# T.C KARABÜK ÜNİVERSİTESİ

# MÜHENDİSLİK FAKÜLTESİ YAZILIM MÜHENDİSLİĞİ

## YAZILIM MÜHENDİSLİĞİ YAZILIM İNŞASI DERSİ PROJE ÖDEVİ:

### TAŞ KAĞIT MAKAS OYUNU GÖRÜNTÜ İŞLEMESİ

PROJE SAHIPLERI

2310238039 YAĞMUR İNCİ

2310238041 FATMA BERİK

2310238038 İREM NAZ GEÇİKLİ

2310238512 MERYEM ALSALEH

## 1. Seçim Ekleme

*Tasarımdaki hedefler:*

1. **Kullanılabilirlik:** Kullanıcıların seçimlerini hızlı ve kolay bir şekilde yapabilmesi önemlidir. Bu, oyun arayüzünün sade ve anlaşılır olmasını gerektirir. Kullanıcılar Taş, Kağıt veya Makas seçimini kolayca yapabilmelidirler.
2. **Performans**: Seçim ekleme işlemi, oyunun hızlı çalışmasını sağlamalıdır. Kullanıcı seçim yaptıktan sonra, AI'nin ve oyun bilgisinin işlenmesi hızlı olmalıdır.
3. **Veri Güvenliği:** Kullanıcıların seçim verileri, güvenli bir şekilde işlenmeli ve saklanmalıdır. Oyun sırasında kullanıcıların verilerinin kötüye kullanılmaması için uygun güvenlik önlemleri alınmalıdır.
4. **Ölçeklenebilirlik:** Sistem daha fazla kullanıcıya hizmet edebilecek şekilde tasarlanmalıdır. Oyun ekibi veya yapay zeka sürekli olarak yeni özellikler eklemeyi planlıyorsa, seçim ekleme altyapısı bu büyümeyi desteklemelidir.
5. **Taşınabilirlik:** Python dilinde geliştirilen bu sistem farklı platformlarda çalışacak şekilde tasarlanabilir. Web tabanlı veya masaüstü uygulama olarak kullanıcıların farklı cihazlarda seçim yapabilmesi sağlanmalıdır.
6. **Genişletilebilirlik**: Seçim ekleme, ileride başka oyun modları veya farklı seçim türlerinin eklenmesine olanak sağlayacak şekilde modüler olmalıdır.

**2. Karşılaştırma Sıfırlama**

*Tasarımdaki hedefler:*

1) **Kullanılabilirlik:** Kullanıcı, oyun karşılaştırmasını sıfırlamak istediğinde işlemin basit ve sezgisel olması gerekir. Bu, oyun ekranında net bir sıfırlama butonu ile sağlanabilir.

2) **Performans:** Karşılaştırma sıfırlama işlemi hızlı bir şekilde yapılmalıdır. Oyun ekranı, sıfırlama işleminden sonra gecikmesiz olarak yenilenmelidir.

3) **Veri Güvenliği:** Oyun sıfırlanırken önceki oyun verileri ve kullanıcının seçim geçmişi güvenli bir şekilde sıfırlanmalıdır.

4) **Ölçeklenebilirlik:** Daha fazla oyun modu veya seçim türü eklenmesi durumunda, sıfırlama işleminin düzgün çalışması sağlanmalıdır. Özellikle online çok oyunculu modlarda sıfırlama işleminin doğru bir şekilde yapılması gereklidir.

5) **Taşınabilirlik:** Karşılaştırma sıfırlama fonksiyonu farklı platformlarda sorunsuz çalışmalıdır.

6) **Genişletilebilirlik:** Sıfırlama fonksiyonu, ileride yeni özellikler veya oyun modları eklenirse bu yeniliklere kolayca adapte olabilecek şekilde tasarlanmalıdır.

**3. Geçersiz Seçim**

*Tasarımdaki hedefler:*

1) **Kullanılabilirlik:** Geçersiz seçimlerin anlaşılır şekilde kullanıcıya bildirilmesi gerekmektedir. Kullanıcı yanlış seçim yaparsa, sistem hemen uyarı vermeli ve doğru seçim yapabilmesi için rehberlik etmelidir.

2) **Performans:** Geçersiz seçimler hızlı bir şekilde tespit edilmeli ve kullanıcıya anında geri bildirim verilmelidir. Bu, oyunun kesintisiz çalışmasını sağlar.

3) **Veri Güvenliği:** Geçersiz seçimlere dair herhangi bir kritik veri kaybı olmamalıdır. Kullanıcılar yanlış seçim yapsalar bile, sistem veri bütünlüğünü koruyarak doğru işlemleri gerçekleştirmelidir.

4) **Ölçeklenebilirlik:** Geçersiz seçimlere dair yeni kontroller veya farklı doğrulama kuralları eklendiğinde, sistemin bu eklemeleri sorunsuz bir şekilde yapabilmesi sağlanmalıdır.

5) **Taşınabilirlik:** Geçersiz seçim fonksiyonu tüm platformlarda benzer şekilde çalışmalıdır.

6) **Genişletilebilirlik:** Geçersiz seçimler için kullanıcı dostu bildirimler veya yeni kontrol mekanizmaları eklemek kolay olmalıdır.

**4. Sonuç Hesaplama**

*Tasarımdaki hedefler:*

1) **Kullanılabilirlik:** Sonuç hesaplama işlemi hızlı ve anlaşılır olmalıdır. Kullanıcılar sonuçları kolayca görebilmelidir.

2) **Performans:** Hesaplama işlemi hızlı olmalıdır; özellikle AI'nin taş, kağıt, makas seçimlerini analiz etmesi ve sonucu belirlemesi hızlı bir şekilde gerçekleşmelidir.

3) **Veri Güvenliği:** Sonuçlar doğru bir şekilde hesaplanmalı ve güvenli bir şekilde saklanmalıdır. Örneğin, kullanıcıların kazanma ve kaybetme durumları kişisel verileriyle bağlantılı olmamalıdır.

4) **Ölçeklenebilirlik:** Sonuç hesaplama, yeni oyun kuralları eklenmesi durumunda sorunsuz çalışabilmelidir. Oyunun farklı versiyonları ve modları eklendikçe bu süreç değişmemelidir.

5) **Taşınabilirlik:** Sonuç hesaplama algoritması tüm cihazlarda hızlı çalışmalıdır. Hesaplama işlemi platform bağımsız olmalıdır.

6) **Genişletilebilirlik:** Hesaplama algoritması, ileride yeni seçim türleri veya kurallar eklenerek genişletilebilir olmalıdır. Örneğin, yeni zorluk seviyeleri veya AI geliştirmeleri hesaplamaya dahil edilebilir.

**5.Diğer Dengelemeler:**

1) **Kullanılabilirlik vs. Özellik Zenginliği**: Kullanılabilirlik, oyunun anlaşılabilir ve kolay kullanılabilir olmasıyla ilgilidir. Ancak, özellik zenginliği daha fazla fonksiyon ve seçenek eklemeyi gerektirir. Bu dengenin sağlanması, kullanıcıların karmaşık olmayan bir deneyimle, ancak zengin seçeneklerle oyun oynamalarını mümkün kılmalıdır.

2) **Performans vs. Güvenlik:** Hızlı bir performans sağlamak, sistemin veri güvenliği sağlamasına engel olmamalıdır. Güvenlik önlemleri oyun deneyimini etkilemeden arka planda çalışmalıdır.

3) **Geliştirme Süresi vs. Genişletilebilirlik:** Hızlı bir geliştirme süreci, genişletilebilirlikten taviz vermemelidir. Tasarımın esnek olması, oyun geliştikçe yeni özelliklerin eklenmesini kolaylaştırır.

4) **Veri Güvenliği vs. Taşınabilirlik:** Taşınabilirlik, farklı cihazlarda veri güvenliğinin korunmasıyla paralel olmalıdır. Bu, özellikle bulut tabanlı veya çoklu platform desteği gerektiren projelerde önemlidir.

5) **Hızlı Erişim vs. Veritabanı Bütünlüğü: Hızlı** erişim sağlamak, veritabanı bütünlüğüne zarar vermemelidir. Her iki hedef de paralel bir şekilde optimize edilmelidir, özellikle çok oyunculu oyunlarda veri bütünlüğü hayati önem taşır.

## 2)SİSTEM MİMARİSİ

**1. Alt Sistem Ayrıştırma**

a) Kullanıcı Arayüzü (UI) Alt Sistemi

* Kullanıcıların oyuna giriş yapmasını, seçimlerini yapmasını ve sonuçları görmesini sağlar.
* Basit bir konsol tabanlı ya da grafiksel bir arayüzle tasarlanabilir.
* Kullanıcı etkileşimleri şunları içerir:
  + Oyuncu giriş ekranı (isim veya kimlik doğrulama).
  + Oyuncunun taş, kağıt veya makas seçim ekranı.
  + Oyun sonuçlarının ve toplam skorların görüntülenmesi.

b) İş Mantığı Alt Sistemi

* Oyunun ana kurallarını ve algoritmalarını içerir:
  + Kullanıcıların seçimini alır ve karşılaştırır.
  + Taş, kağıt, makas kurallarını (örneğin, taş makası yener) uygular.
  + Kazananı belirler ve sonuçları kullanıcıya döner.
  + Skor takibini yapar.
* İş mantığı, sistemin diğer bölümlerine ayrıştırılmış bir şekilde entegre edilir.

c) Veri Yönetimi Alt Sistemi

* Oyuncu bilgileri ve oyun sonuçlarının geçici veya kalıcı olarak saklandığı sistemdir.
* Oyun sırasında şunları yönetir:
  + Oyuncu seçimleri.
  + Skor tabloları.
  + Oyun sonuçları.
* Veri saklama işlemleri bir dosya sistemi (örneğin, JSON veya CSV dosyası) veya bir veri tabanı kullanılarak yapılabilir.

d) Güvenlik ve Erişim Kontrolü Alt Sistemi

* Oyuncu kimlik bilgilerinin doğruluğunu ve güvenliğini sağlar.
* Yetkisiz erişimi engellemek için kullanıcı doğrulama mekanizmaları uygular.
* Oyun sırasında veri bütünlüğünü korur.
* Giriş kontrolü ve oyun sırasında veri erişim sınırları tanımlanır.

**2. Donanım/Yazılım Eşleme**

a) Oyuncu Tarafı

* Oyuncunun oyun arayüzüne eriştiği cihaz (örneğin, bilgisayar, mobil cihaz veya terminal).
* Yazılım katmanı:
  + Kullanıcı arayüzü (UI) uygulamaları (örneğin, Python konsol tabanlı veya GUI tabanlı uygulama).
  + Oyuncunun seçim yaptığı ve sonuçları gördüğü yer.

b) İstemci Tarafı

* Oyun sunucusuna bağlanan istemci yazılımı.
* Çok oyunculu bir versiyonda, istemci ve sunucu arasında veri iletişimini sağlar.
* Yerel bir ortamda, istemci ile sunucu tek bir cihazda olabilir.

c) Seçim Tarafı

* Oyuncuların yaptığı seçimlerin işlendiği ve karşılaştırıldığı yazılım modülü.
* İş mantığını ve oyun kurallarını uygular.
* Oyuncu seçimini alır ve oyun sonucunu hesaplar.

**3. Kalıcı Veri Yönetimi**

* Oyunun uzun süreli verilerini saklamak için kullanılır.
  + Oyuncu bilgileri (örneğin, isimler ve skorlar).
  + Oyun geçmişi (örneğin, yapılan seçimler ve sonuçlar).
* Veri saklama yöntemleri:
  + Dosya tabanlı yaklaşım: JSON, CSV veya metin dosyaları.
  + Veri tabanı tabanlı yaklaşım: SQLite veya başka bir ilişkisel veri tabanı.
* Kalıcı veri, oyun istatistiklerinin daha sonra analiz edilmesini sağlar.

**4. Erişim Kontrolü ve Güvenlik**

a) Oyuncu Bilgileri Girişi

* Oyuncu, oyun başlamadan önce kullanıcı adı veya bir kimlik bilgisi girer.
* Girilen bilgiler doğruluk ve bütünlük açısından kontrol edilir.

b) Oyuncu Seçim Yapar

* Oyuncuların taş, kağıt veya makas seçimi güvenli bir şekilde alınır.
* Yanlış veya eksik girişlere karşı koruma sağlanır.

c) Veri Güvenliği

* Oyuncuların seçimleri ve sonuçları, şifreli bir biçimde geçici hafızada saklanır.
* Veriler yalnızca gerekli modüller tarafından erişilebilir.

d) Güvenlik Duvarı ve IP Filtreleme

* Çok oyunculu bir uygulamada, yetkisiz erişimleri engellemek için güvenlik duvarları veya IP filtreleme kullanılabilir.
* Oyunun bir istemci-sunucu mimarisi varsa, istemcilerin yalnızca yetkili IP adreslerinden bağlanmasına izin verilir.

e) Oyun Sonuç Kayıtları

* Oyun sonuçları, güvenli bir şekilde saklanır ve erişim kontrollüdür.
* Veriler, yalnızca yönetici veya oyuncu tarafından görüntülenebilir.

**5. Sınır Koşulları**

a) Başlatma Koşulları

* Sistem, oyuncu bilgileri ve başlangıç parametreleri doğru bir şekilde alındığında oyunu başlatır.
* Gerekli yazılım bileşenleri başlatılır ve başlangıç testleri yapılır.

b) Hata Yönetimi

* Yanlış girişler (örneğin, taş/kağıt/makas dışındaki bir seçim) durumunda hata mesajı gösterilir.
* Sistem, hata durumlarında varsayılan değerlere döner veya kullanıcıyı bilgilendirir.

c) Kapanış Koşulları

* Oyun tamamlandığında, oyuncuların skorları kaydedilir.
* Sistem düzgün bir şekilde kapanır ve tüm geçici veriler temizlenir.

d) Beklenmeyen Durumlar

* Sistem çökmesi veya ağ bağlantısı kesintisi gibi durumlar için kurtarma mekanizmaları bulunur.
* Beklenmeyen bir giriş alındığında kullanıcı yeniden seçim yapmaya yönlendirilir.
* Oyun sırasında meydana gelebilecek hatalar loglanır ve geliştiriciye iletilir.

## 3.Alt Sistemi Hizmetleri

**1. Seçim Ekleme Hizmeti**

* **Kullanıcı Arayüzü Alt Sistemi:**
  + **Giriş Ekranı**: Kullanıcı, oyunun başlangıcında taş, kağıt veya makas gibi bir seçenek seçmek için giriş ekranında yönlendirilir. Bu ekranda, oyun hakkında açıklamalar ve seçim butonları bulunur.
  + **Görev Listesi Görüntüleme**: Seçilen görevler veya oyun seansları listelenebilir. Bu ekran, kullanıcının önceki seçimlerini görmesine yardımcı olur.
  + **Görev Ekleme Ekranı**: Oyuncu, sistemde yeni bir seçim eklemek için bu ekrandan seçim yapabilir. Bu seçimler, taş, kağıt veya makas olarak tanımlanır.
* **Yönetim Alt Sistemi:**
  + **Görev İşleme**: Yönetim sistemi, oyuncuların seçtikleri taş, kağıt veya makas görevlerini alır ve işleme koyar. Herhangi bir yeni görev eklenmesi veya seçilmesi işlemi burada gerçekleşir.
  + **Kullanıcı Kimlik Doğrulama**: Seçimlerin doğru bir şekilde yönetilmesi ve izlenmesi için her oyuncunun kimliği doğrulanır.
  + Bu, her oyuncunun yalnızca kendisinin seçimleriyle etkileşimde bulunmasını sağlar.
* **Model Alt Sistemi:**
  + **Görev Depolama ve Erişim**: Kullanıcılar tarafından yapılan seçimler, veritabanında saklanır ve yönetilir. Taş, kağıt veya makas seçimi depolanır ve gerektiğinde bu verilere erişilir.

**2. Seçim Listeleme Hizmeti**

* **Kullanıcı Arayüzü Alt Sistemi:**
  + **Görev Arama ve Filtreleme**: Kullanıcı, önceki oyun seçimlerini kolayca arayabilir ve belirli kriterlere göre filtreleyebilir. Örneğin, sadece "taş" seçmiş olan oyunları görmek isteyebilir.
  + **Görev Listeleme**: Kullanıcı, tüm seçimlerini sıralayabilir ve en son oynadığı oyunları listeleyebilir.
* **Yönetim Alt Sistemi:**
  + **Görev Arama ve Filtreleme Yönetimi**: Yönetim sistemi, kullanıcılara seçimlerinin tüm geçmişini listeleme ve filtreleme imkanı sunar. Bu sayede, geçmiş oyunlar kolayca takip edilebilir.
  + **Bildirim Yönetimi**: Seçimlerle ilgili bildirimler, kullanıcıya hangi seçimlerin yapıldığına dair bilgi verir. Kullanıcı, listelemeden haberdar edilir.
* **Model Alt Sistemi:**
  + **Kullanıcı Verisi Yönetimi**: Seçimlerin depolandığı veritabanı üzerinden kullanıcıların seçim geçmişi listelenebilir. Bu bilgiler, kullanıcı adına özelleştirilmiş listeler sunar.

**3. Karşılaştırma ve Sıfırlama Hizmeti**

* **Kullanıcı Arayüzü Alt Sistemi:**
  + **Görev Düzenleme ve Güncelleme Ekranı**: Kullanıcı, eski oyunlarını karşılaştırmak ve yeni seçimler yapmak için oyunları yeniden düzenleyebilir. Bu ekran, geçmiş oyunların seçimini ve sıralamasını sunar.
  + **Hata ve Durum Mesajları**: Oyun sırasında bir hata meydana gelirse veya seçim geçersizse, kullanıcıya açıklayıcı hata mesajları gösterilir.
* **Yönetim Alt Sistemi:**
  + **Görev İşleme ve Karşılaştırma**: Yönetim, oyuncunun yaptığı seçimleri karşılaştırarak hangi oyuncunun kazandığını belirler. Bu işlem, her oyun turunda tekrarlanır.
  + **Bildirim Yönetimi**: Oyuncuya kazananı, kaybedeni veya berabere durumu bildiren bir sistem bulunur.
* **Model Alt Sistemi:**
  + **Bildirim Depolama ve Erişim**: Sonuçlar ve oyun karşılaştırmalarıyla ilgili bildirimler depolanır ve gelecekteki erişimler için saklanır.

**4. Geçersiz Seçim Hizmeti**

* **Kullanıcı Arayüzü Alt Sistemi:**
  + **Bildirim Gösterimi**: Kullanıcı geçersiz bir seçim yaparsa, ekranında uyarılar veya hata mesajları gösterilir. Bu mesajlar, doğru seçimin yapılmasını teşvik eder.
* **Yönetim Alt Sistemi:**
  + **Kullanıcı Kimlik Doğrulama**: Geçersiz bir seçim yapıldığında, sistem doğrulama mekanizmasını devreye sokarak hatalı seçimlerin önüne geçer. Her seçim, sistem tarafından doğrulanır.
  + **Görev Arama ve Filtreleme Yönetimi**: Geçersiz seçimler sistemde kaydedilmez ve yönetim panelinde yalnızca geçerli seçimler görüntülenir.
* **Model Alt Sistemi:**
  + **Görev Depolama ve Erişim**: Geçersiz seçimler kaydedilmez ve sistem, sadece geçerli verilerle işlem yapar. Geçersiz seçimler, oyunun ilerleyişiyle ilgili kayıtlara dahil edilmez.

**5. Sonuç Hesaplama Hizmeti**

* **Kullanıcı Arayüzü Alt Sistemi:**
  + **Bildirim Gösterimi**: Sonuç hesaplandıktan sonra, kullanıcıya kazanan veya kaybeden hakkında bildirim gösterilir. Aynı zamanda skorlar ve genel oyun durumu hakkında bilgi verilir.
  + **Görev Listeleme**: Sonuçlar, kullanıcının oyun geçmişinde görüntülenir. Kazanan ve kaybeden seçimler listelenir.
* **Yönetim Alt Sistemi:**
  + **Görev İşleme**: Sonuç hesaplandığında, kazanan ve kaybeden doğru şekilde belirlenir. Sistem, sonuca göre oyun akışını yönetir.
  + **Kullanıcı Kimlik Doğrulama**: Her kullanıcının seçimleri doğru şekilde eşleşir ve sonuca etki eder.
* **Model Alt Sistemi:**
  + **Kullanıcı Verisi Yönetimi**: Oyun sonuçları kaydedilir ve gelecekteki karşılaştırmalar için kullanılır. Kullanıcı verisi, sonucu hesaplayan model tarafından işlenir.
  + **Bildirim Depolama ve Erişim**: Sonuçlar ve ilgili bildirimler, sisteme kaydedilir ve gerektiğinde erişilir.

## 4.Düşük Seviyeli Tasarım

**1. Modüller ve Kütüphaneler**

**Açıklama:**

Proje için kullanılacak Python kütüphaneleri ve modülleri aşağıdaki gibidir:

* **OpenCV (cv2):** Görüntü işleme ve kameradan görüntü almak için kullanılır.
* **Mediapipe:** El hareketlerini tespit etmek ve taş-kağıt-makas hareketlerini tanımak için kullanılır.
* **NumPy:** Görüntü verilerini ve koordinatları işlemek için kullanılır.
* **Random:** Bilgisayarın rastgele bir taş, kağıt veya makas seçmesini sağlar.

**2. Sınıf ve Fonksiyon Tanımları**

**2.1 El Tespiti ve Tanıma**

* **El Tespiti Sınıfı (HandDetector)**
  + **Amaç:** Kamera görüntüsünden elin varlığını algılamak ve belirli hareketlerin (taş, kağıt, makas) tanınmasını sağlamak.
  + **Özellikler:**
    - mediapipe modelini başlatma.
    - El pozisyonlarını algılamak için fonksiyon.
  + **Metotlar:**
    - detect\_hands(image): Görüntüdeki elleri algılar.
    - classify\_hand(shape): Tespit edilen el hareketini sınıflandırır (taş, kağıt, makas).

**2.2 Oyun Mantığı**

* **Oyun Sınıfı (RockPaperScissors)**
  + **Amaç:** Oyuncunun seçimini algılamak, bilgisayarın seçim yapmasını sağlamak ve oyunun sonucunu hesaplamak.
  + **Özellikler:**
    - Oyuncunun seçimi (taş, kağıt, makas).
    - Bilgisayarın rastgele seçimi.
    - Skor takibi.
  + **Metotlar:**
    - get\_computer\_choice(): Bilgisayarın rastgele bir seçim yapmasını sağlar.
    - determine\_winner(player, computer): Oyuncu ve bilgisayar seçimlerine göre sonucu belirler.

**2.3 Kullanıcı Arayüzü**

* **Arayüz Fonksiyonları**
  + **Amaç:** Kullanıcıya kamera görüntüsünü ve oyun durumunu göstermek.
  + **Fonksiyonlar:**
    - draw\_text(image, text, position): Görüntü üzerine metin ekler.
    - display\_score(player\_score, computer\_score): Oyuncu ve bilgisayarın skorlarını ekrana yazar.
    - overlay\_image(image, choice): Bilgisayarın seçimini görsel olarak ekrana ekler.

**3. Veri Akışı**

**Açıklama:**

Oyun sırasında veri akışı şu şekilde gerçekleşir:

1. **Kamera Verisi Alımı:**
   * OpenCV aracılığıyla bilgisayarın kamerasından sürekli görüntü alınır.
2. **El Tespiti:**
   * Görüntü, HandDetector sınıfı kullanılarak analiz edilir ve oyuncunun el hareketi (taş, kağıt, makas) tanımlanır.
3. **Bilgisayar Seçimi:**
   * Bilgisayar rastgele bir seçim yapar.
4. **Sonuç Hesaplama:**
   * Oyuncunun ve bilgisayarın seçimleri determine\_winner() fonksiyonuna gönderilir ve kazanan belirlenir.
5. **Görüntüleme:**
   * Kamera görüntüsü, oyuncunun seçimi, bilgisayarın seçimi ve skorlar ekranda gösterilir.

**4. Oyun Akışı**

**Açıklama:**

Oyun sırasındaki adımlar detaylandırılır:

1. Kamera açılır ve kullanıcıdan hareket algılanır.
2. **Algılanan Hareketler:**
   * "Taş" için yumruk.
   * "Kağıt" için açık el.
   * "Makas" için V şeklindeki el işareti.
3. Bilgisayar rastgele seçim yapar.
4. Kazanan, oyunun kurallarına göre belirlenir:
   * Taş > Makas
   * Makas > Kağıt
   * Kağıt > Taş
5. Sonuç ve skor ekranda güncellenir.

**5. Veri Yapıları**

**Açıklama:**

Kullanılan veri yapıları ve formatlar:

* **Liste:** Bilgisayarın seçimleri için (choices = ['rock', 'paper', 'scissors']).
* **Sözlük:** Hareket sınıflandırması için ({'rock': 0, 'paper': 1, 'scissors': 2}).
* **Tam Sayı (int):** Oyuncu ve bilgisayarın skorları için.

**6. Algoritmalar**

**El Hareketi Sınıflandırma Algoritması:**

* Mediapipe koordinatları kullanarak:
  + Avuç içi genişliği, parmak uçlarının pozisyonları gibi ölçütlerle hareket tanımlanır.
  + Örnek: Eğer tüm parmaklar açıksa -> "Kağıt" olarak sınıflandırılır.

**Sonuç Belirleme Algoritması:**

* Oyuncu ve bilgisayar seçimleri karşılaştırılır:

**7. Performans ve Optimizasyon**

**Açıklama:**

* Kamera görüntüsü analizi, her döngüde gerçekleştirileceği için performans kritik bir unsurdur. OpenCV'nin optimize edilmiş metotları kullanılmalıdır.
* **FPS (Frames Per Second):** Algoritmanın gerçek zamanlı çalışması için döngü hızı optimize edilmelidir.
* Gereksiz görüntü işlemleri minimize edilerek kullanıcı deneyimi iyileştirilir.

**8. Test Senaryoları**

**Açıklama:**

* **Doğru Hareket Algılama:** Taş, kağıt, makas hareketleri doğru algılanıyor mu?
* **Kazanan Belirleme:** Oyuncu ve bilgisayarın seçimlerine göre doğru sonuç veriliyor mu?
* **Kullanıcı Arayüzü:** Görsel öğeler doğru görüntüleniyor mu?
* **Performans:** Program düşük ışık koşullarında da sorunsuz çalışıyor mu?