

**Marmara Üniversitesi**

**Teknoloji Fakültesi**

**Bilgisayar Mühendisliği**

Bilgisayar Ağlarına Giriş

**-**

Pygame ve Soket Kullanarak Online Oyun Geliştirme Proje Raporu

Grup 7 – Grup Üyeleri

|  |  |  |
| --- | --- | --- |
| **Simay Ünal** | **Emre Şen** | **Emre Yanalak** |
| **170424822** | **170423044** | **170423056** |
| [**github.com/S1mayUnal**](https://github.com/S1mayUnal) | [**github.com/emresen12**](https://github.com/emresen12) | [**github.com/EmreYnlk**](https://github.com/EmreYnlk) |

**İÇİNDEKİLER**

İçindekiler

[**Projemiz:** 3](#_Toc198583533)

[**Kullanılan Teknolojiler:** 3](#_Toc198583534)

[**Oyun İçi Görseller** 4](#_Toc198583535)

[**Giriş** 6](#_Toc198583536)

[**1.Arayüz** 6](#_Toc198583537)

[**2.Main.py** 6](#_Toc198583538)

[MainMenu Sınıfı: 7](#_Toc198583539)

[ChooseModeScreen Sınıfı: 8](#_Toc198583540)

[HistoryScreen Sınıfı: 8](#_Toc198583541)

[PongApp Sınıfı: 9](#_Toc198583542)

[İnterface.kv: 9](#_Toc198583543)

[<MainMenu>: 10](#_Toc198583544)

[<HistoryScreen>: 11](#_Toc198583545)

[<ChooseModeScreen>: 12](#_Toc198583546)

[**3. Server.py** 13](#_Toc198583547)

[pongDTO Sınıfı: 13](#_Toc198583548)

[Game Sınıfı: 14](#_Toc198583549)

[get\_game\_dto(game\_id) Fonksiyonu: 15](#_Toc198583550)

[get\_game(game\_id) Fonksiyonu: 15](#_Toc198583551)

[update\_game\_dto(dto) Fonksiyonu: 15](#_Toc198583552)

[update\_game\_state(dto) Fonksiyonu: 16](#_Toc198583553)

[get\_game\_player\_id() Fonksiyonu: 17](#_Toc198583554)

[threaded\_client(conn, game\_id, player\_id) Fonksiyonu: 18](#_Toc198583555)

[**4.Client.py** 20](#_Toc198583556)

[pongDTO Sınıfı: 20](#_Toc198583557)

[Bat sınıfı: 21](#_Toc198583558)

[Ball Sınıfı: 22](#_Toc198583559)

[update\_bat\_ball Fonksiyonu: 22](#_Toc198583560)

[Ana Oyun Döngüsü ve İstemci Bağlantısı: 23](#_Toc198583561)

# **Projemiz:**

Bu proje, Python programlama dili kullanılarak geliştirilen çift oyunculu bir Pong oyunudur. Oyun, TCP/IP protokolü üzerinden haberleşen bir istemci-sunucu mimarisi ile çalışmaktadır. Amaç, iki oyuncunun ağ üzerinden bağlanarak gerçek zamanlı olarak Pong oyununu oynamalarını sağlamaktır. Sunucu tarafında, gelen istemci bağlantıları yönetilir, her oyuncuya bir oyun ve oyuncu kimliği atanır. Oyun içindeki topun hareketi, raketlerin konumu ve skorlar gibi bilgiler sunucu tarafından merkezi olarak güncellenir ve her istemciye senkronize bir şekilde iletilir. Bu veri aktarımı, PongDTO isimli özel bir veri sınıfı (Data Transfer Object) aracılığıyla gerçekleştirilir. İstemci tarafında, oyuncular Pygame kütüphanesi ile hazırlanmış görsel arayüz üzerinden oyunu oynarlar. Oyuncular klavye girdileriyle raketlerini kontrol eder ve sunucudan gelen güncel oyun durumu doğrultusunda topun hareketi ve skorlar gibi öğeleri gerçek zamanlı olarak takip ederler.

Projenin temel özellikleri şunlardır:

* Gerçek zamanlı iki oyunculu çevrim içi Pong oyunu
* TCP soket bağlantısı ile oyuncular arası iletişim
* Pygame arayüzü ile kullanıcı dostu görseller
* Raket ve top çarpışma hesapları
* Neon tema ve etkili arka plan müzikleri

# **Kullanılan Teknolojiler:**

* Python
* Socket Kütüphanesi
* Thread Kütüphanesi
* Pygame Kütüphanesi
* Pickle

# **Oyun İçi Görseller**

metin, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

bina, gökdelen, ufuk çizgisi, şehir manzarası içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

metin, ekran görüntüsü, sanat içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

ekran görüntüsü, çizgi film, Animasyon, bilgisayar oyunu içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

metin, su, ekran görüntüsü, tasarım içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

# **Giriş**

## **1.Arayüz**

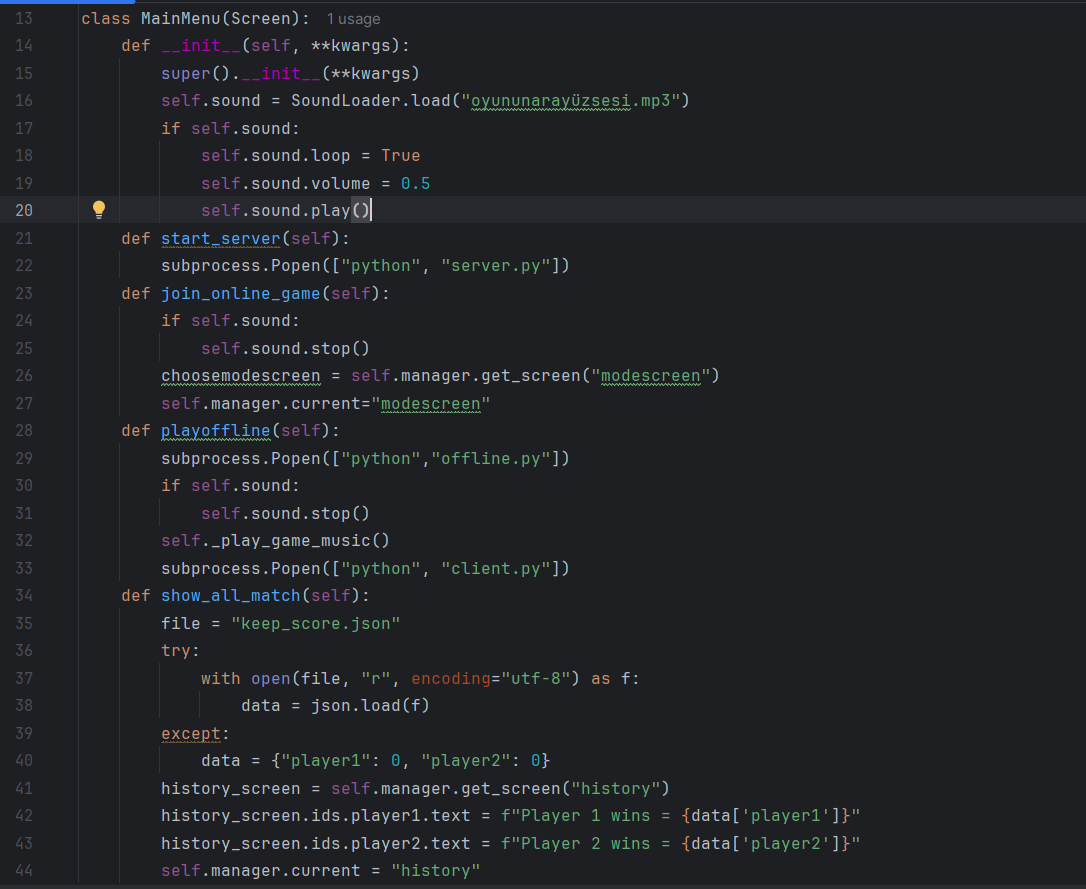
Kullanıcı arayüzü için Kivy kütüphanesi kullanılmıştır. Kullanıcı, oyunu farklı modlarda başlatabilmekte, sunucu kurabilmekte ve önceki maç sonuçlarını görebilmektedir.

## **2.Main.py**

Amaçları:

* Uygulamanın farklı ekranlarını yönetmek
* Oyun modlarını başlatmak
* Sunucu/istemci yapısını başlatmak (subprocess ile)
* Arka plan müziği kontrol etmek
* Skorları JSON dosyasından okumak

### MainMenu Sınıfı:



* Ana menüyü temsil eder.
* Arka plan müziği açılır (oyununarayüzsesi.mp3)
* Dört ana işlev sunar:
* **Start Server →** server.py çalıştırılır
* **Join Online Game** → Oyun modu seçme ekranına geçilir
* **Play Offline** → Hem offline oyun hem de online istemci başlatılır
* **History** → Skorlar JSON dosyasından okunarak görüntülenir

### ChooseModeScreen Sınıfı:

metin, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Oyun modlarını seçmek için kullanılır.
* Üç farklı mod sunar:

1. Normal Mode → client.py
2. Galatasaray Mode → clientgs.py
3. Fenerbahçe Mode → clientfb.py

* Her mod seçildiğinde ilgili müzik çalınır.

### HistoryScreen Sınıfı:

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Skorları JSON dosyasından okuyarak kullanıcıya gösterir.
* keep\_score.json dosyasından "player1" ve "player2" galibiyet sayıları alınır.

### PongApp Sınıfı:



* Ekran yöneticisi (ScreenManager) ile uygulama başlatılır.
* menu, history, modescreen ekranları tanımlanır ve yüklenir.

### İnterface.kv:

Kivy dilinde yazılmıştır.

Arka plan resmi (projearkaplanresmi.jpg) kullanılmıştır.

Üç ana ekran tanımlanır:

* <MainMenu>
* <HistoryScreen>
* <ChooseModeScreen>

Butonlar, konum ve işlev olarak tasarlanmıştır.

### <MainMenu>:

metin, ekran görüntüsü, yazılım, ekran, görüntüleme içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Ana menü ekranında FloatLayout kullanılır.
* Kullanıcı Join Online Game Butonu ile Online olarak bir oyuna bağlanır.
* Kullanıcı internet bağlantısı olmadan yerel oyun başlatmak isterse bu butona tıklar.
* Start Server Butonu sunucuyu başlatmak için kullanılır server.py dosyasını arka planda çalıştırır.
* History butonu kullanıcının geçmiş maçlardaki galibiyet sayılarını görüntülemesini sağlar.

### <HistoryScreen>:

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

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* player1 ve player2 etiketleri, Python tarafından dinamik olarak güncellenir.
* Go Back butonu ile ana menüye dönülür.

### <ChooseModeScreen>:

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

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Kullanıcı, klasik Pong oynanışı tercih ettiğinde Normal Mode butonuna tıklar.
* Kullanıcı Galatasaray Temalı oynamak istediğinde Galatasaray Mode butonuna tıklar.
* Kullanıcı Fenerbahçe Temalı oynamak istediğinde Fenerbahçe Mode butonuna tıklar

## **3. Server.py**

Amaçları:

* TCP soketleri kullanarak iki oyuncunun birbirine bağlanmasını sağlamak
* Pong oyununun durumunu eş zamanlı olarak her iki oyuncu arasında senkronize etmek
* Oyuncuların giriş bilgilerini yönetmek
* Oyun nesnelerinin (top, raketler) durumunu güncellemek
* İstemciler arasında veri alışverişi sağlamak

### pongDTO Sınıfı:

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Ağ üzerinden (örneğin pickle ile) istemcilerle haberleşirken, oyunla ilgili bütün veriler tek bir nesne olarak gönderilir veya alınır.
* Pong oyununda iki oyuncunun pozisyonlarını, topun konumunu, hareket bilgilerini ve skoru barındıran veri taşıyıcısıdır.

Game Sınıfı: metin, ekran görüntüsü, yazılım içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Her biri iki oyuncudan oluşan tek bir oyunun durumunu yönetir.
* Pong oyununun sunucudaki temsilidir.
* Oyuncuları tanımlar, oyunun başlangıç değerlerini ayarlar ve PongDTO aracılığıyla oyunun durumunu takip eder.
* Sunucu, birden fazla Game nesnesi oluşturarak aynı anda birçok oyunu kontrol edebilir.

get\_game\_dto(game\_id) Fonksiyonu: metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Her oyun için oyuncu konumları, top pozisyonu, hız, yön, skor gibi bilgileri içeren PongDTO nesnesine erişim sağlar
* İstemciler arası veri güncellenebilir ve paylaşılabilir.

### get\_game(game\_id) Fonksiyonu:

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* game\_ids listesindeki tüm Game nesnelerini dolaşır.
* game.game\_id değeri, parametre olarak verilen game\_id ile eşleşirse o Game nesnesini döndürür.

### update\_game\_dto(dto) Fonksiyonu:

metin, ekran görüntüsü, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

Oyuncunun y (dikey) konumunu güncellemek için kullanılır.

### update\_game\_state(dto) Fonksiyonu:

metin, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

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

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Oyun sırasında her oyuncunun pozisyonu, puanı, topun konumu gibi **dinamik verileri güncelleyerek** oyunun devamlılığını sağlar.
* Oyunculardan alınan verileri (DTO) işler.
* Çarpışma kontrolü (top-raket veya top-duvar) yapar.
* Oyun bitiş durumu veya skor kontrolü yapabilir.
* Oyun içi objelerin (raket, top, skor) konumlarını günceller.
* Güncellenmiş durumu istemcilere geri gönderir.

### get\_game\_player\_id() Fonksiyonu:

metin, ekran görüntüsü, yazılım içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Oyunu başlatmak için yeni bir bağlantı kabul edildiğinde çağrılır ve oyun kimliği ve oyuncu kimliğine ihtiyaç duyar.
* Yeni oyuncu için kullanılabilir oyun kimliğini ve oyuncu kimliğini sağlar.
* Bir oyuncunun başka bir oyuncuyu beklediği bir oyun varsa, o oyun kimliği ve oyuncu kimliği verilir.
* Bir kullanılabilir yuvası olan mevcut bir oyun yoksa, yeni bir oyun oluşturur ve kimlikleri döndürür.

### threaded\_client(conn, game\_id, player\_id) Fonksiyonu:

metin, ekran görüntüsü, yazılım içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Her istemci bağlantısı için ayrı bir thread içerisinde çalışarak iletişimi yönetir.
* Oyun kuyruğundan oyun için veri aktarım nesnesini alır.
* Oyunda her iki oyuncu da varsa ve başlatma işareti etkinse, oyun durumunu günceller.
* Herhangi bir nedenle istemciyle bağlantı kesilirse oyun durumunu sıfırlar.
* Her iki oyuncu da ayrıldıysa, oyunu oyun kuyruğundan kaldır.

## **4.Client.py**

**Amaçları:**

* Sunucuya bağlanarak oyuna katılmak
* Oyuncudan alınan giriş verilerini (hareket gibi) sunucuya göndermek
* Sunucudan gelen PongDTO nesnesiyle oyunun güncel durumunu almak
* Ekranda oyun grafiklerini (raket, top, skor) çizmek ve güncel tutmak
* Kullanıcı ile etkileşim sağlayarak gerçek zamanlı bir oyun deneyimi sunmak
* Sunucudan gelen verilere göre topun ve raketlerin konumunu güncellemek
* Oyuncunun yukarı/aşağı hareketlerini klavye aracılığıyla kontrol etmek

### pongDTO Sınıfı:

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Ağ üzerinden (örneğin pickle ile) istemcilerle haberleşirken, oyunla ilgili **bütün veriler tek bir nesne** olarak gönderilir veya alınır.
* Pong oyununda iki oyuncunun pozisyonlarını, topun konumunu, hareket bilgilerini ve skoru barındıran **veri taşıyıcısıdır**.

### Bat sınıfı:

**metin, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

* Her bir oyuncunun raketini temsil eder.
* Oyuncunun yatay konumu sabit, dikey konumu ise move fonksiyonu ile klavye girişlerine göre değişir.
* draw fonksiyonu ile raketin ekran üzerine çizilmesini sağlar
* add\_point fonksiyonu ile oyuncunun puanı bir artırılır.
* Her oyuncunun skoru ayrı ayrı tutulur ve oyunun gidişatına göre güncellenir

### Ball Sınıfı:

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

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

* Oyunda kullanılan topu temsil eder.
* Başlangıçta rastgele bir yatay ve dikey yön belirlenir (direction\_x, direction\_y).
* Topun konumu (x, y) ve hızı (velocity\_x, velocity\_y) tanımlanır.
* draw fonksiyonu ile top, ekranda daire şeklinde çizilir.
* Topun hareketi ve çarpışma kontrolü sunucu tarafında yapılırken, istemci tarafı sadece gelen verilere göre topun pozisyonunu güncelleyip ekrana yansıtır.

### update\_bat\_ball Fonksiyonu:

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

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

* Sunucudan alınan PongDTO nesnesi ile raketlerin (bat) ve topun (ball) konumunu günceller.
* Oyuncu kimliğine göre kendi raketinin ve rakibin raketinin renklerini ayarlar.
* Her iki oyuncunun x ve y koordinatları güncellenir.
* Topun konumu ve rengi, gelen DTO verilerine göre ayarlanır.
* Bu sayede oyun ekranında doğru pozisyonlar gösterilerek istemcide gerçek zamanlı görsellik sağlanır.

### Ana Oyun Döngüsü ve İstemci Bağlantısı:

metin, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

metin, ekran görüntüsü, yazılım içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.İstemci (client), TCP protokolü kullanarak belirtilen IP ve port ile sunucuya bağlanır.

* Sunucudan gelen ilk PongDTO verisi alınır ve oyuncunun kimliği belirlenir.
* Oyuncunun ve rakibin kimlikleri belirlendikten sonra iki raket (Bat) ve bir top (Ball) nesnesi oluşturulur.
* update\_bat\_ball fonksiyonu çağrılarak oyun nesnelerinin ilk konumları ayarlanır.
* Pygame ile pencere, yazı tipi ve arka plan görseli hazırlanır.
* Sonsuz bir while döngüsü ile oyun sürekli çalıştırılır; bu döngü FPS’ye göre kontrol edilir.
* Her döngüde ekran güncellenir, ortadaki çizgiler çizilir, raketler ve top yeniden çizilir, skorlar yazdırılır.
* Klavye girişleri ile yukarı (W) ve aşağı (S) hareket algılanır, ESC ile oyun sonlandırılır.
* Oyun sırasında DTO nesnesi sunucuya gönderilir ve güncel DTO tekrar alınır.
* Alınan verilerle update\_bat\_ball tekrar çağrılarak top ve raket konumları güncellenir.
* Oyun kapatılırken bağlantı hataları kontrol edilerek kullanıcıya hata mesajı gösterilir.