Battleship (=Bataille Navale)

<u>Guessing game</u>

Med Nader Hachana
Med Aziz Ben Dhiab
Yassine El FKy

Introdution

- **Battleship** (also **Battleships** or **Sea Battle)** is a strategy type guessing game for two players. It is played on ruled grids (paper or board) on which each player's fleet of ships (including battleships) are marked. The locations of the fleets are concealed from the other player. Players alternate turns calling "shots" at the other player's ships, and the objective of the game is to destroy the opposing player's fleet.

Class Diagram

Player	Board	Boat
-Turn: bool -Victory: bool -Inventory: list<Boat> -YourBoard: Board -OpponentBoard: Board	-NbrBoats: int -BoatList: list<Boat> -Grid: Array[Nmax1][Nmax2]<struct (includeB: bool, hit: bool)>	-width: int -length: int -Coordinates: Array<struct(x: int, y: int, hit: bool)> -Rotation: bool -State: string
→Player() →SetTurn() →AddBoat(Boat) →ShowInventory() →SelectBoat() →PlaceBoat(int x, int y) →Winner()	→Board() →Strike(int x, int y)	→Boat(int, int) →RotateBoat() →Update(int x, int y)

Classes

Class1:

- Name of the class: **Boat**
- Desciption:

  This class represents a Boat.

- Methods:
  - Boat(int, int): Parameterized class Constractor that takes Length and width as parameters.
  - RotateBoat(): Position boat vertically or horizontally.
  - Update(int x, int y): Update boat status after every strike.

- Attributs:
  - Width: Represents a boat's width.
  - Length: Represents a boat's Length.
  - Coordinates: Represents the occupied cells on the board.
  - Rotation: Represents a boat's rotation(Horizental/Vertical).
  - State: Represents a boat's state (Hit/Unhit/Dead).

Class2:

- ➢ Name of the class: **Board**
- ➢ Desciption:

This class represents a Board indicating each cells' state.

➢ Attributs:

- NbrBoats: Represents the number of boats placed on the board.
- BoatList: Represents the list of boats placed on the board.
- Grid: Represents all the cells of the board including their state.

➢Methods:

- Board(): Class Constractor.
- Strike(int x, int y): Strike a specific cell in the Board (Update cell status).

Class3:

➢ Name of the class: **Player**

➢ Desciption:

Represents a player.

➢Attributs:

- Turn: Indicates if the player is ready to play.
- Victory: Indicates if the player has won
- Inventory: Represents the boats yet to be placed on the board
- YourBoard: Represents the player's board
- OpponentBoard: Represents the opponent's board

➢Methods:

- Player(): Class constractor
- SetTurn(): Indicates which player can strike
- AddBoat(Boat): Add boat to the inventory
- ShowInventory(): Show the boats yet to be placed
- SelectBoat(): Select a boat to place
- PlaceBoat(int x, int y): Place boat from the inventory on the player's board
- Winner(): Stops the game in case a player wins

Functionalities

<u>Basic functionalities:</u>

→ Represents the **Fundamental Functions.**

-Include the default boats.

-Select first player.

-Allows the player to place his boats (Select/Rotate/Place).

-Allows the player to choose a cell on the opponent’s board to hit.

-Alert Both players of the strike’s result.

-End the game when all a player’s boats are dead.

-Show the winning player.

Functionalities

<u>Additional functionalities:</u>

→ Represents the **Extra Functions**.

-Include timer (Pass turn/auto strike/instant defeat).

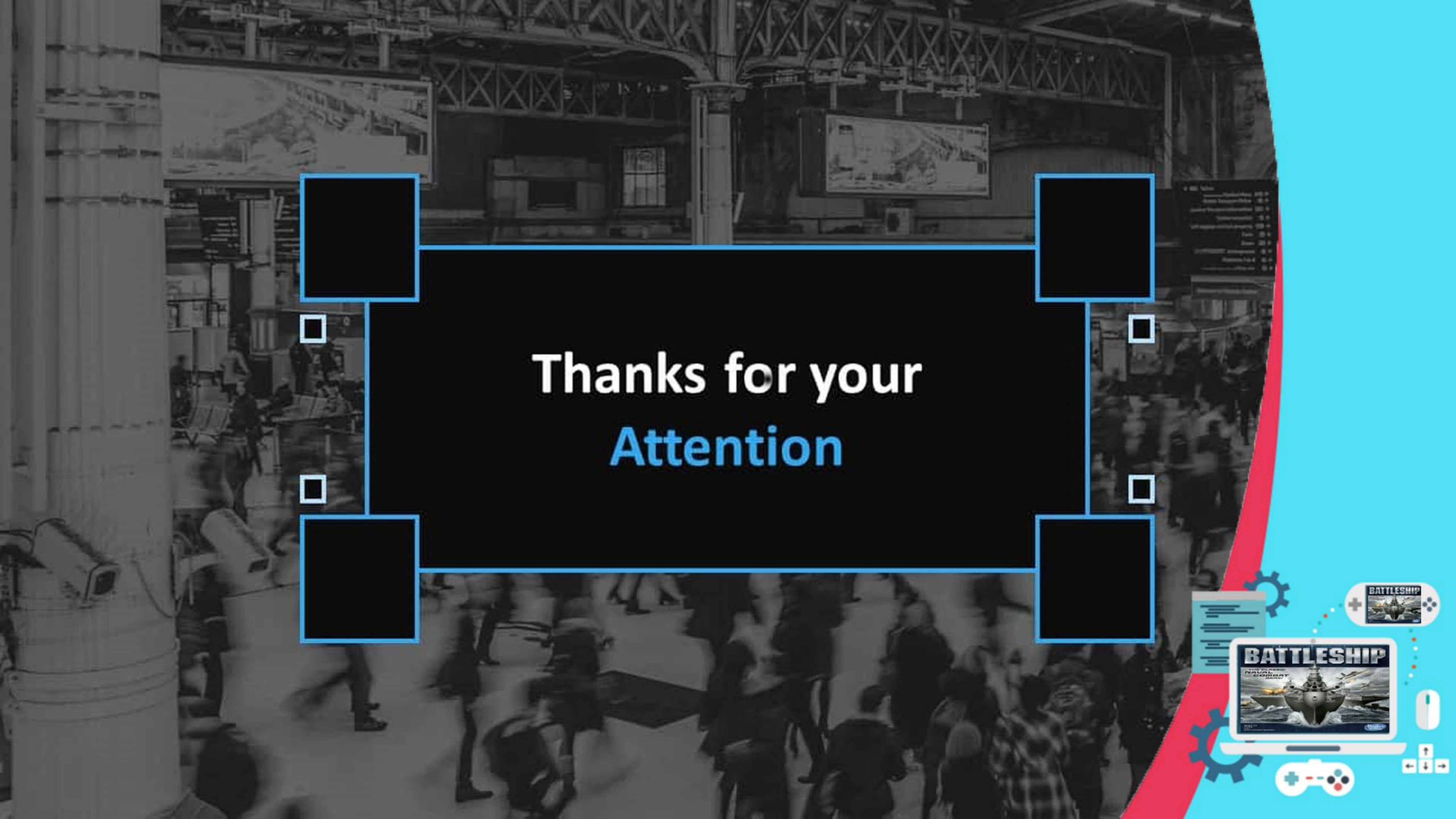
-Revert Boats placement.

-Customize boats.

-Customize the board’s shape.

-Show player’s boats and their state.

-Show stats after the game ends(Accuracy/hit streak/Survived boats…).



Thanks for your Attention