



**FATİH  
SULTAN  
MEHMET**  
VAKIF ÜNİVERSİTESİ

---

**Student:**

Name: AYMAN

Surname: SAEID

ID Number: 2221221366

Department:

**Project:**

Topic: Amiral Battı (Battleship)

**Course:**

Name: Bilgisayar Ağları

Instructor: Samet Kaya

## Content

1- Project Topic.....	Error! Bookmark not defined.
2- Tasks Completed During the Project.....	Error! Bookmark not defined.
3- Additional Notes.....	Error! Bookmark not defined.
4- References.....	Error! Bookmark not defined.



**FATİH  
SULTAN  
MEHMET  
VAKIF ÜNİVERSİTESİ**

# 1-Genel Bakış

klasik Battleship (Deniz Savaşı) oyununun çok oyunculu bir versiyonunu Java kullanarak geliştirmektedir. Oyun, bir sunucu ve birden fazla istemci arasında TCP/IP soket bağlantıları üzerinden iletişim kurar. Kullanıcılar gemilerini yerleştirir ve sırayla rakibin gemilerini vurmaya çalışır. Tüm gemileri vuran oyuncu kazanır.

## Ana Özellikler:

- Çok oyunculu lobi sistemi (2'şer kişilik eşleşmeler)
- Gerçek zamanlı atış ve sonuç bildirimi
- Gemilerin rastgele veya manuel yerleştirilmesi
- Oyun durumunun sunucuda senkronize edilmesi
- AWS üzerinde sunucu barındırma desteği

## Kısa Oyun Kuralları:

- ✓ 2 oyuncu sunucuya bağlanır. Her oyuncunun 10x10'luk tahtası vardır.
- ✓ Her oyuncu 5 gemi yerleştirir: (5, 4, 3, 3, 2 birimlik).
- ✓ Gemiler üst üste gelemez, yatay/dikey yerleştirilir.
- ✓ Oyuncular sırayla hücum eder.
- ✓ İsabet durumunda "Vuruldu", tüm parça vurulursa "Gemi batırıldı" mesajı gelir.
- ✓ Tüm gemileri ilk vuran oyuncu kazanır.

# 2-Sistem Mimarisi ve Teknolojiler

Teknoloji	Açıklama
Java	Oyunun istemci ve sunucu tarafı Java ile yazılmıştır.
TCP/IP Soket Programlama	İstemci-sunucu iletişimi için kullanılmıştır.
Swing (GUI)	Kullanıcı arayüzü için Java Swing kütüphanesi tercih edilmiştir.
AWS EC2	Sunucunun uzaktan çalıştırılması için kullanılmıştır.
Threading (Çoklu İş Parçacığı)	Eş zamanlı oyun yönetimi için kullanılmıştır.

## 3-Sınıflar ve Görevleri

### 3.1. Sunucu Tarafı (Server Side)

#### BattleshipServer.java

- **Görevi:** Ana sunucu yapılandırmasını yönetir.
- **Kullanım Amacı:**
  - Belirtilen port üzerinden gelen bağlantıları dinler.
  - Yeni oyuncuları bekleyenler listesine ekler.
  - 2 oyuncu tamamlandığında yeni bir lobi oluşturur.
- **Thread Kullanımı:** Her yeni oyuncu için ayrı bir thread başlatılır.

#### GameLobby.java

- **Görevi:** İki oyuncunun bir arada oynadığı oyun alanını yönetir.
- **Kullanım Amacı:**
  - Oyuncuların hamlelerini işler.
  - Sıra mekanizmasını kontrol eder.
  - Oyunun bitip bitmediğini kontrol eder.
- **Thread Kullanımı:** Oyun durumu tek bir thread tarafından senkronize edilir.

#### PlayerHandler.java

- **Görevi:** Her bir oyuncunun bağlantısını yönetir.
- **Kullanım Amacı:**
  - İstemciden gelen mesajları işler (SHOT, READY, SHIPS).
  - Gemilerin yerleştirilmesini kontrol eder.
  - Atış sonuçlarını ilgili lobiye iletir.
- **Thread Kullanımı:** Her oyuncu için ayrı bir thread çalıştırılır.

### 3.2. İstemci Tarafı (Client Side)

#### BattleshipClient.java

- **Görevi:** Sunucuyla iletişimi sağlar.
- **Kullanım Amacı:**
  - Sunucuya bağlanır ve mesajları iletir.
  - Gelen mesajları işleyerek GUI'yi günceller.
  - Oyun durumunu takip eder (sıra, kazanan vb.).
- **Thread Kullanımı:** Sunucudan gelen mesajları dinlemek için ayrı bir thread kullanır.

#### BattleshipGUI.java

- **Görevi:** Kullanıcı arayüzünü yönetir.
- **Kullanım Amacı:**
  - Gemilerin yerleştirilmesini sağlar.
  - Rakip tahtasına tıklanınca atış yapılmasını tetikler.
  - Oyun durumunu gösterir (kimin sırası, vurulan gemiler vb.).
- **Thread Kullanımı:** Swing'in Event Dispatch Thread (EDT) yapısını kullanır.

### 3.3. Ortak Sınıflar (Common Classes)

#### Board.java

- **Görevi:** Oyun tahtasını ve gemilerin konumlarını yönetir.
- **Kullanım Amacı:**
  - Gemilerin yerleştirilmesini kontrol eder.
  - Atışların isabet durumunu hesaplar.
  - Oyunun bitip bitmediğini belirler.

### Cell.java

- **Görevi:** Tahtadaki her bir hücreyi temsil eder.
- **Kullanım Amacı:**
  - Hücrede gemi olup olmadığını tutar.
  - Hücrenin vurulup vurulmadığını kontrol eder.

### Ship.java

- **Görevi:** Gemilerin özelliklerini saklar.
- **Kullanım Amacı:**
  - Gemi boyutunu ve koordinatlarını tutar.
  - Kaç kez vurulduğunu takip eder.

### ShotResult.java (Enum)

- **Görevi:** Atış sonuçlarını tanımlar.
- **Değerler:**
  - **HIT** (İsabet)
  - **MISS** (İska)
  - **SUNK** (Gemi batırıldı)
  - **GAME\_OVER** (Oyun bitti)
  - **ALREADY\_HIT**
  - **INVALID**

## 4- Algoritmalar ve Oyun Kuralları

### 4.1. Oyun Kuralları

#### 1. Gemi Yerleştirme:

- Her oyuncu 4 gemi yerleştirir:
  - **Carrier** (5 birim)
  - **Battleship** (4 birim)
  - **Cruiser** (3 birim)

- **Submarine (2 birim)**

- **Gemiler yatay veya dikey yerleştirilebilir.**
- **Gemiler birbiriyle çakışamaz.**

## **2. Oyun Akışı:**

- **Oyuncular sırayla bir koordinata atış yapar.**
- **İsabet edilirse aynı oyuncu tekrar atış yapar.**
- **İska olursa sıra rakibe geçer.**
- **Tüm gemileri batıran oyuncu kazanır.**

## **4.2. Kullanılan Algoritmalar**

<b>Algoritma</b>	<b>Açıklama</b>
<b>TCP Soket İletişimi</b>	<b>İstemci ve sunucu arasında gerçek zamanlı veri aktarımı.</b>
<b>Thread Pooling</b>	<b>Çoklu oyuncu bağlantıları için thread yönetimi.</b>
<b>Grid-Based Ship Placement</b>	<b>Gemilerin tahtaya yerleştirilmesi için matris kontrolü.</b>
<b>Turn-Based Game Logic</b>	<b>Sıralı hamle sistemi.</b>

## 5- DEMO + AWS CONNECTION

EC2

Dashboard

EC2 Global View

Events

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

Images

AMIs

AMI Catalog

Elastic Block Store

Volumes

Snapshots

Lifecycle Manager

Network & Security

Security Groups

Elastic IPs

Placement Groups

Key Pairs

Network Interfaces

Load Balancing

Load Balancers

Target Groups

Instance summary for i-01b20f564036443f8 (battleshipProject) info

Updated 1 minute ago

Instance ID

i-01b20f564036443f8

IPv6 address

-

Hostname type

IP

Answer private resource DNS name

IPv4 (A)

Auto-assigned IP address

13.48.203.60 [Public IP]

IAM Role

-

IMDSv2

Required

Operator

-

Public IPv4 address

13.48.203.60 [open address]

Instance state

running

Private IP DNS name (IPv4 only)

ip-172-31-40-31.eu-north-1.compute.internal

Instance type

t3.micro

VPC ID

vpc-049d70abde1b479e0

Subnet ID

subnet-02af0ee0de3ba9460

Instance ARN

arn:aws:ec2:eu-north-1:274215480695:instance/i-01b20f564036443f8

Private IPv4 addresses

172.31.40.31

Public DNS

ec2-13-48-203-60.eu-north-1.compute.amazonaws.com [open address]

Elastic IP addresses

-

AWS Compute Optimizer finding

Opt-in to AWS Compute Optimizer for recommendations. [Learn more]

Auto Scaling Group name

-

Managed

false

Details

Status and alarms

Monitoring

Security

Networking

Storage

Tags

Instance details info

AMI ID

ami-0c1ac8a41498c1a9c

Monitoring

disabled

Platform details

Linux/UNIX

AMI name

ubuntu/images/hvm-ssd-gp3/ubuntu-noble-24.04-amd64-server-20250305

Allowed image

-

Termination protection

Disabled

Stop protection

Disabled

Launch time

Sat May 24 2025 19:13:08 GMT+0300 (GMT+03:00) (1 day)

AMI location

amazon/ubuntu/images/hvm-ssd-gp3/ubuntu-noble-24.04-amd64-server-20250305


i-01b20f564036443f8 (battleshipProject)

sg-0fb33ccfadbe1a7 (mysec)

Inbound rules

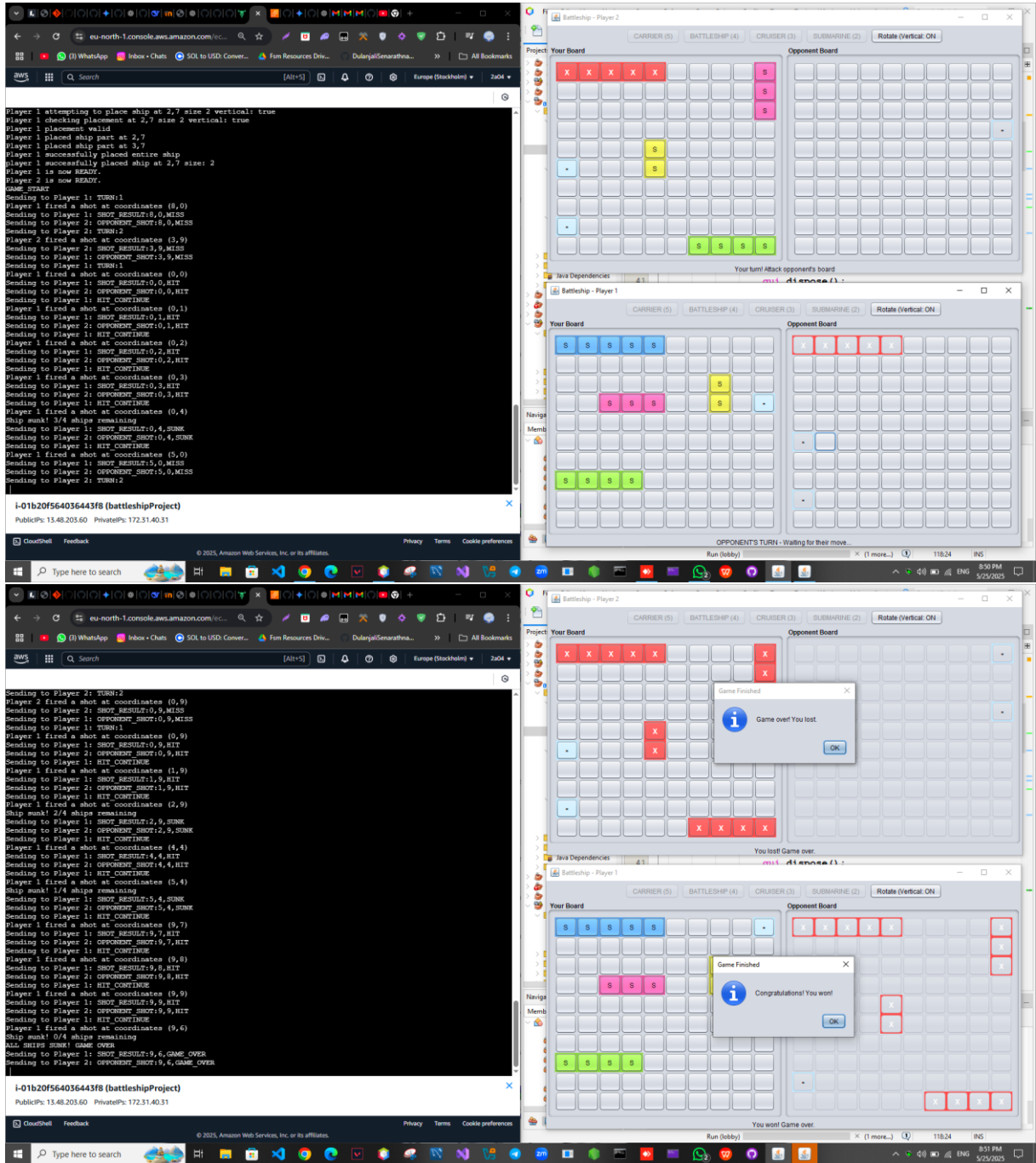
Filter rules

Name	Security group rule ID	Port range	Protocol	Source	Security groups	Description	Launch-wl...
-	sg-r05d12f51a081989ad	22	TCP	0.0.0.0/0	launch-wizard-3	-	⊙
-	sg-r0c85098a29243eb8f	80	TCP	0.0.0.0/0	launch-wizard-3	-	⊙
-	sg-r0b8608de48b9de638	12345	TCP	0.0.0.0/0	mysec	-	-









projenin tüm teknik detaylarını kapsamaktadır. Swing GUI, TCP socket programlama ve çoklu thread yönetimi sayesinde stabil bir çok oyunculu deneyim sunulmuştur. AWS entegrasyonu ile sunucu uzaktan çalıştırılabilir durumdadır.

GITHUB LINK : <https://github.com/aymansaeid/BattleshipGame>