# תרגיל 3 - קורס ג'אווה ואינטרנט

**מגישים:**

**אמיר שביט -**

**יורי ריטבין -**

**אנחנו כאן לכל שאלה, או בקשה**

[**yuri1992@gmail.com**](mailto:yuri1992@gmail.com) **-**

[**ashavit3@gmail.com**](mailto:ashavit3@gmail.com) **-**

## הרצת המשחק

1. **הפעלה:**יש להתקין את BattleShipsWebApp\_war.war על הטומקאט. ניתן לבחור כל Context path.  
   יש לציין כי המשחק נבדק עבור path הבאים: BattleShipsWebApp\_war וגם ships ועובד תקין בשני המקרים.  
   על מנת להפעיל את המשחק, יש להיכנס לדף:

http:// localhost:8080/{context path}

1. **מסך הרשמה:**יש להירשם למשחק, עם שם, כתובת מייל וסיסמה (בת 5 תווים לפחות).  
   עם סיום ההרשמה השחקן יועבר אוטומטית ללוח המשחקים.
2. **לוח המשחקים:**בלוח זה יוצגו רשימת כל המשתמשים הפעילים במערכת בכל זמן נתון, ורשימת המשחקים שהמשתתפים הועלו. כל משתתף יכול:

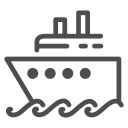
* להעלות משחק: לחיצה על כפתור “*Create Game*” תפתח מסך העלאת משחקים, בה יש לבחור שם למשחק ולבחור את קובץ הגדרות המשחק אותו הוא ירצה לטעון.
* להצטרף למשחק כלשהו (על ידי לחיצה על משולש play ירוק לצד המשחק)
* למחוק משחקים שהעלה בעצמו (משתתף לא יכול למחוק משחק שהועלה על ידי משתמש אחר) באמצעות לחיצה על כפתור X אדום לצד המשחק.
  + משתמש לא יכול למחוק משחק פעיל או שנרשמו אליו שחקנים כלשהם.
* להתנתק באמצעות לחיצה על “*Logout*” בפינה ימנית עליונה

1. **מסך משחק:**עם הצטרפות משתתף למשחק, הוא יועבר אוטומטית למסך המשחק, בו ימתין לתחילת המשחק.  
   אם הוא שחקן ראשון, עליו להמתין להצטרפות שחקן נוסף. עד לתחילת במשחק המשתתף רשאי לפרוש (לחיצה על "*Resign Game*"). אם הוא השחקן השני המשחק יחל מיידית.

* במהלך המשחק יוצגו לכל שחקן לוח הכלים שלו ולוח התקיפות שלו, כמו גם רשימת התורות שביצע וסיכום ביצועים שלו (פירוט בהמשך).
* בתורו, יכול שחקן לבחור לפרוש מהמשחק (לחיצה על "*Resign Game*"). במקרה כזה יוגדר השחקן היריב כמנצח טכני.
* כמו כן, ניתן לנהל שיחת צ׳אט בין 2 השחקנים. את הצ׳אט ניתן לקפל אם הוא מסתיר את הלוח.

## מהלך המשחק

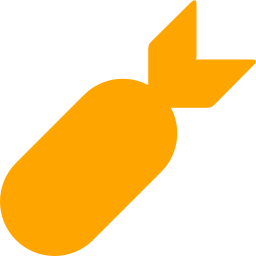
#### My Ships

מציג את הצוללות של השחקן הנוכחי, מציג היכן הן ממוקמות ואת הפגיעות אשר הן ספגו מהשחקן השני. צוללת מסומנת באייקון  , צוללת פגועה מסומנת ב , ומוקשים מסומנים ב-.

במשחקים עבורם הוגדרו שימוש במוקשים, יופיעו מעל הלוח אייקוני המוקשים בהתאם למספר המוקשים שנותרו לשחקן. בתורו, יכול שחקן להציב מוקש על ידי גרירת אחד המוקשים למשבצת הרצויה בלוח הספינות שלו.

#### Attacks Board

מציג את לוח התקיפות של השחקן הנוכחי. תקיפה מבוצעת על ידי לחיצה על המשבצת הרצויה בלוח התקיפות.

תקיפות מוצלחות מסומנות באייקון  החטאות מסומנות באייקון . תקיפה מוצלחת תזכה את השחקן בתור נוסף.

#### Statistics

חלון המידע של המשחק, מציג את הסטטיסטיקות העדכניות של השחקן הנוכחי: ניקוד, תורות, פגיעות והחטאות וזמן ממוצע סיום תור.

#### History

חלון המציג את היסטוריית המהלכים שהשחקן הנוכחי ביצע.

לכל תור מוצג הפעולה אותה ביצע (תקיפה או הצבת מוקש), ומשבצת בה בוצעה הפעולה.

#### Chat

תיבת שיחה עם היריב.

## בונוסים שמומשו

1. **צ׳אט**

אפשרות לשחקנים לקיים chat ביניהם תוך כדי התנהלות המשחק עצמו. הצ'ט מתבצע בין השחקנים הרשומים למשחק בלבד.

## ארכיטקטורת המשחק

### Modules:

1. BattleShipsMain - Game logic and engine, as developed in previous tasks
2. BattleShipsWebApp - The web gui, web client logic, and graphic files.

### BattleShipsMain Module Architecture:

#### Packages

1. Descriptor  
   Have the auto generated class to support JAXB methods. By design descriptor should reflect the xsd we expecting to get when loading the game.
2. Engine
   1. Model - Contain all POJO data models necessary for the game
   2. Managers - Contain all the logic of battle ships game
      1. Game - the core logic of battle ships game
      2. Multi - additional logic relevant for multi player apps (i.e. web app)
   3. Exceptions - Contains all the exception we might throw during the game.
3. Runners  
   Contain the VIEW layer, including the Console View and JavaFx View.

#### New Classes

1. engine.model.multi.Match - Data model for a loaded game by user, including it’s game manager (from ex 2), participating users, and more meta data. This object helps the reuse of game boards.
2. engine.model.multi.User - Data model for an enrolled user (name, email, pass, etc.)
3. engine.managers.multi.MatchManager - Responsible for managing all loaded matches, holds a list of available matches, add or remove matches, etc.
4. engine.managers.multi.UserManager - Responsible for managing all enrolled users, holds a list of active users, and add/remove users to the app.
5. engine.managers.MatchException - A base class for exceptions related for matches. Subclasses define the error itself, i.e. MatchNameTakenException, etc.
6. engine.managers.UserException - A base class for exceptions related for enrolled users. Subclasses define the error itself, i.e. UserNameTakenException, UserNotFoundException, etc.

### BattleShipsWebApp Module Architecture:

#### Packages

1. Constants
2. Models - Contain all POJO data models necessary for the game
3. Servlets - Contains all servlets used for the web app. There are 2 types of servlets:
   1. Redirect Servlets - Used to move users between screens according to their state
   2. API Servlets - Used to perform actions, returning json formatted data to the client
4. Utils - Util classes to help servlets perform logic and actions.

#### New Classes

1. model.UserForJson - Data model wrapping a User object, as a response to clients, easily exposing only relevant data the client needs.
2. model.MatchStatus - Enum for match status (waiting for players to join, in progress, etc.).
3. model.MatchMetaForJson - Data model wrapping a Match object, as a response to clients, easily exposing only relevant data the client needs.
4. model.GameStatusObj - Data model wrapping a Game object (from ex 2), as a response to clients, easily exposing only relevant data the client needs.
5. model.GameStatus - Enum for single game state (who’s turn, who won, etc.).
6. model.GameStatisticsObj - Data model exposing statistics at end of game.
7. model.Chat - Data model to hold an entire chat between players during a game.
8. model.ChatMessage - Data model representing a single chat message.
9. utils.SessionUtils - Utility class. Helper functions to read/write session related properties to http requests
10. utils.ServletUtils - Utility class. Helper functions for servlets to hold and return registered instances of managers from the Main module
11. utils.ChatManager - Responsible for managing chats in active games.
12. servlets.APIUserServlet - Http resource for user related operations, such as login and retrieving list of active users
13. servlets.APISessionServlet - Http resource for getting current logged in user in client
14. servlets.APIGamesHubServlet - Http resource for match related management, such as pulling a single match or a list of all uploaded matches, adding or removing matches, joining games, etc.
15. servlets.APIGameServlet - Http resource for game operations, such as getting game status, leaving a game, playing turns, etc.
16. servlets.APIChatServlet - Http resource for posting and pulling chat conversations

* There are some more servlets (LoginServlet, LogoutServlet, MatchHubServlet, GameServlet) that are responsible for forwarding users between the different screen.

#### BattleShipsWebApp Web Packages

1. Pages - HTML and JSP files
2. Static - including JavaScript, CSS and media files

#### Client Side

Each page was implemented as a self contained javascript class responsible for all the functionality of the page.

1. Vendor - Vendor client side scripts and code
   1. Bootstrap.js
   2. jquery.js
2. Common.js - Common utils used in all pages of the app.
3. Gamelist.js - game list page client side code.
4. Match.js - single match page client side code
5. Signup.js - signup form page client side code.
6. Style.css - Overwrite to bootstrap and additional styling.

### Third Parties

1. Gson - for Servlet JSON responses
2. jQuery - Javascript library, used specially for DOM manipulation.
3. Bootstrap - UI Framework, used as the design theme of the app.