Proje Analizi: Taş Kağıt Makas Oyunu

**1. Gereksinimler**

**İşlevsel Gereksinimler**

**Seçim Ekleme:** Kullanıcı, Taş, Kağıt veya Makas seçimlerinden birini yapabilmelidir. Bu seçim, oyunun temelini oluşturur.

**Seçim Listeleme:** Kullanıcının yaptığı önceki seçimler, bir liste halinde gösterilmelidir. Bu, oyuncunun hangi seçimleri yaptığını takip etmesine yardımcı olur.

**Karşılaştırma ve Sıfırlama:** Kullanıcının yaptığı seçim ile sistemin rastgele yaptığı seçim karşılaştırılmalıdır. Sonuçlar belirlendikten sonra, oyun durumu sıfırlanarak yeni bir oyuna başlanabilmelidir.

**Geçersiz Seçim:** Kullanıcı, yalnızca geçerli seçimler (Taş, Kağıt, Makas) yapabilmelidir. Geçersiz bir seçim yapılması durumunda, kullanıcıya uygun bir hata mesajı gösterilmelidir.

**Sonuç Hesaplama:** Kullanıcının ve sistemin seçimleri karşılaştırıldıktan sonra, oyunun sonucunu belirlemek için bir hesaplama yapılmalıdır. Kullanıcı kazanırsa, "Kazanıldı", sistem kazanırsa "Kaybedildi", berabere durumunda ise "Beraberlik" mesajı gösterilmelidir.

**Örnek Akış**

Kullanıcı oyuna başlar.

Kullanıcı bir seçim yapar (Taş, Kağıt, Makas).

Sistem rastgele bir seçim yapar.

Kullanıcının ve sistemin seçimleri karşılaştırılır.

Sonuç hesaplanır ve kullanıcıya gösterilir.

Kullanıcı, yeni bir oyun oynamak için sıfırlama işlemi yapabilir.

**Sonuç**

Bu proje analizi, Taş Kağıt Makas oyununun temel gereksinimlerini belirleyerek, geliştirme sürecinin yönünü ve kapsamını netleştirmektedir. İşlevsel gereksinimler, oyunun kullanıcı deneyimini ve işleyişini sağlamak için kritik öneme sahiptir.

**1.2. İşlevsel Olmayan Gereksinimler**

**1.2.1.Kullanılabilirlik**: Uygulamanın kullanıcı dostu bir arayüze sahip olması beklenmektedir. Kullanıcıların kolayca oyun ekleyip, güncelleyebilmeleri sağlanmalıdır.

**1.2.2.Performans**: Uygulama, hızlı yanıt sürelerine sahip olmalı ve büyük veri kümeleri ile çalışırken performans kaybı yaşanmamalıdır.

**1.2.3.Güvenlik**: Kullanıcı verileri ve oyun kayıtları güvenli bir şekilde saklanmalı, yetkisiz erişimlere karşı koruma sağlanmalıdır.

**1.2.4.Veri Yedekleme ve Kurtarma**: Oyun kayıtları ve kullanıcı verileri düzenli olarak yedeklenmeli, bir sorun durumunda kolayca kurtarılabilmelidir.

**1.2.5.Taşınabilirlik**: Uygulama, farklı platformlarda (web, mobil) çalışabilmelidir. Farklı işletim sistemlerinde uyumlu olmalıdır.

**1.2.6.Genişletilebilirlik**: Gelecekte yeni özellikler eklenebilmesi için sistem tasarımı genişletilebilir olmalıdır.

**1.2.7.Yedekleme ve Kurtarma İşlemleri**: Kullanıcı, yanlışlıkla silinen oyun kayıtlarını geri alabilmelidir. Bu işlemler için kullanıcı dostu bir arayüz sağlanmalıdır.

**1.2.8.Uyumluluk**: Uygulama, mevcut web tarayıcıları ile uyumlu olmalı ve en güncel standartları desteklemelidir.

**1.2.9.Bakım Kolaylığı**: Yazılım, bakım ve güncellemeler açısından kolay yönetilebilir olmalıdır. Kod yapısı ve belgeleri iyi organize edilmelidir.

**1.2.10.Ölçeklenebilirlik**: Kullanıcı sayısının artması durumunda sistemin performansını koruyarak daha fazla kullanıcıya hizmet verebilmesi sağlanmalıdır.

**2. Kullanım Durum Diyagramı**

**Aktörler**

1. **Kullanıcı**
2. **Sistem Yöneticisi**
3. **Ziyaretçi**

**Kullanım Durumları**

**Kullanıcı Aktörü:**

* **Seçim Ekleme**: Kullanıcı, yeni görevler ekleyebilir.
* **Seçim Listeleme**: Kullanıcı, mevcut görevleri listeleyebilir.
* **Karşılaştırma Sıfırlama:** Kullanıcının yaptığı seçim ile sistemin rastgele yaptığı seçim karşılaştırılmalıdır.

**Geçersiz Seçim:** Geçersiz bir seçim yapılması durumunda, kullanıcıya uygun bir hata mesajı gösterilmelidir.

* **Sonuç Hesaplama:** Kullanıcının ve sistemin seçimleri karşılaştırıldıktan sonra, oyunun sonucunu belirlemek için bir hesaplama yapılmalıdır.

**Ziyaretçi Aktörü:**

* **Kayıt Olma**: Ziyaretçi, yeni bir kullanıcı hesabı oluşturabilir.

**Sistem Yöneticisi Aktörü:**

* **Kullanıcı Yönetimi**: Sistem yöneticisi, kullanıcı hesaplarını yönetebilir (oluşturma, güncelleme, silme).
* **Sistem Ayarları**: Sistem yöneticisi, sistem ayarlarını yapılandırabilir.

**Diyagramın Görsel Temsili**

Açıklama : Kullanıcı, görevlerle ilgili tüm işlemleri gerçekleştirebilir ve bildirimleri alabilir. Giriş yapabilme yeteneği sayesinde kişisel verilerine erişebilir.

Ziyaretçi, sistemde bir hesap oluşturarak kullanıcı olma yolunda ilk adımı atabilir. Sistem yöneticisi, kullanıcıların hesaplarını yönetir ve sistem ayarlarını yapılandırarak sistemin düzgün çalışmasını sağlar.

Bu kullanım durumu diyagramı, sistemin nasıl etkileşimde bulunduğunu ve farklı aktörlerin hangi işlevleri gerçekleştirdiğini göstermektedir.

Projenin işleyişini ve gereksinimlerini daha iyi anlamak için bu diyagram yararlıdır.

**Kullanım Durumları**

**1. Seçim Yapma (Kullanıcı ve Bilgisayar Seçimi)**

Bu işlemde kullanıcıdan bir seçim alınır ve bilgisayar rastgele bir seçim yapar. Yapılacak işlemler adım adım şöyle:

**Açıklama:**

Kullanıcı, sisteme yeni bir görev ekler ve bu göreve bir isim, açıklama, öncelik seviyesi, bitiş tarihi gibi bilgiler ekleyebilir.

**Önkoşul:**

* Kullanıcı sisteme giriş yapmış olmalıdır.
* Görev ekleme formunun doğru şekilde doldurulması gerekmektedir.

**Adımlar:**

1. Kullanıcıdan bir seçim (taş, kağıt veya makas) alınır.
2. Bu seçim, geçerli olup olmadığı kontrol edilir (kullanıcı yanlış bir seçim yaparsa hata mesajı verilir).
3. Bilgisayar rastgele bir seçim yapar (taş, kağıt veya makas).
4. Seçimler karşılaştırılır ve sonucu belirlenir (kazanma, kaybetme veya beraberlik).

**2. Seçim Listeleme (Geçmiş Seçimlerin Görüntülenmesi)**

Bu aşamada, kullanıcı ve bilgisayar arasındaki geçmiş oyunların seçimlerini listeleyebiliriz. Bu, kullanıcıyı geçmiş oyunlar hakkında bilgilendirir.

**Açıklama:**

Kullanıcı, mevcut bir görevde değişiklik yapabilir.

**Önkoşul:**

* Kullanıcı görev sahibi olmalıdır.

Güncelleme yapılacak görev listede mevcut olmalıdır

**Adımlar:**

1. Bir liste oluşturulur ve her oyun sonrası kullanıcı ile bilgisayarın seçimleri bu listeye eklenir.
2. Kullanıcı, geçmiş seçimlerini görmek için bir komut verdiğinde liste yazdırılır.

**3. Seçim Karşılaştırma (Kullanıcı ve Bilgisayar Seçiminin Karşılaştırılması)**

Bu adımda, kullanıcı ve bilgisayar arasındaki seçimi karşılaştırıp sonucu belirleriz: Kazanma, kaybetme ya da berabere durumu.

**Adımlar:**

1. Kullanıcının ve bilgisayarın yaptığı seçimler karşılaştırılır.
2. Eğer seçimler eşitse (beraberlik) sonuç yazdırılır.
3. Taş-Kağıt-Makas kurallarına göre kazanan belirlenir.
   * Taş makası yener.
   * Kağıt taşı yener.
   * Makas kağıdı yener.

**4. Sıfırlama (Geçmişi Temizleme ve Yeni Başlangıç)**

Bu adımda, oyunun geçmişini sıfırlayarak yeni bir oyun başlatılmasını sağlarız. Geçmiş seçimler listesi temizlenir.

**Adımlar:**

1. Kullanıcı sıfırlama seçeneğini seçtiğinde, seçim geçmişi temizlenir.
2. Yeni bir oyun başlatılır.

**Açıklama:**

Sistemdeki tüm Seçim lerin listelenmesini sağlar. Kullanıcı, seçimlerin detaylarını görebilir.

**Önkoşul:**

* Sistemde eklenmiş görevlerin bulunması.
* Sistem, ilgili görevleri veritabanından çeker

**3.Dinamik Model**

**1. Seçim Ekleme (Oyuncu ve Bilgisayar Seçimi)**

Bu aşama, oyuncunun (kullanıcının) ve bilgisayarın seçimlerini almayı içerir. Bu, dinamik modelde önemli bir adım olup, her bir oyun adımında seçim yapılır ve kararlar kaydedilir.

**Adımlar:**

* Kullanıcıdan bir seçim (taş, kağıt veya makas) alınır.
* Bilgisayar, rastgele bir seçim yapar.
* Her iki seçim de bir veri yapısında (örneğin, liste veya tuple) kaydedilir, böylece ilerleyen adımlarda karşılaştırma yapılabilir.

**Dinamik Modeldeki Rolü:**

* Kullanıcı ve bilgisayarın her oyun için seçimlerini dinamik olarak toplar.
* Seçimler, sonraki adımlar (karşılaştırma ve sonuç hesaplama) için temel oluşturur.

**2. Seçim Listeleme (Geçmiş Seçimlerin Görüntülenmesi)**

Bu adımda, kullanıcı ve bilgisayar arasındaki önceki seçimler listelenir. Oyun ilerledikçe, geçmiş seçimler her seferinde güncellenir.

**Adımlar:**

* Tüm geçmiş seçimler bir liste veya veri yapısında tutulur.
* Kullanıcı seçimlerin geçmişini görmek isteyebilir. Bu durumda önceki seçimler ekrana yazdırılır.

**Dinamik Modeldeki Rolü:**

* Geçmiş seçimler, oyuncunun hangi seçimleri yaptığı hakkında bilgi verir.
* Bu bilgi, oyun stratejilerini belirlemek veya istatistiksel analiz yapmak için kullanılabilir.

**3. Karşılaştırma ve Sonuç Hesaplama**

Bu aşama, kullanıcı ile bilgisayar arasındaki seçimin karşılaştırılmasını ve kazananın belirlenmesini içerir. Ayrıca, her seçim için kazanan ve kaybeden sonuçları hesaplanır.

**Adımlar:**

* Kullanıcı ve bilgisayarın yaptığı seçim karşılaştırılır.
* Taş, kağıt, makas oyun kurallarına göre kazanan belirlenir:
  + Taş makası yener.
  + Kağıt taşı yener.
  + Makas kağıdı yener.
  + Eşleşme varsa, oyun berabere biter.
* Sonuç bir değişkende saklanır ve kullanıcıya bildirilir

**Dinamik Modeldeki Rolü:**

* Seçimler karşılaştırılır, dinamik olarak oyunun sonucu hesaplanır.
* Bu bilgi daha sonra kullanıcıya geri döndürülür.

**4. Geçersiz Seçim (Yanlış Giriş)**

Kullanıcı yanlış seçim yaparsa, yani 'taş', 'kağıt' veya 'makas' dışında bir şey girerse, oyun bunu geçersiz olarak kabul eder ve hata mesajı verir. Bu aşama, kullanıcı etkileşimini yönetir ve yanlış girişleri kontrol eder.

**Adımlar:**

* Kullanıcı geçerli bir seçim yapmadığında, hata mesajı verilir.
* Oyun, geçersiz girişe izin vermez ve kullanıcıyı doğru seçim yapmaya yönlendirir.

**Dinamik Modeldeki Rolü:**

* Kullanıcı etkileşimi dinamik olarak kontrol edilir.
* Kullanıcının yanlış bir seçim yapması, oyunun diğer adımlarını engeller ve doğrulama yaparak tekrar seçim yapılması sağlanır.

**5. Sonuç Hesaplama ve Geri Bildirim**

Sonuç, her oyun sonrasında kullanıcının ne yaptığını ve kazananı belirlemek için hesaplanır. Bu bilgi dinamik olarak güncellenir ve kullanıcıya bildirilir.

**Adımlar:**

* Kullanıcı ve bilgisayarın seçimleri karşılaştırılır.
* Kazanan, kaybeden veya berabere durumu hesaplanır.
* Sonuç, kullanıcının ekranına yazdırılır.

**Dinamik Modeldeki Rolü:**

* Oyun her defasında dinamik olarak sonuç hesaplar ve bu sonuç kullanıcıya anında iletilir.
* Bu işlem, her yeni seçim sonrası yenilenir.

**Genel Dinamik Modelin Yapısı:**

* **Seçim Ekleme**: Kullanıcı ve bilgisayarın seçimlerini toplar, her seçim kaydedilir.
* **Geçersiz Seçim**: Yanlış seçimler kontrol edilir, geçersiz olduğunda kullanıcıya bildirilir ve tekrar seçim yapması istenir.
* **Karşılaştırma ve Sonuç Hesaplama**: Kullanıcı ve bilgisayar seçimleri karşılaştırılır, kurallara göre kazanan belirlenir ve sonuç hesaplanır.
* **Sonuç Gösterimi**: Sonuç dinamik olarak kullanıcıya bildirilir, her seçimde yenilenir.
* **Geçmiş Seçimlerin Listelenmesi**: Önceki seçimler kaydedilir ve gerektiğinde kullanıcıya gösterilir.

### ****4. Görev Yönetim Sistemi Nesne Tabanlı Modeli****

## ****4.1. Sınıflar****

### ****1.1 Task (Görev)****

**Açıklama:**  
Bir oyunda yapılacak işlemleri temsil eder (örneğin, kullanıcı girişini işleme veya sonucu hesaplama).

#### ****Metodlar:****

* execute(): Görevi başlatır ve işlemeyi gerçekleştirir.
* update\_status(status): Görevin durumunu günceller.
* get\_info(): Görevin detaylarını döner.

### ****User (Kullanıcı)****

**Açıklama:**  
Oyunu oynayan kişiyi temsil eder. Seçimler yapar ve sonuçları alır.

#### ****Özellikler:****

* user\_id: Kullanıcının benzersiz kimliği.
* choice: Kullanıcının yaptığı seçim (taş, kağıt, makas).
* score: Kullanıcının kazandığı oyun sayısı.

#### ****Metodlar:****

* make\_choice(choice): Kullanıcının seçim yapmasını sağlar.
* update\_score(): Kullanıcının skorunu artırır.
* get\_details(): Kullanıcı bilgilerini döner.

### ****Database (Veritabanı)****

**Açıklama:**  
Kullanıcı ve görev bilgilerini saklayan sistem.

#### ****Özellikler:****

* users: Kayıtlı kullanıcıların bilgileri.
* tasks: Kayıtlı görevlerin bilgileri.
* results: Geçmiş oyun sonuçları.

#### ****Metodlar:****

* add\_user(user): Yeni bir kullanıcı ekler.
* save\_task(task): Yeni bir görev ekler.
* get\_user(user\_id): Kullanıcıyı kimliğiyle arar.
* get\_results(): Geçmiş oyun sonuçlarını döner.

## sınıflar arası ilişkiler

**1. Task - TaskManager İlişkisi**

**İlişki Türü:** *Birçoktan bire (Many-to-One)*

**Açıklama:**

* **TaskManager**, birden fazla **Task** nesnesini yönetir.
* Yeni görevler eklenir, çalıştırılır ve tamamlandıkça sistemden kaldırılır.

**Görevler:**

1. **Task** nesnesi, belirli bir görevi temsil eder ve işlem için kendi verilerini taşır.
2. **TaskManager**, görevlerin sırayla çalıştırılmasını ve durumlarının takip edilmesini sağlar.

**Etkileşim:**

* TaskManager.add\_task(task): Yeni bir görevi yöneticinin listesine ekler.
* Task.execute(): Yöneticinin belirlediği sırada görev çalıştırılır.
* TaskManager.run\_next\_task(): Bir sonraki görev çalıştırılır.

**2. TaskManager - User İlişkisi**

**İlişki Türü:** *Bire bir (One-to-One)*

**Açıklama:**

* **TaskManager**, kullanıcı tarafından başlatılan görevleri kontrol eder.
* Kullanıcı, seçim yaptığında görev yöneticiye işlenmesi için talimat verir.

**Görevler:**

1. **User**, seçim yapar ve bu seçim bir **Task** olarak sisteme eklenir.
2. **TaskManager**, kullanıcının seçiminden oluşan görevleri yürütür.

**Etkileşim:**

* User.make\_choice(choice): Kullanıcının seçimi bir görev olarak yöneticide işlenir.
* TaskManager.add\_task(task): Kullanıcı seçiminden oluşan görev listeye eklenir.

**4. TaskManager - Database İlişkisi**

**İlişki Türü:** *Bire bir (One-to-One)*

**Açıklama:**

* **TaskManager**, görevlerin durumlarını ve geçmişini **Database** nesnesinde saklar.
* Tamamlanan veya iptal edilen görevler veritabanına kaydedilir.

**Görevler:**

1. **TaskManager**, görevleri başlatmadan önce veritabanına kaydeder.
2. **Database**, görevlerin geçmişini tutar ve gerektiğinde bu verilere erişir.

**Etkileşim:**

* TaskManager.add\_task(task): Görev eklenirken veritabanına kaydedilir.
* Database.save\_task(task): Görev bilgisi veritabanına kaydedilir.

**5. User - Database İlişkisi**

**İlişki Türü:** *Bire bir (One-to-One)*

**Açıklama:**

* **User** bilgileri ve skorları **Database** nesnesinde saklanır.
* Sistem, kullanıcının geçmiş oyun bilgilerini veritabanından alır.

**Görevler:**

1. Kullanıcı kimliği ve seçimleri **Database** nesnesine kaydedilir.
2. Kullanıcının skor durumu ve geçmişi veritabanında tutulur

**Class Diagram Özeti**

Aşağıda nesne tabanlı modelini dahi iyi anlamamızı sağlayacak sınıf diyagramları özetlenmiştir:

+------------------------------+

| Oyun |

+------------------------------+

| - secim\_user: str |

| - secim\_computer: str |

| - sonuc: str |

| + kullanici\_secimi(): str |

| + bilgisayar\_secimi(): str |

| + sonuc\_belirle(): str |

| + sonuclar(): str |

| + tekrar\_oyna(): bool |

+---------------------------+

| Kullanici |

+----------------------------+

| - secim: str |

| + secim\_al(): str |

+----------------------------+

+---------------------------+

| Bilgisayar |

+---------------------------+

| - secim: str |

+----------------------------+

| + secim\_al(): str |

+---------------------------+

**Açıklama:**

**Oyun**:

secim\_user: Kullanıcının yaptığı seçim. Kullanıcı taş, kağıt ya da makası seçmeli.

secim\_computer: Bilgisayarın yaptığı seçim.

sonuc: Oyun sonucunu belirler.(Kazanma, kaybetme, beraberlik.)

kullanici\_secimi():Kullanıcıdan seçim alır.

bilgisayar\_secimi():Bilgisayarın rastgele seçim yapmasını sağlar.

sonuc\_belirle():Kullanıcı,bilgisayar seçimlerini karşılaştırarak kazananı belirler.

sonuclar(): Sonucu ekrana yazdırır.

tekrar\_oyna(): Kullanıcıya tekrar oyun oynayıp oynamayacağını sorar.

**Kullanici**:

secim: Kullanıcının yaptığı seçim.

secim\_al(): Kullanıcıdan geçerli bir taş, kağıt ya da makas seçimi alır.

**Bilgisayar**:

secim: Bilgisayarın yaptığı seçim.

secim\_al(): Bilgisayarın rastgele seçim yapmasını sağlar.

**5.Kullanıcı Arayüzü**

Bu proje için kullanıcı arayüzü metin tabanlıdır ve Python'un konsol/terminal ortamında çalışır. Bu nesne tabanlı model, Taş Kağıt Makas oyunundaki nesnelerin arasındaki ilişkileri ve işlevleri daha ayrıntılı bir şekilde sunar.Bu yapıda, her sınıfın belirli bir sorumluluğu vardır. Bu arayüz kullanıcıdan secimi alma, seçimleri karşılaştırma, sonucu belirleme, sonucu ekrana yazma gibi temel işlevleri hızlıca erişilebilir hale getirir.Kullanıcı arayüzü, aşağıdaki adımlarla etkileşime girer:

Arayüz Akışı:

Kullanıcıya bir seçim yapması için seçenekler sunulur (Taş, Kağıt, Makas).

Kullanıcı bir seçim yaptıktan sonra, bilgisayar rastgele bir seçim yapar.

Oyun sonucu ekranda gösterilir (kazanma, kaybetme veya berabere).

Kullanıcıya oyunun yeniden başlatılması için seçenek sunulur.

Örnek Arayüz:

Oyun Başladı! Lütfen bir seçenek girin:

1. Taş 2. Kağıt 3. Makas

Seçiminizi yapın (1/2/3): 1

Bilgisayarın seçimi: Kağıt

Sonuç: Kaybettiniz! Kağıt, Taş'ı yener.

Yeni bir oyun oynamak ister misiniz? (Evet/Hayır)

**Uygulama Özellikleri:**

1.Basit metin tabanlı giriş ve çıkış.

2.Kullanıcı seçimi yapılacak menüler ve girdiler.

3.Oyun sonucu ile ilgili açıklamalar.

**5.1 Ana Ekran**

Ana ekran, kullanıcının oyunun başlatılacağı ve temel seçim yapabileceği ilk ekrandır. Bu ekranda, kullanıcıya oyunun ne olduğu, nasıl oynanacağı ve hangi seçeneklerin mevcut olduğu ile ilgili açıklamalar ve menüler sunulmalıdır.

**Ana Ekran İşlevleri:**

**1.Başlangıç Mesajı ve Kurallar:**

Kullanıcıya oyunun adı ve kısa bir açıklama sunulur.

Oyunun nasıl oynandığına dair basit kurallar açıklanır.

Örnek:

Taş, Kağıt, Makas Oyunu

Taş Kağıt Makas oyununu oynamaya hoş geldiniz!

Kurallar:

Taş, Makas'ı yener.

Makas, Kağıt'ı yener.Kağıt, Taş'ı yener.

Aynı seçimde oyun berabere biter.

Oyuna başlamak için bir seçenek girin:

Oyun Başlat

2. Çıkış

**2.Seçenekler:**

a)Kullanıcıya iki ana seçenek sunulur: oyuna başlama veya çıkma.

b)Kullanıcı bir seçenek seçer, oyun başlar veya uygulama kapanır.

**5.2 Görev Ekleme Ekranı** Bu ekran, oyuncunun bir oyun turu başladığında bir seçim yapması

gerektiği ekrandır.Kullanıcı taş, kağıt, makas birini seçerek oyuna katılır.

**Görev Ekleme Ekranı İşlevleri:**

**1.Seçim Yapma:**

Kullanıcıya taş, kağıt ve makas seçeneklerinden birini seçmesi için metin tabanlı bir menü gösterilir.

Örnek:

Seçiminizi yapın:1. Taş 2. Kağıt 3. Makas

**2.Kullanıcı Girişi:**Kullanıcı bir sayı girer (1, 2, 3).

Seçim yapıldıktan sonra, bilgisayarın rastgele bir seçim yapması için sistem çalışmaya başlar.

**5.3 Sonuç Güncelleme Ekranı**

Sonuç güncelleme ekranı, oyun turunun sonucunu göstermek ve bir sonraki oyuna geçişi sağlamak için kullanılır. Bu ekranda, kullanıcının önceki seçimleri ve bilgisayarın seçimi görüntülenir, ardından oyun sonucu kullanıcıya bildirilir.

**Sonuç Güncelleme Ekranı İşlevleri:**

**1.Kullanıcı, Bilgisayar Seçimlerinin Görüntülenmesi:**

Kullanıcının ve bilgisayarın yaptığı seçimler ekranda görünür.

Örneğin, eğer kullanıcı "Taş" ve bilgisayar "Kağıt" seçtiyse, şu şekilde bir çıktı olabilir:

Kullanıcı Seçimi: Taş

Bilgisayar Seçimi: Kağıt

**2.Sonuç Gösterimi:**

Oyun sonucunun açıklanması yapılır. Örneğin:

“Taş, Kağıt'ı yener" ifadesi veya

"Kağıt, Taş'ı yener" ifadesi.

Beraberlik durumunda: "Beraberlik!" şeklinde bir mesaj.

Örnek:

Sonuç: Kaybettiniz! Kağıt, Taş'ı yener.

**3.Yeni Oyun Başlatma Seçeneği:**Kullanıcıya bir sonraki oyunu başlatmak için seçenek sunulur.

Örnek:

Yeni bir oyun başlatmak ister misiniz?

1. Evet 2. Hayır (Çıkış)

**5.4 Görev Detayları Ekranı**

Bu ekran, kullanıcıya oyun sonucu ve önceki oyun turlarına dair daha ayrıntılı bir bilgi verir. Bu ekran, oyuncunun önceki seçimleri ve sonuçlarını görmek isteyebileceği bir "geçmiş" özelliği de sunabilir.

**Görev Detayları Ekranı İşlevleri:**

**1.Oyun Geçmişi Gösterimi:**Kullanıcıya, önceki oyun turlarındaki seçimler ve sonuçlar gösterilir.

Örnek:

Oyun Geçmişi:

1. Taş vs Kağıt → Kaybettiniz

2. Kağıt vs Makas → Kazandınız

3. Makas vs Makas → Beraberlik

**2.Genel Skor:**

Kullanıcıya toplam kazançlar, kayıplar ve beraberlikler gösterilir.

Örnek:

- Kazanılan Oyunlar: 1

- Kaybedilen Oyunlar: 1

- Beraberlikler: 1

**3.Yeniden Başlatma veya Çıkış Seçenekleri:**

Kullanıcıya, yeni bir oyun başlatma veya oyundan çıkma seçeneği sunulur.

Örnek:

Yeni bir oyun oynamak ister misiniz?

1. Evet 2. Hayır

**5.5 Sonuç Görüntüleme Ekranı**

Sonuç görüntüleme ekranı, her bir oyun turunun ardından kullanıcıya oyunun sonunda elde edilen sonuçları net bir şekilde sunar. Bu ekran, oyuncunun oyunun sonucunu görmek için son adımda etkileşimde bulunduğu ekrandır.

**Sonuç Görüntüleme Ekranı İşlevleri:**

**1.Oyun Sonucu:**

Kullanıcıya oyun sonucunun açık bir şekilde gösterildiği ekrandır.Kazanma, kaybetme veya beraberlik durumu belirtilir.

Örnek:

Sonuç:

- Kullanıcı Seçimi: Taş

Bilgisayar Seçimi: Kağıt

Sonuç: Kaybettiniz! Kağıt, Taş'ı yener.

**2.Oyun Sonu Mesajı:**

Eğer oyun tamamlanmışsa, kullanıcıya oyunun bitişi hakkında bilgi verilir.

Örnek:

Oyun sona erdi! Skor: Kazanma: 2 - Kaybetme: 3

**3.Yeni Oyun veya Çıkış Seçeneği:**Kullanıcıya yeni bir oyun başlatma veya oyundan çıkma seçeneği sunulur.

Örnek:

Yeni bir oyun başlatmak ister misiniz?

1. Evet 2. Hayır (Çıkış)