tabanı mimarisini hazırlar ve tablolar arasındaki ilişkileri belirler. Stored Prosedürler ve trigger tasarımları gibi veri tabanı bileşenlerini yönetir.
* **Web Tasarımcı :** Projenin web arayüzünü tasarlar ve web ortamını hazırlar. Web kodlamasından sorumludur ve yazılım geliştiricilerle iş birliği yaparak projenin web bileşenlerini entegre eder.
* **Yazılım Destek Elemanı :** Projenin uygulanması ve işletilmesi sürecinde teknik destek sağlar. Kullanıcıların karşılaştığı sorunları çözer ve sistemlerin düzgün çalışmasını sağlar.

**2. Donanım ve Yazılım Kaynakları**

* Ana Bilgisayarlar: Geliştirme ve test amacıyla kullanılacak ana bilgisayarlar.
* Sunucular: Web sunucuları, veri tabanı sunucuları ve e-posta sunucuları gibi çeşitli sunucu kaynakları.
* Kullanıcı Bilgisayarları: Kullanıcıların uygulamaya erişmek için kullanacakları bilgisayarlar.
* Yerel Alan Ağı (LAN) Altyapısı: İş yerindeki bilgisayarların ve sunucuların birbirine bağlanmasını sağlayan yerel ağ altyapısı.
* Geniş Alan Ağı (WAN) Altyapısı: Farklı lokasyonlardaki kullanıcıların uygulamaya erişimini sağlayan geniş alan ağı altyapısı
* Proje Geliştirme Ortamı: Visual Studio Code, JetBrains IntelliJ IDEA
* Programlama Dili: Python, JavaScript
* Veri Tabanı Yönetim Sistemi: Oracle RDBMS, MySQL
* Arayüz Tasarım Araçları: Adobe XD, Figma , Sketch
* Destek Aracı: Git, Github , Microsoft Teams

**3. Proje Maliyetinin Belirlenmesi**

Projenin zaman ve bütçe planlaması açısından kullanılan maliyet kestirimi yöntemlerinden Etkin Maliyet Yöntemi (COCOMO) kullanılmıştır.

Bu modelde ilk olarak yazılımın ayrıntı düzeyine göre uygun model seçilmiştir. Ayrıntı düzeyine göre modeller;

• Temel model,

• Ara Model,

• Ayrıntılı Model

olarak 3’ e ayrılmıştır. Bu projede kullanılmak üzere Ara Model seçilmiştir ve bu proje ayrık proje sınıfındadır.

İş Gücü (K) = 3,2\* S1,05

Zaman (T) = 2,5\* Kd0,38

Düzeltilmiş İş Gücü (Kd)= K\*C

S -> 1000 türünden satır sayısıdır.

C -> Maliyet çarpanıdır ve 15 maliyet etmeninin çarpılması sonucu elde edilir.

C = C1 \* C2 \* C3 \* …. \* C15

**Problemin bilgi ortamının incelenmesi**

**Kullanıcı Girdileri:** Yazılıma verilen her farklı uygulama bileşeni.

**Kullanıcı Çıktıları:** Kullanıcıyı ilgilendiren her türlü mantıksal çıktı. (Örnek: Raporlar, ekran çıktıları, hata iletileri)

**Kullanıcı Sorguları:** Çevrim içi olarak bilgisayara verilen girdi sonucu alınan çıktılar.

**Kütükler:** Her mantıksal bilgi yığını ya da kütük.

**Dışsal Arayüzler:** Geliştirilecek bilgi sisteminin, kurum içinde veya dışında bir başka bilgi sistemiyle bilgisayar ortamında iletişimi.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Ölçüm Parametresi** | **Sayı** |  | **Ağırlık Faktörü** | | |  |  |
| Yalın | Ortalama | Karmaşık |
| Kullanıcı Girdi Sayısı | 12 | x | 3 | **4** | 6 | = | 48 |
| Kullanıcı Çıktı Sayısı | 15 | x | 4 | 5 | **7** | = | 105 |
| Kullanıcı Sorgu Sayısı | 15 | x | 3 | 4 | **6** | = | 90 |
| Kütük Sayısı | 6 | x | 7 | **10** | 15 | = | 60 |
| Dışsal Arayüz Sayısı | 2 | x | **5** | 7 | 10 | = | 10 |
| **Toplam Sayı** | | | | | | = | 313 |

Buna göre AİN (Ayarlanmamış İşlev Nokta sayısı) = 313 olarak hesaplanmıştır.

**Problemin teknik karmaşıklığının incelenmesi**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TFK Soruları** | **Hiç etkisi yok(0)** | **Çok Az etkisi var(1)** | **Etkisi var(2)** | **Ortalama etkisi var(3)** | **Önemli etkisi var(4)** | **Mutlaka olmalı, kaçınılmaz(5)** |
| Uygulama, güvenilir yedekleme ve kurtarma gerektiriyor mu? |  |  |  |  |  | ✓ |
| Veri iletişimi gerekiyor mu? |  |  |  |  |  | ✓ |
| Dağıtık işlem işlevleri var mı? |  |  |  |  | ✓ |  |
| Performans kritik mi? |  |  |  | ✓ |  |  |
| Sistem mevcut ve ağır yükü olan bir işletim ortamında mı çalışacak? |  |  | ✓ |  |  |  |
| Sistem, çevrim içi veri girişi gerektiriyor mu? |  |  |  |  |  | ✓ |
| Ana kütükler çevrim-içi olarak mı güncelleniyor? |  |  |  |  |  | ✓ |
| Girdiler, çıktılar, kütükler ya da sorgular karmaşık mı? |  |  |  | ✓ |  |  |
| İçsel işlemler karmaşık mı? |  |  |  | ✓ |  |  |
| Tasarlanacak kod, yeniden kullanılabilir mi olacak? |  |  |  | ✓ |  |  |
| Dönüştürme ve kurulum tasarımda dikkate alınacak mı? |  |  | ✓ |  |  |  |
| Sistem birden çok yerde yerleşik farklı kurumlar için mi geliştiriliyor? |  |  |  | ✓ |  |  |
| Tasarlanan uygulama, kolay kullanılabilir ve kullanıcı tarafından kolayca değiştirilebilir mi olacak? |  |  | ✓ |  |  |  |

Buna göre TFK (Teknik Karmaşıklık Faktörü) = 45 olarak hesaplanmıştır.

**İşlev Noktası Hesaplama**

Bu son adımda, bilinen ve sık kullanılan deneysel formül kanalıyla ilk iki adım sonucundaki bulgular kullanılarak işlev nokta sayısı belirlenir.

Gerek AİN, gerekse TKF önceki bölümlerde açıklandığı gibi hesaplandıktan sonra, geliştirilecek bilgi sistemine ilişkin İşlev Nokta Sayısı (İN) aşağıdaki formül kullanılarak hesaplanır.

İN(İşlev Nokta sayısı) = AİN x (0.65x0.01xTKF)

Üretkenlik = İN/Kişi-Ay

Kalite = Hatalar/İN

Maliyet = Fiyat/FP

İN= 313\*( 0.65\*0.01\*45)

İN= 91.55

İN ≈ 92 olarak hesaplanmıştır.

Buna göre;

|  |  |
| --- | --- |
| **Programlama Platformu** | **Satır Sayısı/İN(Ortalama)** |
| Assembly Dili | 300 |
| COBOL | 100 |
| FORTRAN | 100 |
| Pascal | 90 |
| C | 90 |
| Ada | 70 |
| Nesne Kökenli Diller | 30 |
| 4. Kuşak Dilleri | 20 |
| Kod Üreticiler | 15 |

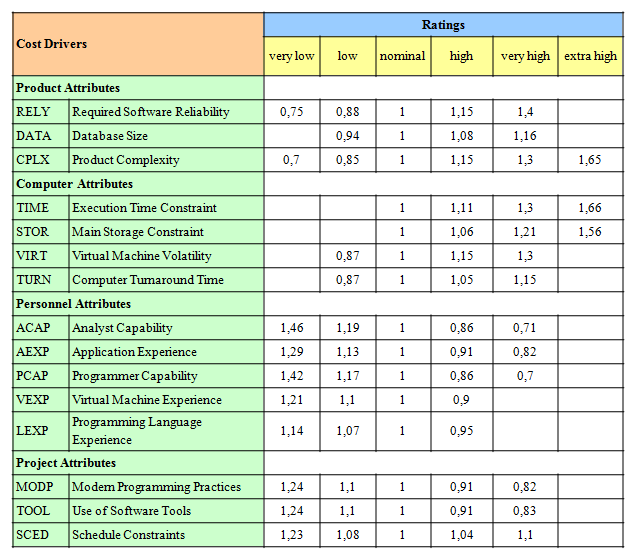
Kullanılan programlama platformu nesne kökenli olduğu için yaklaşık satır sayısı kestirimi;

Satır Sayısı = 92\*30 = 2760

S sayısı formülde yerine konularak K, C, Kd, T değerleri hesaplanacaktır ;

İş Gücü (K) = 3,2\* (2,76)1,05

K=9,29 Kişi-ay



Ürün Özellikleri

RELY: Yazılımın güvenirliği.

DATA: Veritabanının büyüklüğü.

CPLX: Karmaşıklığı.

Bilgisayar Özellikleri

TIME: İşletim zamanı kısıtı.

STOR: Ana bellek kısıtı

VIRT: Bilgisayar platform değişim olasılığı. Örn; bellek ve disk kapasitesi artırımı, CPU upgrade…

TURN: Bilgisayar iş geri dönüş zamanı. Örn; hata düzeltme süresi.

Personel Özellikleri

ACAP: Analist yeteneği. Deneyim, birlikte çalışabilirlik.

AEXP: Uygulama deneyimi. Proje ekibinin ortalama tecrübesi.

PCAP: Programcı yeteneği.

VEXP: Bilgisayar platformu deneyimi. Proje ekibinin geliştirilecek platformu tanıma oranı.

LEXP: Programlama dili deneyimi.

Proje Özellikleri

MODP: Modern programlama teknikleri. Örn; Yapısal programlama, görsel programlama, yeniden kullanılabilirlik.

TOOL: Yazılım geliştirme araçları kullanımı. Örn; CASE araçları, metin düzenleyiciler, ortam yönetim araçları

SCED: Zaman kısıtı.

C=1,15\*1,08\*1\*1,3\*1,21\*1,15\*1\*1\*0,86\*0,91\*0,86\*1\*1\*1,1\*1\*1

C= 1,66

Düzeltilmiş İş Gücü (Kd)= K\*C

Kd=9,29\*1,66

Kd=15,42

Zaman (T) = 2,5\* Kd0,38

T=2,5\*(15,42)0,38

T=7,06 ≈ 7 ay

Projenin aylık gideri 15.000 dolar olduğu varsayılarak, toplam proje maliyeti 105.000 dolar olarak hesaplanmıştır

**4.1.3 Proje İş – Zaman Planı oluşturulması**

Planlama Aşaması (3 hafta)

* Proje kapsamının belirlenmesi
* Proje kaynaklarının belirlenmesi
* Proje ekip yapısının oluşturulması
* Proje maliyetlerinin kestirilmesi
* Ayrıntılı proje planının yapılması

Sistem Çözümleme Aşaması (6 hafta)

* Gerekli olan gereksinimlerin ayrıntılı belirlenmesi
* Mevcut sistemin çözümlenmesi
* Kullanıcı arayüz modelinin oluşturulması

Tasarım Aşaması (4 hafta)

* Veri tasarımın, Yapısal tasarımın (genel tasarım, ayrıntı tasarım-süreç işlem tasarımı)
* Kullanıcı arayüz tasarımının oluşturulması
* Tasarım çalışmasının değerlendirilmesi
* Tasarım raporunun hazırlanması

Gerçekleştirim Aşaması (4 hafta)

* Yazılım geliştirme ortamlarının belirlenmesi
* Kod stilinin belirlenmesi
* Yazılım kodlamasının gerçekleştirilmesi
* Olağan dışı durumların çözümlenmesi
* Yazılan kodun gözden geçirilmesi

Yazılım Doğrulama ve Gerçekleme Aşaması (3 hafta)

* Birim sınama, alt sistem sınama, sistem sınama, kabul sınama işlemlerinin yapılması
* Sonuç raporlarının oluşturulması

Kurulum-Eğitim-Bakım Aşaması (8 hafta)

* Sistem yazılımlarının kurulması
* Veri tabanının kurulması
* Eğitim verilmesi
* Yerinde destek organizasyonu çalışmaları
* Düzeltici bakımın yapılması
* Uyarlayıcı bakımın yapılması

**4.2 Sistem Çözümleme**

**4.2.1 Gereksinimlerin Belirlenmesi**

**4.2.1.1 Fiziksel Ortam**

* Proje, uygulama ve veri tabanı sunucusu hosting firmalarından sağlanacaktır.
* Sunucu, sistemin kullanılacağı yerde bulunmayacaktır.
* Sisteme internet aracılığıyla uzaktan erişilecektir. Kullanıcılar, internet bağlantısı olan herhangi bir cihazdan sisteme erişebilecektir.
* Sistem elemanları herhangi bir fiziksel etkenden etkilenmeyecek şekilde tasarlanacaktır. Sıcaklık, nem ve manyetik etkiler sistem performansını olumsuz yönde etkilemeyecektir. Bu nedenle, sunucu ve diğer sistem bileşenleri, fiziksel ortamdan bağımsız bir şekilde çalışabilecek yapıda olacaktır.

**4.2.1.2 Arayüz**

* Arayüzler, kullanıcıların sistemi kolayca kullanabilmesi için belirlenen standartlara uygun olarak basit ve anlaşılır bir şekilde tasarlanacaktır.
* Kullanıcıların işlemleri hızlıca gerçekleştirebilmeleri için menüler, düğmeler ve formlar gibi arayüz öğeleri kullanıcı dostu bir şekilde yerleştirilecektir.
* İşlemler adım adım ve mantıklı bir sıra ile sunulacak, kullanıcıya gereksiz detaylardan kaçınılacak ve gereksinim duyulan bilgilere kolayca erişim sağlanacaktır.
* Muhasebe işlemleri; fatura takibi, gelir-gider yönetimi, cari hesaplar gibi finansal işlemler arayüzde kolayca anlaşılır ve erişilir şekilde tasarlanacak.

**4.2.1.3 Kullanıcılar**

* Sistemde birkaç çeşit yönetici bulunacaktır. Bunlar yazılım geliştiriciler, veri tabanı yöneticisi gibi üst düzey sistem yöneticileri ve sistem kullanıcıları .
* Üst düzey yöneticiler sistemi yönetimini ve bakımını yapabilir. Kullanıcılar ise sistemde tanımlanmış işlemleri kullanabilirler.

**4.2.1.4 İşlevsellik**

* Sistem fatura takip, e fatura, cari hesap, gelir-gider takibi, stok takibi, özet rapor ve nakit yönetimi gibi bölümler içerecektir.
* Sistem internete bağlı her kullanıcı sistemi için çalışır durumdadır.
* Sistemde tutulan her bilgi istenilmesi halinde fiziksel çıktılar alınabilecektir.
* Sistemin çalışmasında istenilen herhangi bir hız bulunmamaktadır. Ancak sistem gerçekleştirilirken hız unsuru da göz önüne alınarak kullanıcıyı yoracak yavaşlıklar önlenecektir.

**4.2.1.5 Veri ve Güvenlik**

* Sistem, kullanımı sırasında sürekli olarak veri kaydetme veya veri görüntüleme işlemleri yapar.
* Sistemdeki bilgi erişim denetimi yapılmaktadır. Her kullanıcı her bilgiye erişme yetkisine sahip değildir. Sadece yöneticiler her kullanıcı bilgilerine erişebilir.
* Yedek kopyalar sistemden ayrı bir yerde saklanacaktır.

**4.2.1.6 Kalite ve Kaynak**

* Sistemsel sorunlarda Yazılım destek elemanı çözümleyecektir.
* Sisteme yapılan bakım sistemde kaynaklanan hatalardan dolayı ve sistemin geliştirmesine yönelik olacaktır.
* Yazılım ek olarak herhangi bir fiziksel yer kaplamayacaktır. Yazılımın ana hatlarının depolanması çok küçük boyutlarda yazılım dosyalarıyla web sunucusunda depolanacaktır.

**4.2.2 Sistem Çözümleme Çalışması**

**4.2.2.1 Mevcut Sistemin İncelenmesi ve Değerlendirilmesi**

Günümüzde birçok işletme, muhasebe işlemlerini manuel olarak yürütmekte ve bu işlemler için ekstra bir muhasebe personeline ihtiyaç duymaktadır. Bu durum, işletmelerin maliyetlerini artırmanın yanı sıra, işlemlerin etkinliğini de azaltmaktadır. Manuel işlemler, kağıt tabanlı belgeleme ve defter tutma gibi geleneksel yöntemlerle gerçekleştirilmektedir. Ancak, bu yöntemler hatalara ve karmaşıklığa yol açmakta ve işletmenin zamanını ve kaynaklarını boşa harcamaktadır. Bu sorunlar, işletmenin verimliliğini ve rekabet gücünü olumsuz yönde etkiler. Bu nedenle, geleneksel muhasebe yöntemlerinin dijitalleştirilmesi ve otomatikleştirilmesi gerekmektedir. Dijitalleşmiş sistem, işletmenin muhasebe işlemlerini daha hızlı, daha doğru ve daha verimli bir şekilde yürütmesine olanak tanır.

**4.2.3 Önerilen Sistemin Mantıksal Modeli**

**4.2.3.1 Use – Case Diyagramı**

diyagram, taslak, çizgi, beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

**4.2.3.2** **Nesne İlişki Şeması**

metin, ekran görüntüsü, sayı, numara, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu

**4.2.3.3 Veri Sözlüğü**

Nesne ilişki şemalarında belirtilen nesne özelliklerinin ayrıntılı tanımları aşağıda belirtildiği şekildedir;

Yönetici (Admin):

* Yönetici ID: Tüm yöneticiler için benzersiz tanımlayıcı.
* İsim: Yöneticinin adı.
* Eposta: Yöneticinin e-posta adresi.
* Şifre: Yöneticinin hesabına erişmek için kullanacağı şifre.
* Yetki Seviyesi: Yöneticinin yetki seviyesi (örneğin, süper yönetici, normal yönetici).

Kullanıcı (User):

* Kullanıcı ID: Tüm kullanıcılar için benzersiz tanımlayıcı.
* İsim: Kullanıcının adı.
* Eposta: Kullanıcının e-posta adresi.
* Şifre: Kullanıcının hesabına erişmek için kullanacağı şifre.
* Rol: Kullanıcının rolü (örneğin, işletme sahibi, çalışan).

Fatura (Invoice):

* Fatura ID: Fatura kayıtları için benzersiz tanımlayıcı.
* Kullanıcı ID: Fatura kaydını oluşturan kullanıcıyı belirlemek için kullanılır.
* Fatura Türü: Faturanın türü (örneğin, satış, alım).
* Tarih: Faturanın oluşturulduğu tarih.
* Tutar: Fatura miktarı.
* Müşteri Tedarikçi İsmi: Fatura ile ilgili müşteri veya tedarikçi ismi.
* Durum: Faturanın durumu (örneğin, ödenmiş, ödenmemiş).
* Açıklama: Fatura ile ilgili açıklama.

Gelir (Income):

* Gelir ID: Gelir kayıtları için benzersiz tanımlayıcı.
* Kullanıcı ID: Gelir kaydını oluşturan kullanıcıyı belirlemek için kullanılır.
* Tarih: Gelir kaydının oluşturulduğu tarih.
* Tutar: Gelir miktarı.
* Gelir Türü: Gelirin türü (örneğin, satış, hizmet).
* Açıklama: Gelirle ilgili açıklama.

Gider (Expense):

* Gider ID: Gider kayıtları için benzersiz tanımlayıcı.
* Kullanıcı ID: Gider kaydını oluşturan kullanıcıyı belirlemek için kullanılır.
* Tarih: Gider kaydının oluşturulduğu tarih.
* Tutar: Gider miktarı.
* Gider Türü: Giderin türü (örneğin, malzeme, hizmet).
* Açıklama: Giderle ilgili açıklama.

Stok (Stock):

* Stok ID: Stok kayıtları için benzersiz tanımlayıcı.
* Ürün İsmi: Stokta bulunan ürünün ismi.
* Miktar: Stok miktarı.
* Birim Fiyat: Ürünün birim fiyatı.
* Kategori: Ürünün kategorisi.
* Son Güncelleme Tarihi: Stok kaydının son güncelleme tarihi.

Mali Rapor (Financial Report):

* Rapor ID: Mali raporlar için benzersiz tanımlayıcı.
* Kullanıcı ID: Raporu oluşturan kullanıcıyı belirlemek için kullanılır.
* Rapor Türü: Raporun türü (örneğin, gelir-gider tablosu, bilanço).
* Oluşturma Tarihi: Raporun oluşturulma tarihi.

**4.2.3.3 İlişkiler ve Kardinaliteler**

Yönetici - Kullanıcı: Bir yönetici birden fazla kullanıcıyı yönetebilir.

Kullanıcı - Gelir: Bir kullanıcı birden fazla gelir kaydına sahip olabilir.

Kullanıcı - Gider: Bir kullanıcı birden fazla gider kaydına sahip olabilir.

Kullanıcı - Fatura: Bir kullanıcı birden fazla fatura kaydına sahip olabilir.

Kullanıcı - Stok: Bir kullanıcı birden fazla stok kaydını yönetebilir.

Kullanıcı - Mali Rapor: Bir kullanıcı birden fazla mali rapor oluşturabilir.

**4.3 Tasarım**

Hayata geçirilmesi planlanan web tabanlı muhasebe uygulamasının tasarım aşamasında, sistem çözümleme aşamasında üretilen mantıksal model fiziksel modele dönüştürülmüştür. Bu süreçte, uygulamanın hangi bileşenlerden oluşacağı, bu bileşenler arasındaki ilişkilerin nasıl olacağı ve her bir bileşenin iç yapısının ayrıntıları belirlenmiştir. Ayrıca, sistemin işleyişi ve performansını optimize edecek teknik detaylar da tasarım aşamasında netleştirilmiştir.

**4.3.1 Genel Tasarım**

Web tabanlı muhasebe uygulamasına, internet üzerinden bir domain aracılığı ile erişilecektir. Sisteme aynı anda birden fazla kullanıcı erişebilir ve işlem yapabilir. Kullanıcılar, sisteme giriş yaptıklarında yönetici veya standart kullanıcı olarak sınıflandırılır.

Sistem, gelir gider takibi, stok yönetimi, mali raporlama ve nakit yönetimi gibi temel muhasebe işlemlerini desteklemektedir. Kullanıcılar, sistem üzerinden gelir ve gider kayıtlarını ekleyebilir, güncelleyebilir ve silebilir. Stok yönetimi modülü, ürün ekleme, stok güncelleme ve stok durumu takibi gibi işlemleri içerir. Mali raporlama özellikleri, kullanıcıların finansal durumlarını ayrıntılı raporlar halinde görmelerine olanak tanır.

Bu işlemler, kullanıcıların ihtiyaçlarını karşılayacak şekilde tasarlanmış basit ve anlaşılır arayüzler üzerinden gerçekleştirilecektir. Sistem, kullanıcı deneyimini optimize etmek için her aşamada bilgi ve rehberlik sağlayacaktır. Sisteme ait tüm özellikler ve işlevler, ayrıntılı olarak proje tanımı ve proje niteliklerinde belirtilmiştir.

**4.3.2 Veri Tasarımı**

Bu aşamada geliştirilecek olan yazılımda kullanılacak olan veri tabanı tablolarının tanımlamaları yapılmış ve tablolar arasındaki ilişkiler gösterilmiştir.

**4.3.2.1 Tablo Tanımları**

1. Yönetici (Admin) Tablosu:

Sistemi yöneten ve kullanıcı hesaplarına erişim sağlayan yöneticilerin bilgilerini içeren tablodur. Bu tablo, yöneticilerin kimlik bilgilerini ve yetki seviyelerini tutar.

|  |  |
| --- | --- |
| **Nitelik** | **Tür** |
| YöneticiID | INT( PRIMARY KEY) |
| İsim | VARCHAR(100) |
| Eposta | VARCHAR(100) |
| Şifre | VARCHAR(100) |
| YetkiSeviyesi | VARCHAR(50) |

2.Kullanıcılar Tablosu:

Yazılımı kullanacak olan kullanıcıların sisteme kayıt olurken kendileri tarafından girilen kullanıcı adı, şifre, e-mail ve rol bilgilerini içeren tablodur. Ayrıca, kullanıcıların hangi yönetici tarafından yönetildiğini de belirtir.

|  |  |
| --- | --- |
| **Nitelik** | **Tür** |
| KullanıcıID | INT (PRIMARY KEY) |
| İsim | VARCHAR(100) |
| Eposta | VARCHAR(100) |
| Şifre | VARCHAR(100) |
| Rol | VARCHAR(100) |
| YöneticiID | INT( FOREIGN KEY) |
|  | REFERENCES (YöneticiID) |

3.Gelirler Tablosu:

Kullanıcıların elde ettikleri gelirlerin kaydını tutar. Bu tabloda, gelir türü, tutarı, tarihi ve gelirle ilgili açıklama bilgileri bulunur.

|  |  |
| --- | --- |
| **Nitelik** | **Tür** |
| GelirID | INT (PRIMARY KEY) |
| Tarih | DATE |
| Tutar | DECIMAL(10, 2) |
| GelirTürü | VARCHAR(100) |
| Açıklama | TEXT |
| KullanıcıID | INT (FOREIGN KEY) |
|  | REFERENCES (KullanıcıID) |

4.Giderler Tablosu:

Kullanıcıların yaptıkları harcamaların kaydını tutar. Bu tabloda, gider türü, tutarı, tarihi ve giderle ilgili açıklama bilgileri bulunur.

|  |  |
| --- | --- |
| **Nitelik** | **Tür** |
| GiderID | INT (PRIMARY KEY) |
| Tarih | DATE |
| Tutar | DECIMAL(10, 2) |
| GelirTürü | VARCHAR(100) |
| Açıklama | TEXT |
| KullanıcıID | INT (FOREIGN KEY) |
|  | REFERENCES (KullanıcıID) |

5.Faturalar Tablosu:

Kullanıcıların oluşturdukları fatura kayıtlarını tutar. Bu tabloda, faturanın türü, tutarı, tarihi, müşteri veya tedarikçi bilgileri ve fatura durumu gibi bilgiler yer alır.

|  |  |
| --- | --- |
| **Nitelik** | **Tür** |
| FaturaID | INT (PRIMARY KEY) |
| FaturaTürü | VARCHAR(50) |
| Tarih | DATE |
| Tutar | DECIMAL(10, 2) |
| MüşteriTedarikçiİsmi | VARCHAR(100) |
| Durum | VARCHAR(50) |
| Açıklama | TEXT |
| KullanıcıID | INT (FOREIGN KEY) |
|  | REFERENCES (KullanıcıID) |

6.Stoklar Tablosu:

Kullanıcıların sahip oldukları stokların kaydını tutar. Bu tabloda, stokta bulunan ürünlerin isimleri, miktarları, birim fiyatları, kategorileri ve son güncelleme tarihleri bulunur.

|  |  |
| --- | --- |
| **Nitelik** | **Tür** |
| StokID | INT (PRIMARY KEY) |
| Ürünİsmi | VARCHAR(100) |
| Miktar | INT |
| BirimFiyat | DECIMAL(10, 2) |
| Kategori | VARCHAR(50) |
| SonGüncellemeTarihi | DATE |

7.Mali Raporlar Tablosu:

Kullanıcıların oluşturdukları mali raporların kaydını tutar. Bu tabloda, rapor türü, oluşturma tarihi ve raporu oluşturan kullanıcının bilgileri yer alır.

|  |  |
| --- | --- |
| **Nitelik** | **Tür** |
| RaporID | INT (PRIMARY KEY) |
| RaporTürü | INT |
| OluşturmaTarihi | VARCHAR(50) |
| KullanıcıID | DATE(FOREIGN KEY) |
|  | REFERENCES (KullanıcıID) |

**4.3.3 Veri Tabanı Diyagramı**

metin, diyagram, plan, teknik çizim içeren bir resim

Açıklama otomatik olarak oluşturuldu

**4.4 Gerçekleştirim**

Bu aşama, tasarım sonucu üretilen fiziksel modelin web tabanlı muhasebe uygulaması olarak işlevsel bir yazılım biçimine dönüştürülmesini içerir. İlk olarak, planlama aşamasında belirlenmiş olan geliştirme ortamları yeniden gözden geçirilerek son kararlar verilir. Uygulamanın geliştirilmesi için kullanılacak araçlar, programlama dilleri ve veri tabanı yönetim sistemleri netleştirilir.

Daha sonra, yazılım geliştirme süreci başlatılır. Bu süreçte, gelir gider takibi, stok yönetimi, mali raporlama ve nakit yönetimi gibi temel işlevler kodlanır ve entegre edilir. Geliştirme aşamasında, her modül ayrı ayrı test edilerek hataların tespit edilip düzeltilmesi sağlanır. Kodlama tamamlandığında, sistemin tüm bileşenleri bir araya getirilerek entegre testler yapılır ve yazılımın beklenen işlevleri yerine getirdiği doğrulanır.

Bu aşama, uygulamanın sorunsuz çalışmasını sağlamak için titizlikle yürütülür ve tüm geliştirme süreci boyunca sürekli geri bildirim ve iyileştirmeler yapılır. Sistemin kullanıcı dostu olmasını ve tüm işlevlerin beklenen performans ve doğrulukla çalışmasını sağlamak için gerekli tüm önlemler alınır.

**4.4.1 Yazılım Geliştirme Ortamları**

Proje kapsamında kullanılacak olan yazılım geliştirme ortamları, programlama dili, veri tabanı yönetim sistemi, kullanılan hazır program kitapçıkları ve CASE (Computer-Aided Software Engineering) araçlarından oluşmaktadır. Bu araçlar, geliştirme ve sınama süreçlerinde etkin bir şekilde kullanılacaktır. Bu yazılım geliştirme ortamları, projenin her aşamasında etkin bir şekilde kullanılarak, uygulamanın yüksek kalitede ve hatasız bir şekilde geliştirilmesini sağlayacaktır.

* **Programlama Dili:** Web tabanlı muhasebe uygulamasının geliştirilmesi için ana programlama dili olarak C# seçilmiştir. Bu dil, projenin gereksinimlerini karşılamak için uygun ve esnek bir yapıya sahiptir.
* **Veri Tabanı Yönetim Sistemi (VTYS):** Veritabanı yönetimi için MySQL kullanacağız. MySQL, geniş veri depolama kapasitesi ve hızlı erişim imkanı sunarak muhasebe verilerinin etkin bir şekilde yönetilmesini sağlayacaktır.
* **Geliştirme Ortamı:** Visual Studio, C# ile kodlama yaparken kullanılacak olan ana geliştirme ortamı olacaktır. Visual Studio, güçlü hata ayıklama özellikleri ve kullanıcı dostu arayüzü ile geliştirme sürecini kolaylaştıracaktır.
* **Hazır Program Kitapçıkları:** Projenin belirli bölümlerinde, zaman kazandıracak ve kod tekrarını azaltacak hazır program kitapçıkları (frameworks) kullanılacaktır. Örneğin, Entity Framework veri erişim katmanı için kullanılacak.
* **CASE Araçları:** Yazılım mühendisliği sürecini desteklemek için çeşitli CASE araçları kullanılacaktır. Bu araçlar, tasarım, modelleme ve belge yönetimi süreçlerinde önemli kolaylıklar sağlayacaktır.
* **Destek Araçları:** Projenin başarılı bir şekilde tamamlanması için kullanılan destek araçları arasında Microsoft Windows 11 işletim sistemi, Git, Github veya Zoom gibi iletişim araçları yer alacaktır. Bu araçlar, ekip içi iletişimi ve iş birliğini artıracaktır.

**4.4.2 Kodlama Stili**

Projenin gerçekleştirim aşamasında kullanılacak kodlama stili, yazılımın okunabilirliğini, sürdürülebilirliğini ve bakımını kolaylaştırmak amacıyla belirlenmiştir. Geliştiricilerin koda baktıkları zaman ne yapılmak istendiğini kısa sürede anlayabilmeleri yazılım geliştirme açısından önemlidir. Bu doğrultuda, yazılım projemiz camelCase kodlama stili baz alınarak yazılmıştır. Bu yaklaşım, projenin hem geliştirme aşamasında hem de bakım aşamasında daha verimli ve etkili olmasını sağlayacaktır.

* **Kod Düzeni:** Kodlar, belirli bir düzen ve formatta yazılmıştır. Bu, hem okunabilirliği artırır hem de yazılımcılar arasında tutarlılığı sağlar.
* **Anlaşılabilir İsimlendirme:** Classlar, metodlar, değişkenler ve sabitler için mantıklı ve açıklayıcı isimler verilerek kodların anlaşılabilir ve okunabilir olması sağlanmıştır.
* **Açıklama Satırları:** Sistemin kodlanması aşamasında birden fazla yazılım uzmanı görev alacağından, yazılımcıların birbirlerini anlamaları için açıklama satırları kullanmaları ve kod yazım düzenine dikkat etmeleri önemlidir. Gereken yerlerde açıklama satırları eklenmiş ve kod yazım düzenine dikkat edilmiştir.
* **Bakım Kolaylığı:** Programın daha sonra yapılacak olan bakım aşaması için de kodlama stilinin önemi büyüktür. Kodlama stili, gelecekteki değişikliklerin ve güncellemelerin daha kolay yapılmasını sağlar.

**4.4.3 Tasarlanan Sistemin Kodlanması**

Seçilmiş olan geliştirme ortamları kullanılarak tasarım sonrası oluşturulan fiziksel modelin bilgisayar üzerinde çalışan yazılıma dönüştürülmesi için gerekli kodlamalar yapıldı.

Bu kodlardan bazıları;

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

**4.4.4 Kod Gözden Geçirme**

Yapılmış olan kodlama çalışması sonucu kod gözden geçirme işlemi yapılmıştır. Bu sayede kodun kalitesi artmış, olası hatalar yazılım teslim edilmeden önce ortaya çıkarılmış ve düzeltilmiştir. Ayrıca, proje geliştiricileri arasında bilgi birikimi ve deneyim paylaşımı sağlanmıştır. Kod gözden geçirme işlemi, kodun anlaşılabilirliğini artırarak bakım ve güncellemelerin daha kolay yapılmasını sağlamaktadır.

Gerçekleştirilecek olan sistem sınaması ve kabul sınamasının ardından, sistem kullanıma sunulacak ve kullanıcıların hizmetine açılacaktır. Bu süreç, sistemin güvenilirliğini ve performansını artıracak, son kullanıcıya sorunsuz bir deneyim sunulmasını sağlayacaktır.

**4.5 Yazılım Doğrulama ve Geçerleme**

Geliştirilecek web tabanlı muhasebe uygulamasının doğrulanması ve geçerlemesi işlemi, üretim ve geliştirme süreci boyunca devam eden etkinliklerden oluşur. Bu etkinlikler:

* Her bir değişiklik sonunda alınan çıktıların tam, doğru, açık ve tutarlı olduğunun doğrulanması,
* Her değişiklik sonrası ürünün teknik yeterliliğinin değerlendirilmesi ve uygun çözüm elde edilene kadar aktivitelerin tekrarlanması,
* Geliştirilen belirtimlerin önceki belirtimlerle karşılaştırılması,
* Yazılım ürünlerinin tüm uygulanabilir gereksinimlerinin sağlandığının sınamalarla doğrulanmasıdır.

**4.5.1 Sınama Kavramları ve Yöntemleri**

Sınama süreci, yazılımın doğruluğunu ve güvenilirliğini sağlamak için planlı bir strateji içinde gerçekleştirilir. Projemizde sınamalar dört adımda yapılmıştır:

* Birim Sınaması: Bu aşamada, diğer sistem unsurlarından bağımsız olarak birimlerin doğru çalışıp çalışmadığı test edilir. Bu testler, her bir birimin işlevselliğini yalnızca kendi içinde değerlendirir.
* Alt-Sistem Sınaması: Alt sistemler, modüllerin birleştirilmesiyle oluşturulur ve bağımsız olarak test edilir. Bu aşamada özellikle arayüz hataları üzerinde durulur ve bu hataların sınaması yoğunlaştırılır.
* Sistem Sınaması: Üst düzey bileşenlerin bir araya gelerek sistemin tamamında nasıl çalıştığının test edildiği aşamadır. Burada bileşenler arası etkileşimde ortaya çıkabilecek hatalar aranır.
* Kabul Sınaması: Sistemin son test aşamasıdır ve gerçek veriler kullanılarak gerçekleştirilir. Bu aşamada sistemin kullanıma hazır olduğunun son doğrulaması yapılır.

metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şeklinde sınama işlemleri aşağıdan yukarıya doğru devam eder.

**4.5.2 Sınama Yöntemleri**

İki farklı sınama yöntemi bulunmaktadır. Bunlar;

* Kara Kutu Testi (Black Box)
* Beyaz Kutu Testi (White Box)

Projemizde Beyaz Kutu Testi yöntemini kullanarak sınamalar gerçekleştirdik. Beyaz Kutu Testi yöntemi şu özelliklere sahiptir:

* **Bütün Bağımsız Yollar ve Bileşenler:** Her bir bağımsız yol ve bileşen en az bir kez test edilmiştir.
* **Mantıksal Karar Noktaları:** Tüm mantıksal karar noktaları iki farklı karar için test edilmiştir.
* **Döngülerin Sınır Değerleri:** Bütün döngülerin sınır değerleri test edilmiştir.
* **İç Veri Yapıları:** Proje içerisinde kullanılan iç veri yapıları da detaylı bir şekilde test edilmiştir.

Bu yöntemlerin kullanılmasıyla projenin yazılım kalitesi artırılmış, hataların erken tespiti sağlanmış ve sistemin güvenilirliği sağlanmıştır.

**4.5.3 Sınama ve Bütünleştirme Stratejisi**

Web tabanlı muhasebe uygulamasının bütünleştirilmesi ve sınanmasında aşağıdan yukarıya sınama ve bütünleştirme stratejisi izlenmiştir. Bu stratejiye göre, ilk olarak uygulamanın temel işlevlerini sağlayan bileşenler bağımsız olarak sınanmıştır. Daha sonra, bu bileşenler bir araya getirilerek entegrasyon testleri gerçekleştirilmiştir. Bu süreç, uygulamanın her aşamasında olası hataların erken tespit edilmesini ve düzeltilmesini sağlamıştır. Bu strateji, uygulamanın her aşamasında kaliteyi artırmayı ve son kullanıcıya güvenilir bir ürün sunmayı amaçlamaktadır.

Sınama işlemleri sonucunda sınama belirtimleri, sınamayı yapan kişi, sınama tarihi, sınama yapılan yer ve açıklamalar türündeki bilgiler eklenerek sınama raporları oluşturulmuştur.

**4.6 Kurulum ve Bakım**

**4.6.1** **Hazırlık**

Web tabanlı muhasebe uygulaması, yazılım geliştirme ve doğrulama süreçlerini başarıyla tamamlamış ve kuruluma hazır hale getirilmiştir. Bu süreçte, uygulamanın performansı, güvenilirliği ve uyumluluğu detaylı şekilde test edilmiş ve onaylanmıştır.

**4.6.2** **Kurulum**

Onaylanmış muhasebe uygulaması, belirlenen web sunucusuna yüklenerek kullanıcı erişimine açılmıştır. Kullanıcılar artık muhasebe işlemlerini web tarayıcıları üzerinden kolaylıkla yapabilirler. Uygulama, kullanıcıların ihtiyaçlarına uygun olarak konfigüre edilmiş ve çalışır durumda kullanıma sunulmuştur.

**4.6.3 Kullanıcı Eğitimi**

Kullanıcılar için kullanım kolaylığı sağlamak amacıyla, uygulamaya ilk giriş yapıldığında kullanım kılavuzları ve video rehberler sunulacaktır. Bu kaynaklar, kullanıcıların uygulamayı etkin bir şekilde kullanabilmeleri için adım adım yönlendirmeler sağlayacak ve uygulamanın temel özelliklerini açıklayacaktır.

**4.6.4** **Bakım ve Destek**

Web tabanlı muhasebe uygulaması düzenli olarak bakım ve güncelleme süreçlerine tabi tutulacaktır. Kullanıcı geri bildirimleri ve sistem performansı göz önünde bulundurularak, uygulamada olası hatalar düzeltilecek ve iyileştirmeler yapılacaktır. Kullanıcı destek ekibi, kullanıcıların karşılaştığı sorunlara hızlı ve etkili çözümler sunacak ve sürekli olarak kullanıcı memnuniyetini sağlamak için çalışacaktır.