

# Software Engineering Project Report

[1]

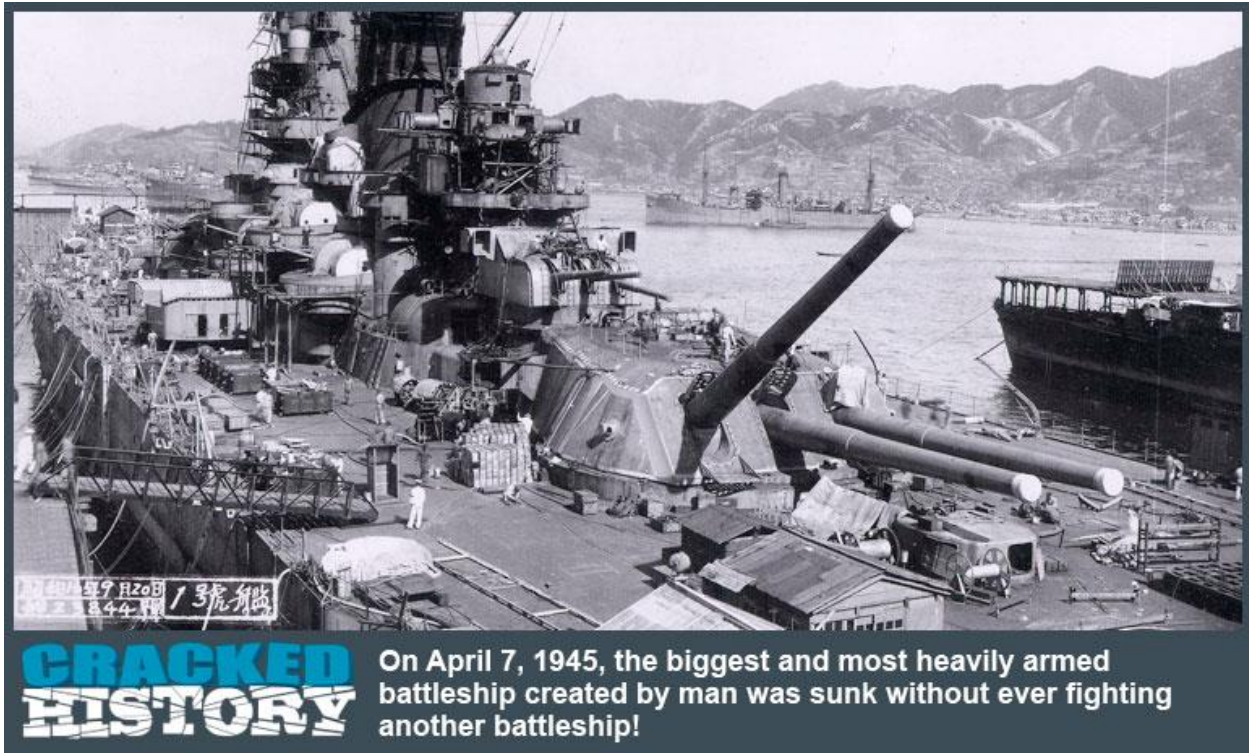


Figure 1 Title Picture of Yamato Ship

## Project Development Report for Age of Battleships

Prepared by  
Group 3 - Abraheem Irheem, Bhavya Mahajan,  
Mukund Subakanthan, Prabhjot Singh  
CS 440  
University of Illinois at Chicago

September 2015

## Table of Contents

	List of Figures.....	7
	List of Tables.....	9
I	Project Description .....	10
1	Project Overview .....	10
2	The Purpose of the Project .....	10
2a	The User Business or Background of the Project Effort.....	10
2b	Goals of the Project .....	11
2c	Measurement.....	11
3	The Scope of the Work.....	11
3a	The Current Situation.....	11
3b	The Context of the Work.....	11
3c	Work Partitioning.....	12
3d	Competing Products .....	12
4	The Scope of the Product .....	12
4a	Scenario Diagram(s) .....	13
4b	Product Scenario List .....	16
4c	Individual Product Scenarios .....	16
5	Stakeholders .....	17
5a	The Client.....	17
5b	The Customer .....	17
5c	Hands-On Users of the Product .....	17
5d	Priorities Assigned to Users .....	20
5e	User Participation.....	20
5f	Maintenance Users and Service Technicians.....	20
5g	Other Stakeholders .....	20
6	Mandated Constraints.....	21
6a	Solution Constraints.....	21
6b	Implementation Environment of the Current System .....	21
6c	Partner or Collaborative Applications .....	22
6d	Off-the-Shelf Software .....	22
6e	Anticipated Workplace Environment .....	22
6f	Schedule Constraints.....	22
6g	Budget Constraints .....	23
7	Naming Conventions and Definitions .....	23
7a	Definitions of Key Terms .....	23
7b	UML and Other Notation Used in This Document .....	23

	7c	Data Dictionary for Any Included Models .....	23
8		Relevant Facts and Assumptions.....	24
	8a	Facts .....	24
	8b	Assumptions .....	25
II		Requirements .....	26
9		Product Use Cases .....	26
	9a	Use Case Diagrams .....	26
	9b	Product Use Case List .....	26
	9c	Individual Product Use Cases .....	27
10		Functional Requirements.....	34
11		Data Requirements .....	38
12		Performance Requirements .....	39
	12a	Speed and Latency Requirements .....	39
	12b	Precision or Accuracy Requirements .....	40
	12c	Capacity Requirements .....	40
13		Dependability Requirements .....	40
	13a	Reliability Requirements.....	40
	13b	Availability Requirements.....	41
	13c	Robustness or Fault-Tolerance Requirements.....	41
	13d	Safety-Critical Requirements .....	41
14		Maintainability and Supportability Requirements.....	41
	14a	Maintenance Requirements .....	41
	14b	Supportability Requirements.....	42
	14c	Adaptability Requirements.....	42
	14d	Scalability or Extensibility Requirements.....	42
	14e	Longevity Requirements .....	42
15		Security Requirements.....	43
	15a	Access Requirements .....	43
	15b	Integrity Requirements.....	43
	15c	Privacy Requirements .....	43
	15d	Audit Requirements.....	44
	15e	Immunity Requirements.....	44
16		Usability and Humanity Requirements .....	44
	16a	Ease of Use Requirements.....	44
	16b	Personalization and Internationalization Requirements .....	45
	16c	Learning Requirements .....	45

16d	Understandability and Politeness Requirements .....	45
16e	Accessibility Requirements .....	46
16f	User Documentation Requirements .....	46
16g	Training Requirements .....	46
17	Look and Feel Requirements .....	47
17a	Appearance Requirements .....	47
17b	Style Requirements .....	47
18	Operational and Environmental Requirements .....	47
18a	Expected Physical Environment .....	47
18b	Requirements for Interfacing with Adjacent Systems .....	48
18c	Productization Requirements .....	48
18d	Release Requirements .....	48
19	Cultural and Political Requirements .....	49
19a	Cultural Requirements .....	49
19b	Political Requirements .....	49
20	Legal Requirements .....	49
20a	Compliance Requirements .....	49
20b	Standards Requirements .....	50
III	Design .....	50
21	System Design .....	50
21a	Design goals .....	50
22	Current Software Architecture .....	51
23	Proposed Software Architecture .....	51
23a	Overview .....	51
23b	Class Diagrams .....	52
23c	Dynamic Model .....	54
23d	Subsystem Decomposition .....	62
23e	Hardware / software mapping .....	64
23f	Data Dictionary .....	64
23g	Persistent Data management .....	65
23h	Access control and security .....	66
23i	Global software control .....	66
23j	Boundary conditions .....	66
24	Subsystem services .....	67
25	User Interface .....	67

26	Object Design .....	72
26a	Object Design trade-offs .....	72
26b	Interface Documentation guidelines.....	73
26c	Packages .....	74
26d	Class Interfaces .....	74
IV	Test Plans.....	76
27	Features to be tested / not to be tested .....	76
28	Pass/Fail Criteria .....	77
29	Approach .....	77
30	Suspension and resumption .....	78
31	Testing materials (hardware / software requirements) .....	78
32	Test cases.....	79
33	Testing schedule .....	81
V	Project Issues .....	81
34	Open Issues.....	81
35	Off-the-Shelf Solutions .....	81
35a	Ready-Made Products .....	81
35b	Reusable Components .....	82
35c	Products That Can Be Copied .....	82
36	New Problems .....	82
36a	Effects on the Current Environment.....	82
36b	Effects on the Installed Systems.....	82
36c	Potential User Problems .....	82
36d	Limitations in the Anticipated Implementation Environment That May Inhibit the New Product .....	83
36e	Follow-Up Problems .....	83
37	Tasks.....	83
37a	Project Planning .....	83
37b	Planning of the Development Phases .....	83
38	Migration to the New Product .....	84
38a	Requirements for Migration to the New Product .....	84
38b	Data That Has to Be Modified or Translated for the New System .....	84

39	Risks .....	84
40	Costs .....	85
41	Waiting Room .....	86
42	Ideas for Solutions .....	86
43	Project Retrospective .....	86
VI	Glossary .....	86
VII	References / Bibliography .....	87
VIII	Index .....	87

## List of Figures

Figure 1 Title Picture of Yamato Ship.....	1
Figure 2 Starting Age of Battleships Game .....	13
Figure 3 First player's turn in 1st player's perspective .....	14
Figure 4 First player turn in 2nd player's perspective .....	15
Figure 5 Use Case Diagram .....	26
Figure 6 Database Design .....	39
Figure 7 Three tier architecture diagram .....	52
Figure 8 UML class diagram .....	53
Figure 9 New Account Creation .....	54
Figure 10 Login Use Case.....	55
Figure 11 Theme Selection .....	56
Figure 12 Ships Selection .....	57
Figure 13 Select Difficulty Level .....	57
Figure 14 Opponent Selection .....	58
Figure 15 Ships Placement.....	59
Figure 16 Aim and Shoot.....	59
Figure 17 History Question.....	60
Figure 18 Pause / Resume Selection .....	60
Figure 19 Display Leader Board.....	61
Figure 20 Logout Selection.....	62
Figure 21 Game Organization Subsystem .....	63
Figure 22 Game Play Design Subsystem.....	63
Figure 23 Deployment Diagram .....	64
Figure 24 Login User Interface.....	68

Figure 25 Sign Up User Interface .....	68
Figure 26 Profile User Interface .....	69
Figure 27 Choose Options User Interface.....	69
Figure 28 Select Opponent User Interface.....	70
Figure 29 Position Ships User Interface .....	70
Figure 30 Game Board User Interface .....	71
Figure 31 Leaderboard User Interface .....	72
Figure 32 Class Diagram (i).....	75
Figure 33 Class Diagram (ii).....	75
Figure 34 Class Diagram (iii) .....	76



## List of Tables

Table 1- Author and Version Logs of ‘Age of Battleships’ .....	9
--	---

Date	Author	Activities	Notes
09/19/2015	Prabhjot Singh	Section 2 and 6	Initial draft
09/20/2015	Bhavya Mahajan	Section 7 and 8	Initial draft
9/20/2015	Abraheem Irheem	Section 3 and 4	Initial draft
9/22/2015	Mukund Subakanthan	Section 1 and 5	Initial draft
9/24/2015	Abraheem Irheem	Part I	Proof reading
10/24/2015	Prabhjot Singh	Part II Sec: 10, 15, 17	Initial draft
10/25/2015	Abraheem Irheem	Part II Sec: 11, 12, 13	Initial draft
10/25/2015	Bhavya Mahajan	Part II Sec:14, 18, 19, 20	Initial draft
10/26/2015	Mukund Subakanthan	Part II Sec:9, 16	Initial draft

**Table 1- Author and Version Logs of ‘Age of Battleships’**

# **I Project Description**

## **1 Project Overview**

Age of Battleships game is a strategy game where the player destroys opponent's warships by using his/her knowledge on History. The already existing game "**Battleship**" is modified and new functionalities have been added to make learning history a fun activity.

The game is accessed as a desktop application. The player opens the application, logs in and starts playing. The database will be hosted at a server and the user plays the game over an internet connection. After logging in, the player can choose to play against a computer or a human.

Before the start of the game, players pick a historical event from a list of available topics (like World War I, World War II). Then players get to choose their warships. After choosing the warships, players can choose a difficulty level from a list of three levels which are Preliminary, Intermediate and Advanced. In the next stage, players will position their ships in the grid. The game proceeds in series of rounds. In each round, each player takes a turn and targets a spot in the opponent's grid and shoots the missile. The number of missiles that are shot at every turn will be equal to the number of player's living ships. If it is a hit, the player will be asked a question related to the historical event that he/she selected at the beginning. If he/she answers the question correctly, the opponent's ship gets damaged. Otherwise, opponent's ship is safe. Apart from this, players are given bonuses, penalties, option to move their ships and all these additional features varies depending on the difficulty level chosen at the beginning.

This game can be purchased by educational institutions like Universities, schools and they can have their own database. Teachers can have the access to update the topics and the questions. In this way, students can learn history while playing and teachers can also monitor students' activities by having a separate leaderboard for a particular section. Apart from this, the game can also be played globally where any person can play with any other person and they can view their rankings in a global leaderboard.

## **2 The Purpose of the Project**

### **2a The User Business or Background of the Project Effort**

A user logs in to Age of Battleships game and selects a historical event that they would like to play. The visual effects of the game mimic the historical event they selected to start the gaming session. The user plays the game either with a computer or with another player, and they take turns to shoot at each other's ships. When a shot is hit at the opponent's ship that shot is not validated until the user answer a history question relating to the event they chose at the beginning of the game. The main motivation behind development of Age of Battleships game is to make learning history engaging and pleasurable at the same time. Probably the main reason why history has been considered a boring subject is because of the monotonous and

seemingly non-participative way of teaching it. The problem, lack of interest among new generation learning our history must be solved by combining elements that new generation likes such as learning with technology and learning while playing a game. A Teacher can assign their class a history event and teacher gets real time leaderboard update on their students to monitor their activities.

## **2b Goals of the Project**

We want to change learning experience of the History subject from a boring tiresome experience to a fun activity for young generation by combining gaming technology. We want to provide a tool to the teachers to assign learning activity for their class and be able to gauge and monitor students' activities at the same time.

## **2c Measurement**

Age of Battleships game provides a tool for teachers to measure their student's interest in the subject and gauge the class activity. Teachers can check statistical data from the server that how many and how often students log in to play this game. Further, teacher will have access to the leaderboard to identify those students who are not scoring as good comparing to their peers. Teachers can arrange special help for the struggling students to do better in their class or they can evaluate themselves if they are successful teaching their students. Users of the game will learn the subject of History without having to get bored and further they will have competition spirit to achieve better scores in the game to stay on the leaderboard and eventually in the class exams as well.

# **3 The Scope of the Work**

## **3a The Current Situation**

Currently, there are many desktop and web based apps for playing games similar to Battleships; however, they have minimal strategy involved in them. Our product will have many features including playing over the network, choosing different difficulty levels, the ability to move your ships, and much more. It will also have features which encourage learning the history of the world in order to formulate a successful strategy.

Nowadays, people think of history as a boring subject, and there is no good motivation in learning it; however, we wish to change that by helping the player enjoy learning history by making that part of their game strategy. Now players with a good history background will be the most successful ones.

## **3b The Context of the Work**

Not applicable.

### **3c Work Partitioning**

Not applicable.

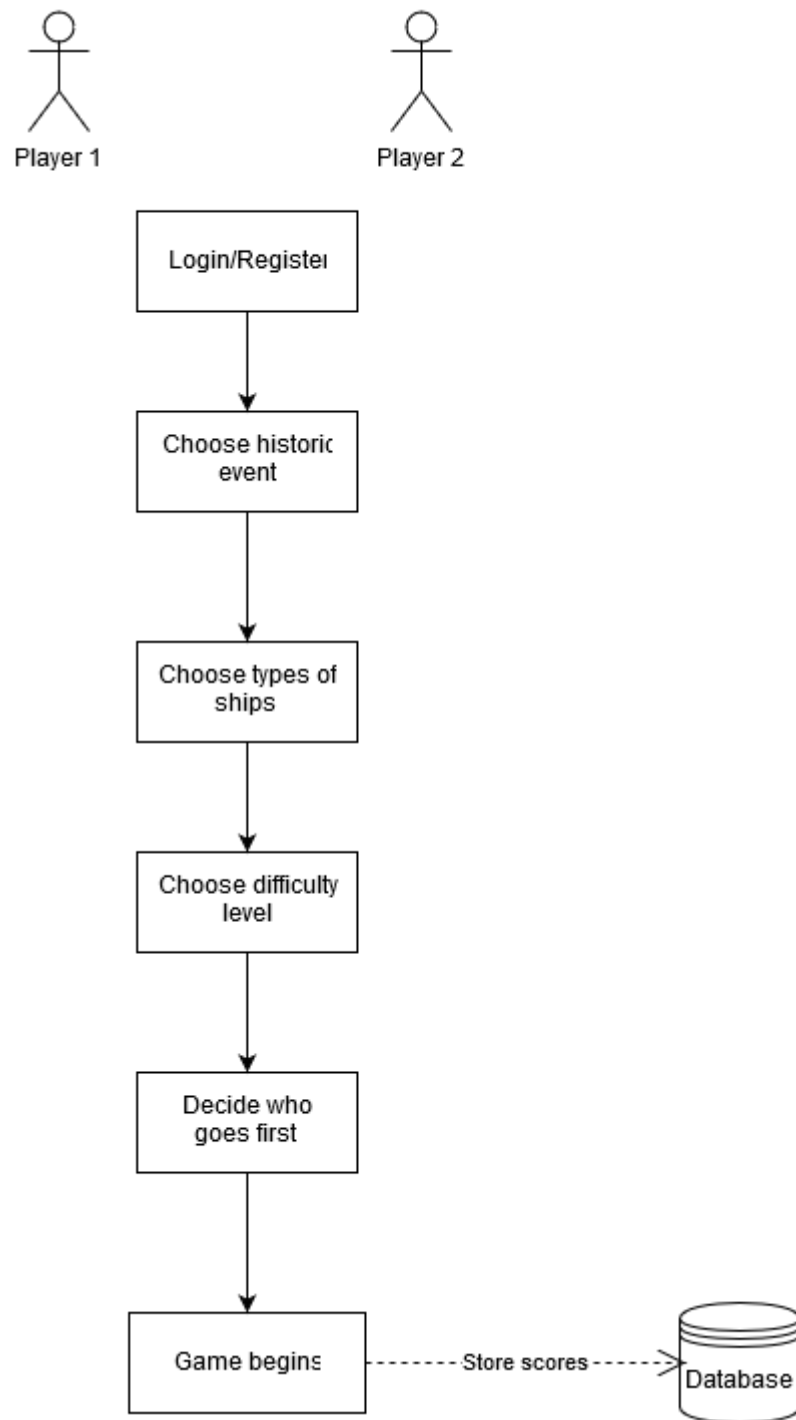
### **3d Competing Products**

Competing products include all the different variations of the game Battleships. However, what we can provide that is different from the competition is the history learning aspect, and it has not been done this way before...

## **4 The Scope of the Product**

The purpose of this section is to define what is included in the product to be created, and what is not. It is also important to identify what entities external to the product interact directly with the product, and in what manner. This then defines the boundary of the product and the interface it has with external entities (which may be humans, hardware, other software systems, etc.)

#### 4a Scenario Diagram(s)



**Figure 2 Starting Age of Battleships Game**

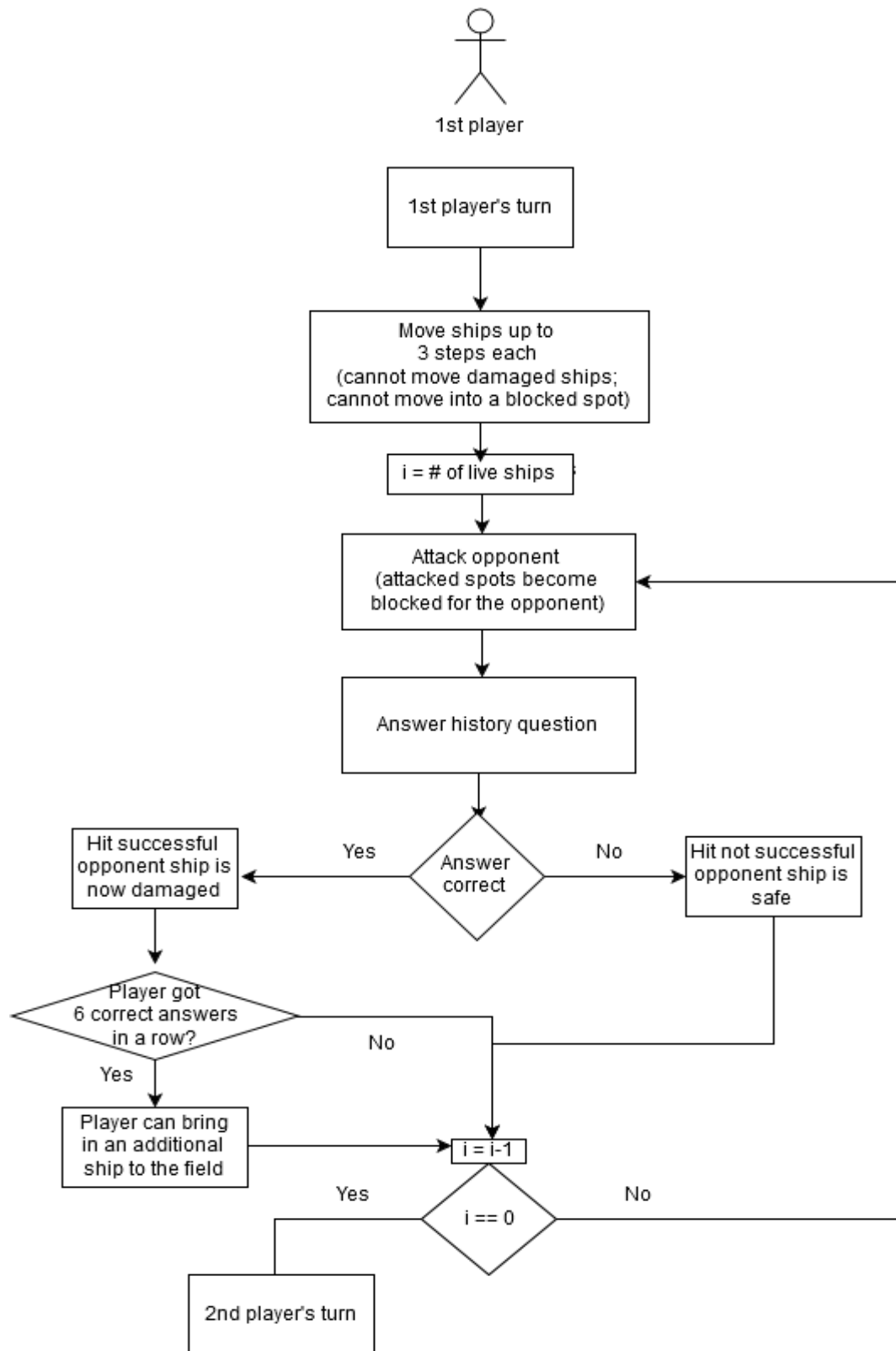
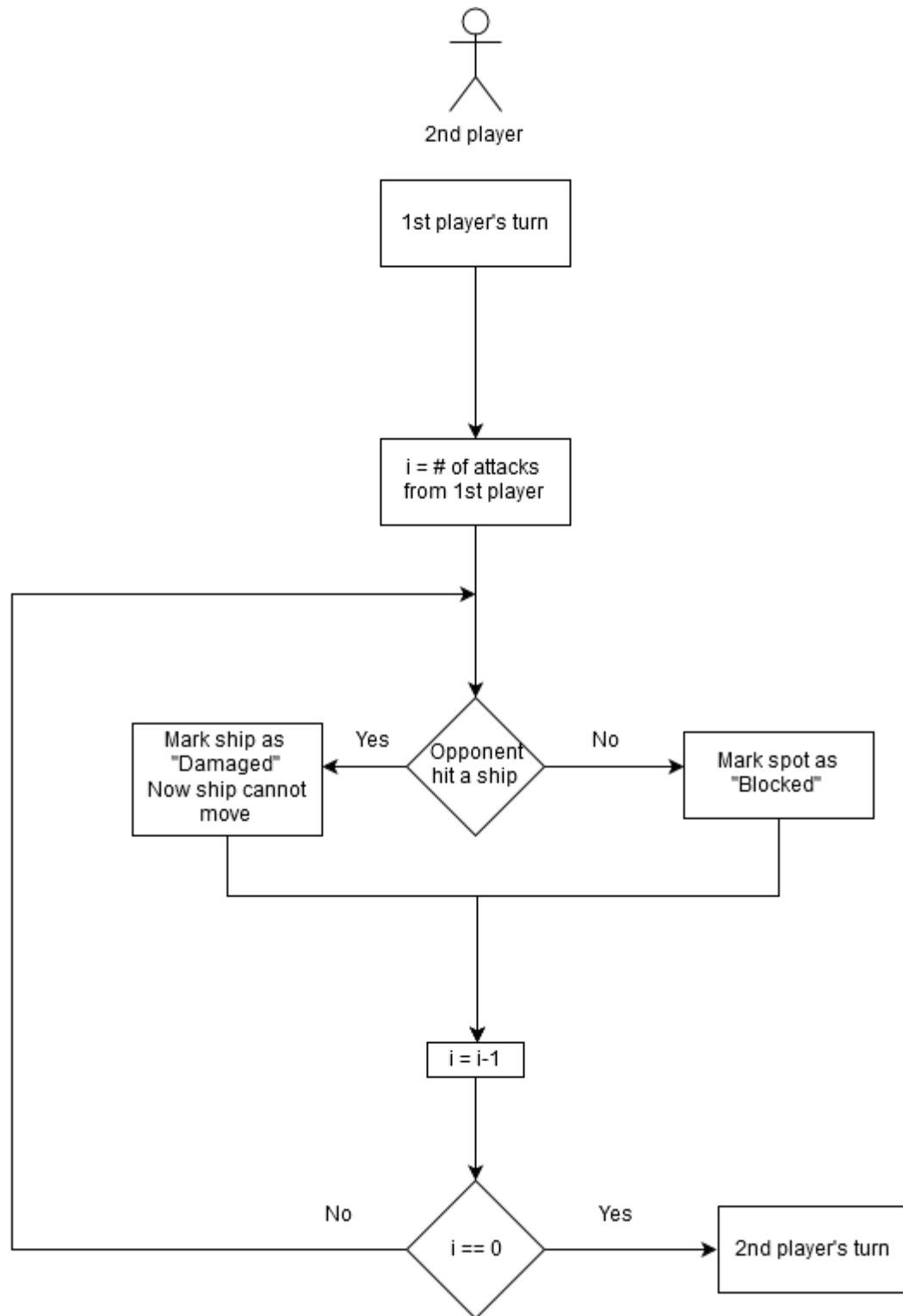


Figure 3 First player's turn in 1st player's perspective



**Figure 4** First player turn in 2nd player's perspective

#### 4b Product Scenario List

- (1) Players login/register
- (2) Choose historic event
- (3) Choose types of ships (will be relevant to historic event)
- (4) Choose difficulty level
- (5) Move ships
- (6) Attack opponent
- (7) Answer history questions to make attacks successful
- (8) Bonus for getting multiple correct answers in a row

#### 4c Individual Product Scenarios

- (1) New players will register; returning players will login.
- (2) **Choosing historic event:** The historic events will be related to battleship battles that have happened through history. Say the player Bob chooses the *Battle of Casablanca* as his desired historic event, then during the course of the game, if Bob succeeds in landing a hit, then he will be asked a question such as *Who was America fighting against in the battle of Casablanca?* Or; *where did the battle of Casablanca took place?*
- (3) **Choose types of ships:** Now when Bob is choosing his ships, he will have choices such as *Jean Bart* which was a French battleship or other ships that were present at that time.
- (4) **Choose difficulty level:** There are three difficulty levels: Preliminary, Intermediate, and Advanced. These levels reflect the way players are able to move their ships and the penalties in getting incorrect answers. (5) And (7) explain these in more detail.
- (5) **Move ships:** In the preliminary level, players cannot move their ships, and there are no penalties in getting multiple incorrect answers. In the intermediate and advanced levels, players will be able to move each of their ships a certain number of steps during their turn.

In the intermediate level, players will be able to move their attacked ships forward or backward from the direction they are facing, but they cannot change the rotation of their ships. In the advanced level, players will be able to do everything in the intermediate level; in addition, they will be able to change the rotation of their attacked ships if their opponent got multiple incorrect answers.



- (6) **Attack opponent:** Players will be able to attack their opponent once for every live ship that they have on the field.
- (7) **Answer history question:** When Bob attacks Alice, if he successfully aimed for one of Alice's ships, he must answer a question related to the historic event that he chose in the beginning of the game. If he answers correct, then the attack becomes successful and Alice's ship becomes damaged; however, if the answer was incorrect, Alice's ship will be safe. The type of questions Bob will be asked are like the ones mentioned in (2). In the intermediate and advanced levels, there will be a penalty for getting multiple incorrect answers in a row. The penalty will be that the player will not be able to move their ship.
- (8) **Bonus:** If Bob or Alice get multiple correct answers in a row (this number depend on the difficulty level), they will be able to bring in a new ship to their side of the field, and they can place it wherever they want.

## 5 Stakeholders

### 5a The Client

The client for this game will be the company that will purchase the game from the developers. The client will be in charge of maintaining and updating the game. The client will also have some responsibility of marketing and providing the game to the customers. The Client can run the game globally and apart from that, they would also approach different schools and colleges to sell this product.

### 5b The Customer

The main customers are various educational institutions like schools and colleges. They purchase this game from the company that is selling and can customize the historical events and the question sets according to their needs. In this way, schools and colleges can use this game as a tool and teachers can use this game to assign events for students in a class and can track students' interests and involvement.

The students who take the history class can use this game to learn history in a playful manner. Instead of reading boredom topics, they can ask the teacher to add the topics to this game and can start playing with their classmates.

This game will also be hosted at the global level where anyone having an account can login and play with any other person.

### 5c Hands-On Users of the Product

The hands-on users can be divided into three different categories. The first category consists of school and college students who are studying History. They can really benefit from this game because they can learn history while playing which would also help them academically. Next group is the teachers who are teaching history class. They can use this game to assign some events among the students and also to track students' performances. The last group includes any common person from the real

world who is interested in playing games. They play this game for fun, to pass some time, to test their knowledge on history and also to learn about history. These three different types of hands-on users are described in detail below.

**User Name:** Students who are studying history.

**User Role:** Can compete with their classmates. Some might have some trouble in remembering certain events or years. They can play this game to learn those things in a fun way.

**Subject matter experience:** No prior knowledge is required. It is completely possible to learn the game by playing few times.

**Technological experience:** The user must be capable of using a web connected computer.

**Other user characteristics:**

**Physical Abilities/Disabilities:** Person with weak eye sight is advised not to spend long periods of time on the computer. It will be easy to get addicted to the game. So it is advised that the users take breaks every two hours.

**Intellectual Abilities/Disabilities:** The game is designed to combine knowledge and fun part. Therefore, anyone is welcome to play this game. Any person can play despite any intellectual disabilities.

**Attitude toward job:** It is highly recommended that the students playing the game have positive attitude and desire to learn and enhance their knowledge.

**Education:** No advanced education is required but students who are strong in history will have higher rate of success.

**Linguistic Skills:** No special linguistic skills are required to play this game.

**Age Group:** This age group will most likely include students who are between the ages of 12-24. It can also include individuals who are going back to school or those attaining higher level degrees.

**Gender:** This game is for both males and females.

**User Name:** Teachers who teach history.

**User Role:** Can monitor students' activities and can test the students' knowledge.

**Subject matter experience:** They have prior knowledge about history. And they can use their knowledge to update the question sets.

**Technological experience:** The user must be capable of using a web connected computer.

#### **Other user characteristics:**

**Physical Abilities/Disabilities:** Person with weak eyesight is advised not to spend long periods of time on the computer. It will be easy to get addicted to the game. So it is advised that the users take breaks every two hours.

**Intellectual Abilities/Disabilities:** The game is designed to combine knowledge and fun part. This group of users should have some knowledge about history in order to make some updates to the question sets.

**Education:** No advanced education is required but students who are strong in history will have higher rate of success.

**Linguistic Skills:** No special linguistic skills are required to play this game.

**Age Group:** This age group will most likely include teachers and mostly they are between the ages of 25-55.

**Gender:** This game is for both males and females.

**User Name:** Common man

**User Role:** Can play and compete with anyone in the real world. They play for various reasons like fun, time pass and to gain some knowledge.

**Subject matter experience:** No prior knowledge is required. It is completely possible to learn the game by playing few times.

**Technological experience:** The user must be capable of using a web connected computer.

#### **Other user characteristics:**

**Physical Abilities/Disabilities:** Person with weak eye sight is advised not to spend long periods of time on the computer. It will be easy to get addicted to the game. So it is advised that the users take breaks every two hours.

**Intellectual Abilities/Disabilities:** The game is designed to combine knowledge and fun part. Therefore, anyone is welcome to play this game. Any person can play despite any intellectual disabilities.

**Education:** No advanced education is required but students who are strong in history will have higher rate of success.

**Linguistic Skills:** No special linguistic skills are required to play this game.

**Age Group:** This age group includes any person starting from kids to elders. So they fall in the age group of 8-60.

**Gender:** This game is for both males and females

#### **5d Priorities Assigned to Users**

- **Key users:** We believe that this game can be an important resource for students who are studying history either in their school or college. Therefore, these students are our key users. Students have the opportunity to use this game to enhance their knowledge on history. Other key users includes teachers who teach history. Teachers use this game to test students' performance and to monitor their interests. These people are important because they recommend this game to the school/college board and they can take steps to purchase and install this game.
- **Secondary users:** This includes all the individuals from the real world starting from kids to elder people. The advantage of this game is that the player can have fun and learn things at the same time. These users help in spreading the game across the globe. Therefore, this group is very important for the success of this game.

#### **5e User Participation**

User's inputs could be very helpful while developing the game especially in modules like developing graphics, deciding on the music, sounds etc., and also about the overall usability of the product. This is a game and fun part depends mainly on the gaming experience. If the user feels the game functionalities are confusing or if the user does not feel comfortable to play the game, then the main objective of developing this product fails. So user participation is very important in development of this game.

#### **5f Maintenance Users and Service Technicians**

Maintenance users will include a group technicians who will work together to remove and fix any problems that are discovered by the users. They will also provide technical support to customers. This group will also update the database by adding or updating new topics and question sets. The game becomes monotonous if the user gets used to the questions that are asked. For this reason, topics and question sets should be updated so that user faces new challenges and learns new information as the game progresses.

#### **5g Other Stakeholders**

- **Testers:** Testers will make sure to discuss and fix any bugs that may exist in the program. This will help keep the game updated and keep the users happy.
- **Marketing experts:** This group is extremely important in helping spread the game throughout the world with their advertisements. It will be marketing team's job to find ways to advertise the game through online advertisements or billboards or other ways.

## 6 Mandated Constraints

### 6a Solution Constraints

There might be number of stumbling blocks that developers will encounter; however, there are some constraints that have been put in place in the following section.

- i. Description: The user installs the client side user interface on their personal computer. The product's database shall be hosted on a server, and the client will interact and play the game over an internet connection.

Rationale: The school administration or the teacher using this product would like to host this application on their own server and have easy access to it while students/users don't have to be on campus to play game. Users can simply install the application on their computer and login to the application using internet protocols to connect to the servers.

Fit criterion: The users of the product shall be able to play the game using their personal computers over the internet.

- ii. Description: The product shall provide a graphical user interface to play the game.

Rationale: The user will not be motivated to play the game using a command line because they will have to learn various commands to interact with the system.

Fit criterion: User is presented with graphical user interface to play the game with ease of use without having to follow user manuals.

- iii. Description: The product shall not require a constant internet connection to connect to the database server to store achievement scores or get updates.

Rationale: Some users may not have connection to the internet at all times during the day due to commuting, etc. In such a situation, the game shall not hang up, crash, or simply refuse to work, but it shall store the user's achievements in local cache and ping the server at a later time whenever the connection to the server is established to store the achievements or get the latest updates.

Fit criterion: The product shall work by displaying a warning message; for instance, when there is no internet to connect to the database server..

### 6b Implementation Environment of the Current System

The product shall be cross platform regardless of the operating system. The users will be using MS Windows, MacOS or Linux operating system on their personal

computers and may not afford to get another computer in order to play the game. The product shall be approved as cross platform compliant by the QA testing group.

## **6c Partner or Collaborative Applications**

Age of Battleships game will be multi-tier design, and the database tier will be hosted on the central server; therefore, it is important to have the client user interface with the ability to establish a connection to the database server. Since Age of Battleships requires to be cross platform compliant, different network protocol from different operating system may be used. It is up to the developer what package is suitable to use as long as it fulfills the cross platform compatibility.

## **6d Off-the-Shelf Software**

Age of Battleships game requires from the administrator of the product to get the user's statistical information, top and low scorer user's information and ability to update or add new history question to existed themes. The apparent approach will be to store all of this information on database server. To fulfill this requirement, an off the shelf but reliable and cheaper database management system (DBMS) will be used. There are many open source free of charge DBMS that are available to accommodate our requirements. Depending on the server that will host the DBMS, it is recommend to use from MySQL, Microsoft SQL Server Express, Apache Derby and PostgreSQL. Developer may use different DBMS, if they wish to choose this option the only requirement DBMS must satisfy is that it should be able to backup database periodically and in case of emergency i.e. power failure, it should not lose the non-backed up data. Further if developer choose their choice of DBMS, it must be able to handle large traffic.

## **6e Anticipated Workplace Environment**

Age of Battleships is designed mainly for educational benefit for school or university students. The user will be using this application either from their home/dorm or school/college campus. It is required that the game shall provide quick access to an option to toggle the volume button, so that way it does not cause distraction among other students in the case of later. Since this application is also considered a distance learning tool, the application must let the users to play and interact with it in absence of internet or network connection to the database server. In the absence of database connection, the application shall resort on local caching mechanism. It is recommended that to use xml or Json file format caching for quick access.

## **6f Schedule Constraints**

Age of Battleships is targeted for schools and universities as a learning tool; therefore, it is important that the application is developed and tested before the start of the school calendar year. Ideally application should be deployed on the school servers two weeks ahead of the school opening to accommodate the training of school admin and teaching staff. If the application is not deployed before the school/college classes begin, it will force the product to get undesired delay to get redeployed and distributed among users until the next semester/quarter/year.

## **6g Budget Constraints**

Age of Battleships game will be deployed on school supplied server unless school wish to have development team purchase one. It is require that to make this project a reality to heavily use open source development tools. Depending on the choice of language the project module is built on, it is recommended that the game development shall not rely on paid version of IDEs.

## **7 Naming Conventions and Definitions**

### **7a Definitions of Key Terms**

All Terms, Including Acronyms and Abbreviations, Used in the Project are

MP- Multiplayer

SP- Single Player

WW1- World War 1

KO- Knockout.

### **7b UML and Other Notation Used in This Document**

Not Applicable.

### **7c Data Dictionary for Any Included Models**

#### **Miss**

When a user shoots a shot on the opponent's ship and it misses the opponent's ship it's called a miss. It is denoted as white dot in the game to the corresponding cell of the opponent's grid.

#### **Hit**

When a user shoots a shot on the opponent's ship and it hits the opponent's ship it's called a hit. But the hit is not validated unless and until the user answers a particular question related to the historical event the user has selected. It is denoted as yellow dot.

#### **Validated hit**

When a user shoots a shot on the opponent's ship and it hits the opponent's ship it's called a hit. After hitting the user will be prompted a question related to historical event the user has selected. If he answers it correctly then the hit is validated, but if he answers it wrongly then it's considered a miss. A validated hit in the game is denoted by a red dot.

### **Knockout (KO)**

If player Roger destroys all of the opponent's ships, it means that the player Roger has KO the opponent.

### **Clean Sweep**

A clean sweep is a situation in which a player destroys all the enemy ships with none of the player ships getting destroyed in the battle.

### **Battle Report**

At the end of the game each player will be presented with a battle report which depicts the total number of ships destroyed by the player along with the Accuracy rate

Accuracy Rate= (Total number of validated hits / Total number of shots fired)\*100

### **Career Report**

A player can view his career report, which includes total number of games won by the player, total number of questions answered, total number of ships destroyed and his ranking on the leaderboard.

## **8 Relevant Facts and Assumptions**

### **8a Facts**

#### **Selecting the Battlefield**

At the beginning, the players can select a battlefield from a number of battlefields available. For example World War 1(WW1), Vietnam war etc. Depending on the battlefield he selects he gets the questions related to that event.

#### **Selecting the difficulty level**

There are three difficulty levels and the user can select any one of them. The three levels are:

- Preliminary:

In this level if a player doesn't answer a question correctly, then the opponent cannot change the position of his ship

- Intermediate:

In this level if a player misses three answers, the opponent can move his ship 3 steps forward or backward provided those spots are unoccupied.



- Advance:

In Advance level if a player misses three answers, the opponent can move his ship in any direction by 3 steps, provided those spots are unoccupied.

### **Selecting the number of Ships**

The players playing the game, with mutual consent select the number of ships they want to choose for battle.

### **Number of shots**

The number of shots each player has in his turn is equal to the number of live ships that the player has.

## **8b Assumptions**

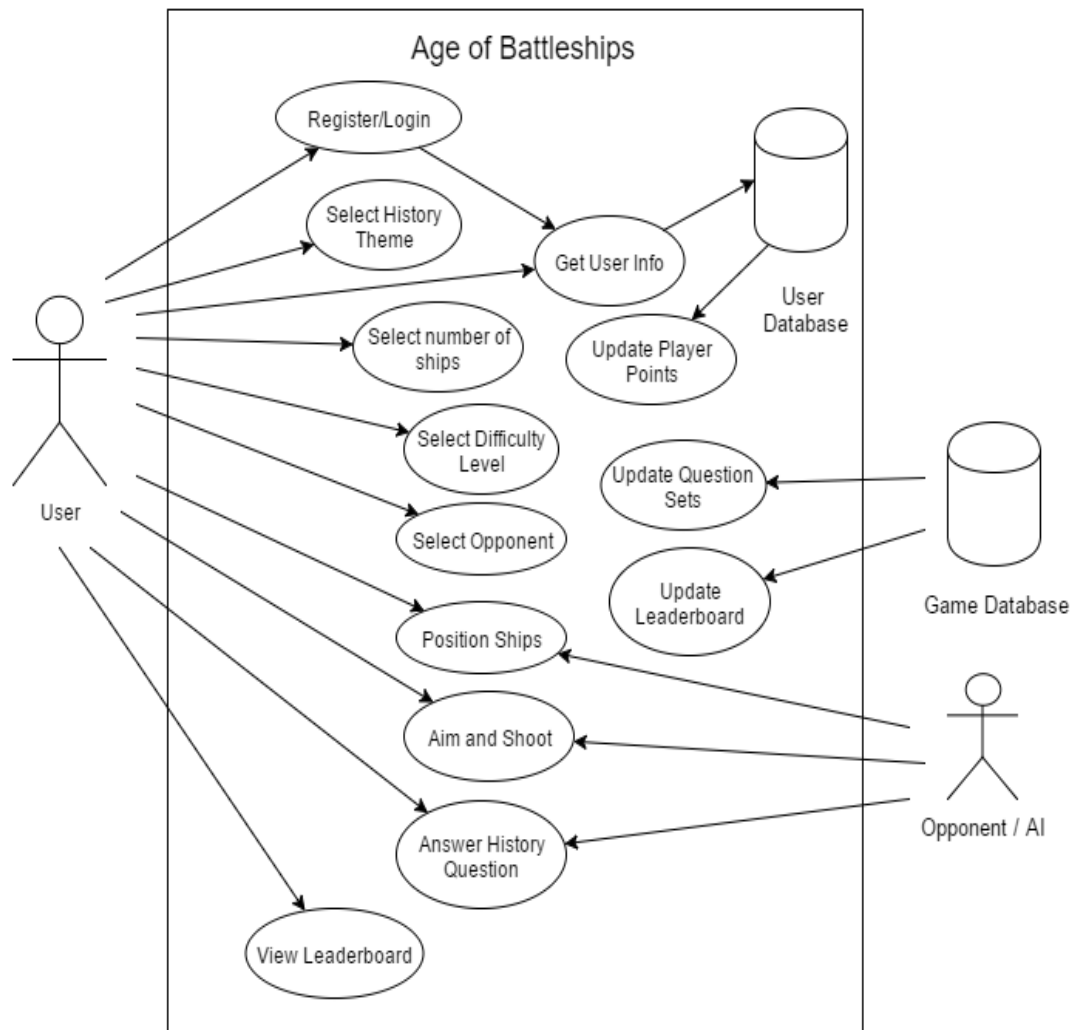
If a player plays the game against the computer, then we assume that the computer (AI component) plays the game with the difficulty level the player selected in the beginning. For example if the player initially selects preliminary level then the computer should play against the player considering that the player is just a beginner.

Another assumption that we are making is that the players should mutually agree on some points. For example in selecting the number of ships the players should mutually agree upon the number of ships they are going to use in the battle, and they should also agree upon the length of each ship used in the battle.

## II Requirements

### 9 Product Use Cases

#### 9a Use Case Diagrams



**Figure 5 Use Case Diagram**

#### 9b Product Use Case List

The following is the list of use cases, further developed in the next section.

1. Register User
2. Login Account

3. Choose History Theme
4. Choose Ships
5. Choose Difficulty Level
6. Choose Opponent
7. Position Ships
8. Aim and Shoot
9. Answer History Question
10. Pause and Resume Game
11. Display Leaderboard
12. Logout

#### 9c Individual Product Use Cases

Use Case Name	Register User
Participating Actors	Initiated by New User Communicates with the User database
Flow of Events	<ol style="list-style-type: none"> <li>1. The system displays welcome screen with a “Sign Up” button.</li> <li>2. User clicks “Sign Up” button.</li> <li>3. The system displays a pop up window asking the user to enter Username, First Name, Last Name, E-mail and password.</li> <li>4. User enter required information and clicks “Submit”</li> <li>5. The user is brought to the welcome screen from where the user can login using the credentials.</li> </ol>
Entry Condition	User access the game through desktop application
Exit Condition	User can successfully login using the credentials provided at the time of Sign Up.

Quality Requirements	User's information stored in the User database.
----------------------	---

Use Case Name	Login Account
Participating Actors	Initiated by User  Communicates with the User Database
Flow of Events	<ol style="list-style-type: none"> <li>1. The system displays the welcome screen with a text field for Username and Password and a "Login" button</li> <li>2. User enters required information and clicks "Login" button.</li> <li>3. The system verifies username and password from the database and if found incorrect displays "Incorrect Username and Password"</li> <li>4. If user enters correct information, the user is logged in and the system displays User's portfolio and the user can start playing the game.</li> </ol>
Entry Condition	User access the game through desktop application
Exit Condition	User can view the portfolio and can start playing the game.
Quality Requirements	<ol style="list-style-type: none"> <li>1. Authentication should be done within 200 Milliseconds.</li> <li>2. Portfolio should be displayed first and then game may start.</li> </ol>

Use Case Name	Choose History Theme
Participating Actors	Initiated by the system. Communicates with the Game database
Flow of Events	1. User clicks “Play” button. 2. The system displays a list of History themes. 3. User selects any one theme and clicks “Next”.
Entry Condition	User is logged in.
Exit Condition	User is brought to the page where the user can select ships.
Quality Requirements	User’s selection is recorded for that game.

Use Case Name	Choose Ships
Participating Actors	Initiated by the system Communicates with the game database
Flow of Events	1. User clicks “Next” after selecting History theme. 2. The system asks the user to select the number of ships. By default, user has 3 ships. 3. User can have maximum of 5 ships. 4. After selecting, user clicks “Next”
Entry Condition	User is logged in and has selected the history theme.
Exit Condition	User is brought to the page where the user can select difficulty level.

Quality Requirements	User's selection is recorded for that game.
----------------------	---

Use Case Name	Choose Difficulty Level
Participating Actors	Initiated by the system  Communicates with the Game database.
Flow of Events	1. User clicks "Next" after selecting number of ships.  2. The system asks the user to select the difficulty level. The user can select any level among Preliminary, Intermediate and Advanced.  3. User selects a level and clicks "Next"
Entry Condition	User is logged in, selected history theme and number of ships.
Exit Condition	User is brought to the page where the user can select difficulty level
Quality Requirements	User's selection is recorded for that game.

Use Case Name	Choose Opponent
Participating Actors	Initiated by the system Communicates with the Game database.
Flow of Events	1. User clicks “Next” after selecting difficulty level.  2. The system asks the user to select the opponent. The user can either play with Computer (AI) or with another user.  3. If user wants to play with another user, user can select any player from a pool of online players.  3. User selects opponent and clicks “Next”
Entry Condition	User is logged in, selected history theme, number of ships and difficulty level
Exit Condition	User is brought to the game grid where the user can position the ships.
Quality Requirements	User’s selection is recorded for that game.

Use Case Name	Position Ships
Participating Actors	Initiated by the system Communicates with the Game database.
Flow of Events	1. User clicks “Next” after selecting opponent.  2. The game grid is displayed with the ships.  3. User positions the ships and selects “Start”
Entry Condition	User is logged in, selected history theme, number of ships, difficulty level and opponent.
Exit Condition	The game is started and the user can start shooting.

Quality Requirements	User's ships' positions are recorded for that game.
----------------------	---

Use Case Name	Aim and Shoot
Participating Actors	Initiated by the user
Flow of Events	<ol style="list-style-type: none"> <li>1. User clicks "Start" after positioning the ships</li> <li>2. Opponent's grid is displayed without the ships and now the user can aim a point in the grid and shoot.</li> </ol>
Entry Condition	User has started the game
Exit Condition	Either user hits the opponent's ship or misses hitting them.
Quality Requirements	User's hit and miss are recorded.

Use Case Name	Answer History Question
Participating Actors	Initiated by the system
Flow of Events	<ol style="list-style-type: none"> <li>1. User shoots and hits opponent's ship.</li> <li>2. The system asks the user to answer a question which is related to the theme selected by the user at the beginning.</li> <li>3. If the user answers correctly, the hit is validated and opponent's ship is damaged. If not, opponent's ship is not damaged.</li> </ol>
Entry Condition	User aims a spot in the grid and shoots and hits opponent's ship.
Exit Condition	Based on user's answer, the ships gets damaged.



Quality Requirements	The system waits for 10 seconds for the user to answer.
----------------------	---

Use Case Name	Pause and Resume
Participating Actors	Initiated by the user Communicates with the Game database.
Flow of Events	1. User clicks “Pause” from the portfolio screen 2. The system save the current instance of the game along with scores. 3. The system gives option to exit and resume on later time or stay on pause screen and wait for the user to resume game in a while.
Entry Condition	User is logged in.
Exit Condition	User have saved the current instance of their game to resume from the situation at later time.
Quality Requirements	The system takes 350 Milliseconds to save the current instance and display the confirmation.

Use Case Name	Display Leaderboard
Participating Actors	Initiated by the user Communicates with the Game database.
Flow of Events	1. User clicks “Leaderboard” from the portfolio screen 2. The system displays all-time leaderboard. 3. The system gives option to view the leaderboard on yearly, monthly and weekly basis.
Entry Condition	User is logged in.
Exit Condition	User views leaderboard.

Quality Requirements	The system takes 200 Milliseconds to load the leaderboard.
Use Case Name	Logout
Participating Actors	Initiated by the user
Flow of Events	1. User clicks “Logout” from the portfolio page.  2. The system saves any unsaved data and logs out the user.
Entry Condition	User is logged in.
Exit Condition	User is logged out and brought to welcome screen.
Quality Requirements	The system logs out the user in 200 Milliseconds.

## 10 Functional Requirements

Requirement #: 1

Description: The system shall provide meanings for the user to register and create new account with username and password.

Rationale: To be able to know the identity of the user, monitor and record their performance in the game.

Fit Criterion: A new user is provided with an option at the login menu to create a new account, where user enter their name, email, password recovery question, school and class/course to create a new user account. Once user successfully creates an account, they can login to the game with their username and password.

---

Requirement #: 2

Description: The system shall allow the user to select a player from a pool of online players when system is connected to the internet or intranet network. In such a case where connect to either internet or intranet cannot be established system shall allow the user to play against Artificial Intelligence.

Rationale: The game will be for competitive and user will be addictive to the game, resulting good knowledge of history.

Fit Criterion: A user who is connected to internet or intranet network can view a list of online available users and they can select a user who they want to play against. A user who is not connected to internet or intranet network is notified that they are not connected and prompted to start a game session against Computer Artificial Intelligence.

---

Requirement #: 3

Description: The system shall provide an option to recover password for existing users in case if they forget their password.

Rationale: Recover password and login to the game to previously saved stage.

Fit Criterion: An existed user can recover their password by answering the security question they selected and answered at the time of their registration. Once they recover their password they can successfully log back into the game where they left it last time.

---

Requirement #: 4

Description: The system shall allow the user to select different game themes of their choice.

Rationale: To be able to choose the game the user want to play instead forcing them to play a particular game scenario.

Fit Criterion: The user is presented with various choices of game themes to choose from, once the user select the game, the system render and let the user play the theme they choose.

---

Requirement #: 5

Description: The system must allow the user to play the game, in case there is no network connection to the main database server. In such an event system shall store the scores and related information in the local database, which is pushed to the main system whenever the connection to the main server is reestablished.

Rationale: To be able to interact with the system and play game while away from internet, i.e. traveling.

Fit Criterion: User is able to play the game when they leave the network and they are notified that the system is saving their score on local database, this saved information

is pushed to the main server database later time when connection has been reestablished.

---

Requirement #: 6

Description: The system shall allow the user to personalize user level preference for language, font and colors. The system shall save this information and display this preference unless user decide to change it to default.

Rationale: To be able to customize the look and feel.

Fit Criterion: A user customize system's supported language and color font schema to access the game. Once the preference is saved the game always start in the customized preference.

---

Requirement #: 7

Description: The system shall provide access to the top ten scoring player's leader board. The leaderboard display the top ten performing players' username and their current scores. The leader board can viewed with filters to see all-time leader by month or week as well.

Rationale: To be able to view, how other players are performing and compare their personal performance with them.

Fit Criterion: User can select an option to view the top ten leader and apply the filters to view the leaderboard for a selected month, week or all time since the game is in its existence.

---

Requirement #: 8

Description: The system shall provide base level and admin level access. An admin can monitor every user's performance with their name instead of only username. A base level privilege can only view the leaderboard only based on username but full names of the users.

Rationale: To be able to hide the user names for anonymity and to be able for the administrator or teacher can view each individual's' performance.

Fit Criterion: Admin level user have ability to select an option to search for all users and monitor their daily and total performance. The base level user is denied such access.

---

Requirement #: 9

Description: The system shall provide ability for admin level user to access the main database server and modify or/and upload new set of questions for supported game themes. Base level user is prohibited to modify or/and update these set of questions.

Rationale: To be able to upload new question sets, updated typos to the question sets and update the answers.

Fit Criterion: Admin level user can successfully update the question and upload new questions to the main database server. Such attempt by base level user is denied.

---

Requirement #: 10

Description: The system shall provide ability for all level of user to access their personal portfolio. The system shall provide details to the user regarding their scores, highest and lowest scores session. The user can view their detailed summary of every session, the questions with answer they provided and the right answer to that question.

Rationale: To be able to give the user ability to monitor their activities in the system.

Fit Criterion: The user has an option to select and view their personal detailed information along with ability to view the result of every game played on the system.

---

Requirement #: 11

Description: The system shall be able to run on major operating systems in the market. The system shall support Windows, Mac and Linux at the minimum. The user is provided with auto installer that checks and install appropriate pre-requisite on the user system. Upon successful installation, the user can double click on shortcut to start running the game on their choice of computer.

Rationale: To be able to run the game independent of operating system and ease of use for the user. The user is not forced to have particular choice of operating system in order to run the game.

Fit Criterion: The user is able to install and run the game on Windows, Mac or Linux Operating system.

---

Requirement #: 12

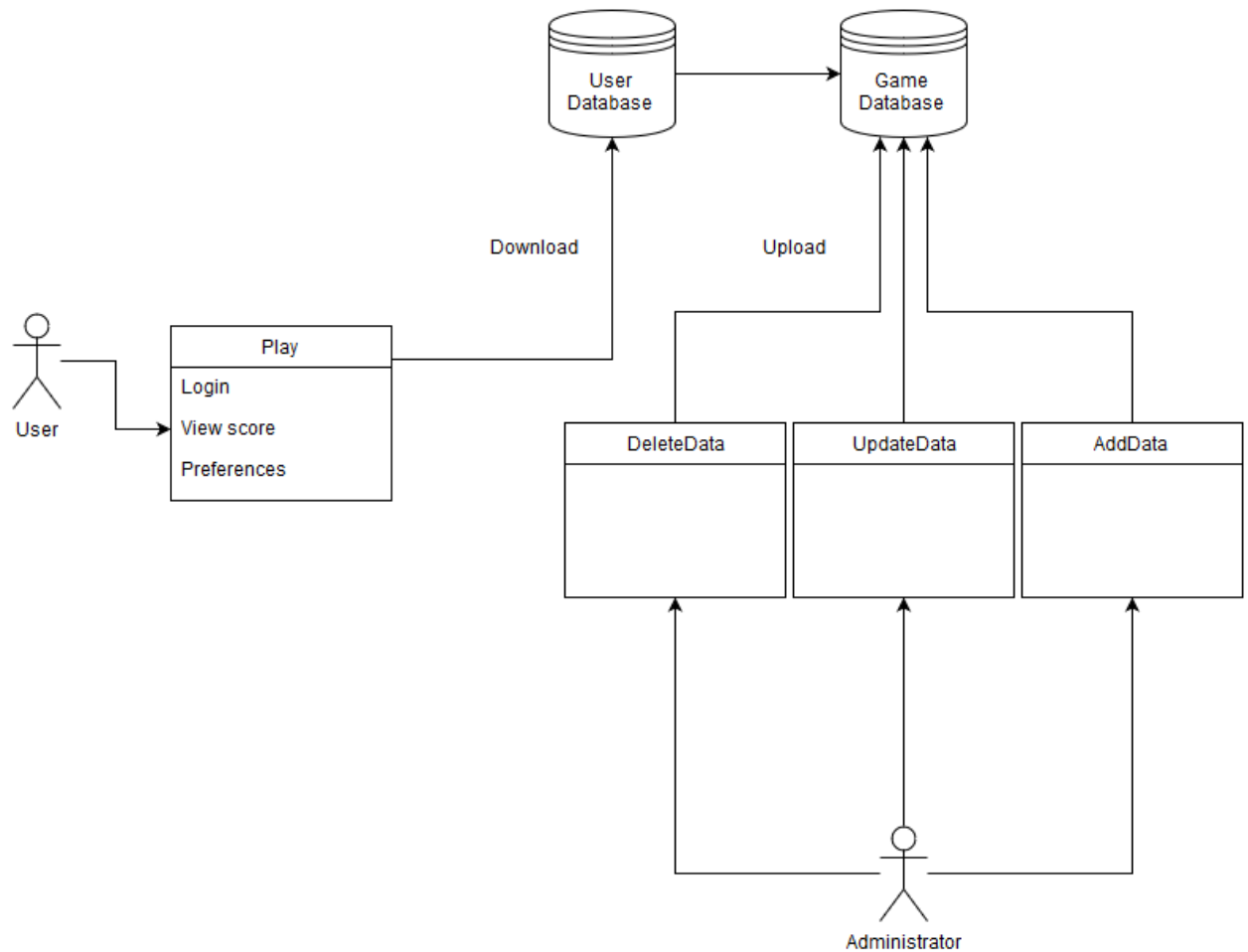
Description: The system shall provide an ability for the user to save their current game session to pause or exit to resume the game at later time. The user may want to take a break for their current game session and they should be able to resume the game from this stage onward.

Rationale: To be able to pause or leave the game and start from the last saved stage. The user will not be forced to finish the whole game or otherwise start the game from the beginning.

Fit Criterion: The user is able to save the current stage of the game to pause or exit to start playing from the saved instance of the game.

## **11 Data Requirements**

The diagram shows that there will be two databases, a user database and a game database. The user database contains all user information which include user login, user score, and user preferences. The game database hosts online games and provides players with their scores. The administrator interacts with the database with multiple classes including but not limited to: adding new data, deleting data, and updating the database.



**Figure 6 Database Design**

## 12 Performance Requirements

### 12a Speed and Latency Requirements

- The game must be loaded within five seconds after the running the application.
- New screens and pop-up windows must be loaded within 1.5 seconds, and no response time shall take longer than five seconds.
- After the system receives login credentials from the user, returning users must be able to login within one second.
- The user's score and preferences must be loaded within 500 milliseconds.
- After a user achieves a new score, the system must update the user's score in the database within 2 seconds.

- System must load the user's chosen ships within 1 second.
- System must move a ship within 500 milliseconds when the user chooses to move a ship.
- System must generate a history question when required within 200 milliseconds.

#### **12b Precision or Accuracy Requirements**

- The leaderboard must be updated daily to keep track of top players.
- The user must be able to return to their saved game.
- The AI must act exactly as chosen in the beginning of the game, so an easy level AI must be easier to play against than an intermediate level AI, and an intermediate level AI must be easier than an expert level AI.

#### **12c Capacity Requirements**

- The game server must be able to handle up to 50,000 active users logged in simultaneously.
- The game server must handle up to 7,000 games running simultaneously
- The game server must be able to store up to 500,000 user profiles.
- The game server must be able to store up to 1,000,000 distinct history questions.

### **13 Dependability Requirements**

#### **13a Reliability Requirements**

- No user data such as login credentials or virtual account shall be lost in the event of a system failure.
- The game must not experience any glitches more than once a day.
- The game must not experience any failures more than once a month.
- Maintenance must take place once a week.
- The system must notify users of the time and duration of the maintenance four days in advance.



### **13b Availability Requirements**

- The game must be available 24 hours a day, 365 days a year except when doing maintenance.
- The system must allow users to access their portfolio at all times even during maintenance periods.
- Updates for the game must be available once every 2 months.
- New versions of the game must be available once a year.
- The product must be functional and available 99 percent of the time.

### **13c Robustness or Fault-Tolerance Requirements**

- The system must notify all users if the database connection goes down which may restrict users from being able to log back in.
- In the case of lost internet connection, the system must allow users to save their current progress and return to the game wherever they left off as soon as they get back internet connection.

### **13d Safety-Critical Requirements**

- The game must not cause any physical damage to the players.
- The game must use a well-balanced color scheme so as not to disturb players or cause them any eye injuries.
- The game must use smooth transitions whenever necessary so as not to surprise the user with a sudden and drastic change in color.
- The system must be designed with efficiency so as not to cause the user's computer processor to overheat.

## **14 Maintainability and Supportability Requirements**

### **14a Maintenance Requirements**

- The maintenance team shall review and consider all of the suggestions for improvements that the users suggested in suggestion box.
- If the game is down for any reason, the maintenance team shall have it up and running within 2 hours of the exact time they are notified of the problem.
- There shall be a member of the maintenance team on duty at all times, even after the first release.

- The product shall be maintained by the maintenance team to ensure that all the bugs discovered after the releases are fixed promptly.
- Any patches for bugs shall be released within 10 days of the day that the bug is found.

#### **14b Supportability Requirements**

- The system shall allow quick and easy fetching of history questions from the central database
- The product shall provide assistance through email for users at all times. The maintenance team will be in charge of addressing the emails initially.
- The users shall receive the reply to their email within 10 hours.
- After the third release, users shall be able to receive support through telephone. This will be the responsibility of the customer support team, which will be formed after the third release of the game. The third release signifies that the game is popular among the users.
- If the user forgets their identification, the product shall email the user their identification information through email, provided that they correctly answered a security question.

#### **14c Adaptability Requirements**

- The product shall be compatible with the latest versions of Windows, Mac OS X, as well as latest Linux OS.
- All features of the previous versions of the product shall be compatible with the latest version.

#### **14d Scalability or Extensibility Requirements**

- The product shall be able to support nearly 5,000 users during the first 6 months of the first release.
- After one year, the game shall be able to support 15,000 users.

#### **14e Longevity Requirements**

- The product shall be able to run as long as it is hosted on a website.
- The features of the game shall continue to improve over time, removing any bugs when they are discovered and introducing newer, better features to enhance gameplay.

## **15 Security Requirements**

### **15a Access Requirements**

- Only the maintenance team has access to the source code.
- User's personal information such as username, password and email addresses shall not be accessed by anyone other than the user.
- The user database and game database shall only be accessed by the admin level users and the leader of maintenance team.
- User's portfolio can be viewed by the base level users and admin level users.
- Leaderboard can be viewed by base level and admin level users.

### **15b Integrity Requirements**

- The data in the user database shall not be manipulated or made available to public at any time.
- A concerted effort shall be made to protect user data and game data and no data shall be lost at any point.
- The maintenance team shall also conduct weekly checks to ensure that none of the data has been manipulated or modified.
- The product shall protect itself from outside users or hackers, attempting to manipulate the data.
- All of the user data shall be backed up in a separate database to ensure safety. The backups shall take place every week to ensure that the backup database is up to date.

### **15c Privacy Requirements**

- The privacy policy shall be made available to the user at the outset of the game, on the main menu. This can simply be done under the "Policies" section.
- The product shall notify the users through email and by posting a large message on the sign in page if any changes are made to the privacy policy.
- Besides the information/statics provided on the leaderboard, no other information about other users shall be made available to other users.
- The distributors of the game shall also comply by the privacy laws in their respective state, company and country to ensure that the user data is safely protected and cannot be violated.

## **15d Audit Requirements**

- The information about all the users are retained in order to track the users in case if any mishap happens.
- The information is retained only for security purposes so that the user may not deny having used the product.
- The information includes the name, e-mail id of the user and also the time intervals when the user has actively used the product.
- Any update/edit made to the system is stored in a separate database along with the timestamp.

## **15e Immunity Requirements**

- The application shall scan all entered data against the published definitions of known computer viruses, worms, and Trojan horses.
- The application shall delete the infected file if it cannot disinfect the infected data or software.
- The application shall daily update its list of published definitions of known harmful programs
- The application shall notify a member of the maintenance team if it detects a harmful program during a scan.

## **16 Usability and Humanity Requirements**

### **16a Ease of Use Requirements**

- The game shall not take more than five seconds to start, the game shell display a busy spinner or a signal indicating the user for long running background process when the user is supposed to wait for it to finish.
- The game shall provide an intuitive graphical user interface that the new user can navigate without having to read intense documentation. The new user should be provided with hints how to navigate the system. The experienced user shall have an ability to refer to more information menu regarding how to navigate and play the game.
- The game shall provide hints if the user is inactive for more than 2 minutes at any given time. The user might be having difficulty how to navigate to the next step.
- The user shall have an ability to recover their password by answering their security question and getting their password in email listed on their account.

- The game shall provide correct and up to date information when the user is connected to the network from the main database server. In an event when system is not connected to the main database server, they should be provided with feedback of the last time when the information currently displayed was fetched from the main database server.
- The game shall be appropriate for user of age 12 and up.

#### **16b Personalization and Internationalization Requirements**

- The game shall be accessible in English language at minimum. The game may provide additional languages to choose from.
- The game shall cover world historical question sets. The user should have ability to choose a region or country of their preference.
- The game shall provide ability for the user to change font, color and resize the window.
- The game shall provide an ability for the user to choose their desired username and password when username is available.

#### **16c Learning Requirements**

- The game shall be easy for a 3rd grade and up student to learn.
- An administrator maintaining the gaming database shall be able to be productive with one to two days training time.
- The game shall be able to be used by members of the group selected by school teachers and receive no training before using it.
- The game shall provide a tutorial for novice or new users to understand how the game is played.

#### **16d Understandability and Politeness Requirements**

- The game shall not lock the user computer and provide ability to be uninstalled from the user computer with ease of use.
- The game should always prompt the user with “please” prefix for their inputs.
- The game shall not include any violent graphical images.
- The game shall not use any word that can be offensive to any member of the community.

- The game shall not use any technical developers' jargon in the information or prompts. The language used for either of these should be layman or everyday spoken common word, so that the users are not confused.

#### **16e Accessibility Requirements**

- The game shall be accessible by all individuals as long as they can access a computer.
- The game shall conform to the Americans with Disabilities Act to be qualified as learning tool used by schools or universities.
- The game shall provide a conversion tool for different colors to set of user preferred colors, in case of individuals with color blindness.

#### **16f User Documentation Requirements**

- The game shall provide an electronic technical specifications to accompany the product with ability to print.
- The game shall contain user manuals for reference in electronic format with ability to print.
- The game shall provide a service manuals for administrators in printed and electronic format with ability to print.
- The game shall provide emergency procedure manuals in case of data outage for the administrators.
- The installation manuals should accompany in electronic format with game installer for ease of use and step by step installation guide for the user.

#### **16g Training Requirements**

- The administrators of the gaming database will require a one to two day training by technical support staff of the game to make sure administrator understand how to perform various admin related tasks for example, backing up database, recovering lost database or updating question sets of the game.
- The game shall provide new user tutorial with step by step video or gif format pictures to guide the new users. No formal training should be required for the users to interact or play the game.

## **17 Look and Feel Requirements**

### **17a Appearance Requirements**

- The product shall try to portray the real world feel of the battleground, with canons firing at enemy ships and shots hitting or missing the enemy ships.
- The product must allow the user to select from the various battlegrounds, and each battleground should have different background and theme depending on the battle.
- The product shall be attractive to users of all ages.
- The product shall not display any images deemed to be inappropriate.
- The product shall not display any content deemed to be inappropriate. This includes, but is not limited to, profanity and any other inappropriate words or phrases.

### **17b Style Requirements**

- The product shall make a user feel as if they are participating in a real time battle.
- The product shall make answering the history questions look as if they are part of the task to be done by soldiers in the battle.
- The game should look simple and avoid clustering a lot of materials together.

## **18 Operational and Environmental Requirements**

### **18a Expected Physical Environment**

- The game shall be used by a student on their computer, in library, study room, public coffee shop, public transit or a dorm room.
- The game shall be used in noisy conditions of a public transportation, coffee shop or a quiet study room or a library.
- The product shall be installed on a school computer on desktop or student's laptop that fit in a backpack or purse that can fit their laptop.
- The game shall be usable in dim light or brightly lit room or sunny day outside.
- The game shall be accessible at any time of the day with exception of pre-scheduled down times.

#### **18b Requirements for Interfacing with Adjacent Systems**

- The game shall work on the last two releases of supported operating system at the time of release.
- The game shall have the capability to reestablish a lost connection when the system connects back to the network.
- The game installed on the user's computer must be able to ping the database server.
- The game shall be able to access the database server either through internet or intranet.
- The local database installed on the user computer should be configured through the installed game on the user's computer.
- The administrator should be able to configure the database server through the game interface rather than having to interact directly with the server computer.

#### **18c Productization Requirements**

- The game shall be distributed as a setup file but not limited to, through database server, school website, CD or a flash drive.
- The game shall be able to be installed by double clicking on the setup file by following step by step prompts without separately printed instructions.
- The game shall contain all pre built binaries of all the prerequisite required for the application to function in the setup file without relying on internet to fetch those prerequisites.

#### **18d Release Requirements**

- There shall be a maintenance release every year that will be offer all new features and accumulated bug fixing patches from the previous releases.
- All previous features shall not be affected by the new release.
- The maintenance team require to provide a hotfix patch in timely manner for any identified bug.



- A new release require thorough testing by QA before it can be released to the user.
- Every new release will contain release notes that will explain the new features and bug fixed in the release.

## **19 Cultural and Political Requirements**

### **19a Cultural Requirements**

- The game shall not purposefully intend to offend individuals of any particular culture.
- The game shall be released in multiple languages, so as to support multiple regions and cultures.
- The game shall not favor any particular country during the battle.

### **19b Political Requirements**

- The game shall adhere by all laws laid out by the constitution of the United States
- The game shall not depict violence, so that individuals of all the age group can play the game.

## **20 Legal Requirements**

### **20a Compliance Requirements**

- The game shall not distribute user information to third-party source without the consent of the user, thereby complying with the Data Protection Act.
- The game shall avoid all copyright infringements.
- The game is available free online and shall not be redistributed by independent individuals or companies seeking personal gains.
- The product shall comply with gaming industry's rules and regulations and as the game shall also be used by school students it should also follow school board's rules and regulations.

## **20b Standards Requirements**

- The product shall meet the standards set by the Federal Government pertaining to the gaming industry.
- The product shall be developed following the agile development process.

## **III Design**

### **21 System Design**

During the system design, the system is to be transferred from the analysis model into a system design model. We have defined the design goals of the game and decomposed the system into smaller subsystems.

#### **21a Design goals**

The design goals represent the desired qualities of the Age of Battleships game, and it represents a consistent set of criteria that must be considered when making design decisions. Our main purpose is to deploy robust, maintainable, well-designed, and reusable software with object-oriented analysis and design. We are determined to define and visualize each and every perspective of the system explicitly in order to completely materialize our object-oriented approach. Next, we also pay attention to how to diminish the influence and impact of alterations or security exploits.

The design goals identified in details are as follows:

- **Adaptability:** Java is a popular programming language in terms of providing cross-platform portability. Therefore, it would be an optimal choice for the development of our program. Java enables our system to work in all Java Runtime Environment (JRE) installed platforms; therefore, the user will not have to worry about the operating system requirements. In order to fulfil this adaptability feature, we suggest programming the game in Java.
- **Efficiency:** The system is going to be responsive and be able to run with high performance. The game will run in at least 40 frames per second (fps) in order to provide smoothness in the movement of the game objects such as the battleships. This is a crucial design goal because a game that lags in performance will throw off users.
- **Reliability:** The System will be released bug-free consistent in the boundary conditions. Any bugs that might be discovered after the game is released shall be maintained and fixed by the maintenance team. The system must not crash with incorrect inputs.

- Usability: Easiness in the usage is an important design goal in games as well as any successful software system in terms of user's comfort. It makes the game more friendly and attractive. Therefore, the system will be designed such that user can easily interact with the system without any prior knowledge. However, user-friendliness of the system does not imply making the gameplay easier, which might make the player bored sooner than expected.
- Extensibility: Object oriented architecture of the game enables system customizations without causing any bugs during modifications. The game should be able to run with at least 7,000 active games concurrently at a point in time. The game is expected to capture more users as the time progresses. Therefore, this design architecture minimizes the possibility to cause malfunctioning in other classes.

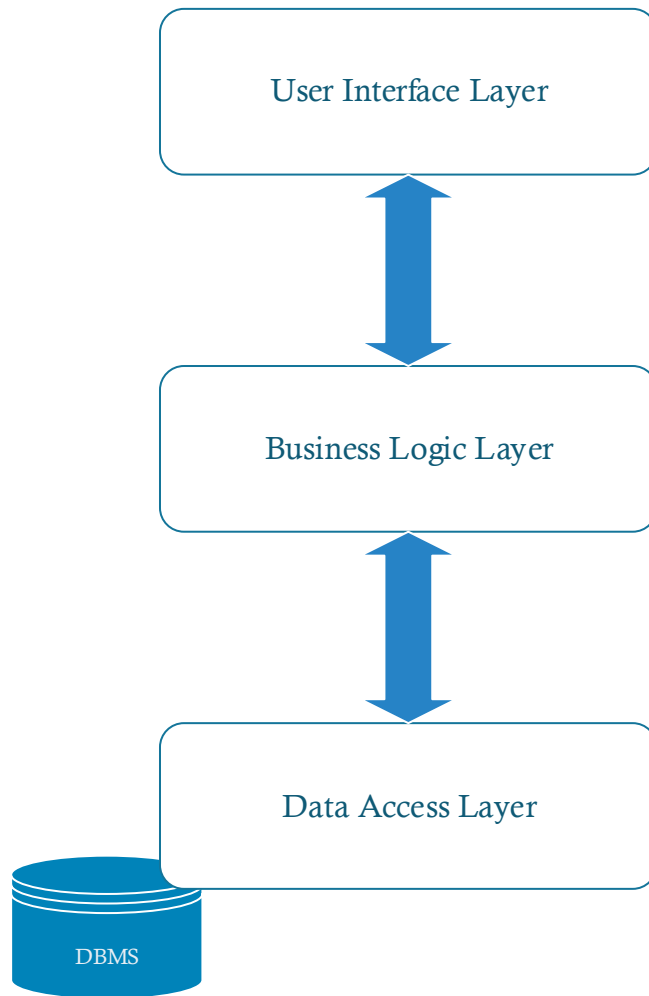
## **22 Current Software Architecture**

There is no current software architecture; however, bellow is the proposed software architecture.

## **23 Proposed Software Architecture**

### **23a Overview**

The architecture model will compose of three tier design for Age of Battleships project. It entails user interface layer, the application business logic layer and the storage layer for data bases. The User Interface will provide a means for the user to interact with Age of Battleships game. It will display appropriate information for the user to make progress in the game and it will prompt the user when user input is require to make advancement in the game. The application business logic layer contains the logic to check against if the user information entered is correct, if the user's move to shoot at target was hit or miss and it will also validate the user's answer for every history question. The data access layer will deal as interface for the application to interact with data base management systems. The storage layer also contains up to date DBMS server address path along with require credentials to connect to the remote server to fetch and update with data bases. The User interface layer shall not interact with database server, hence it must go through business layer and business layer will connect through data base access layer in order to retrieve or update information with DBMS server. The three layer model is shown below:



**Figure 7 Three tier architecture diagram**

### **23b Class Diagrams**

By large Age of Battleships contains three design patterns as mentioned in details below:

#### **1. Factory Method Design Pattern**

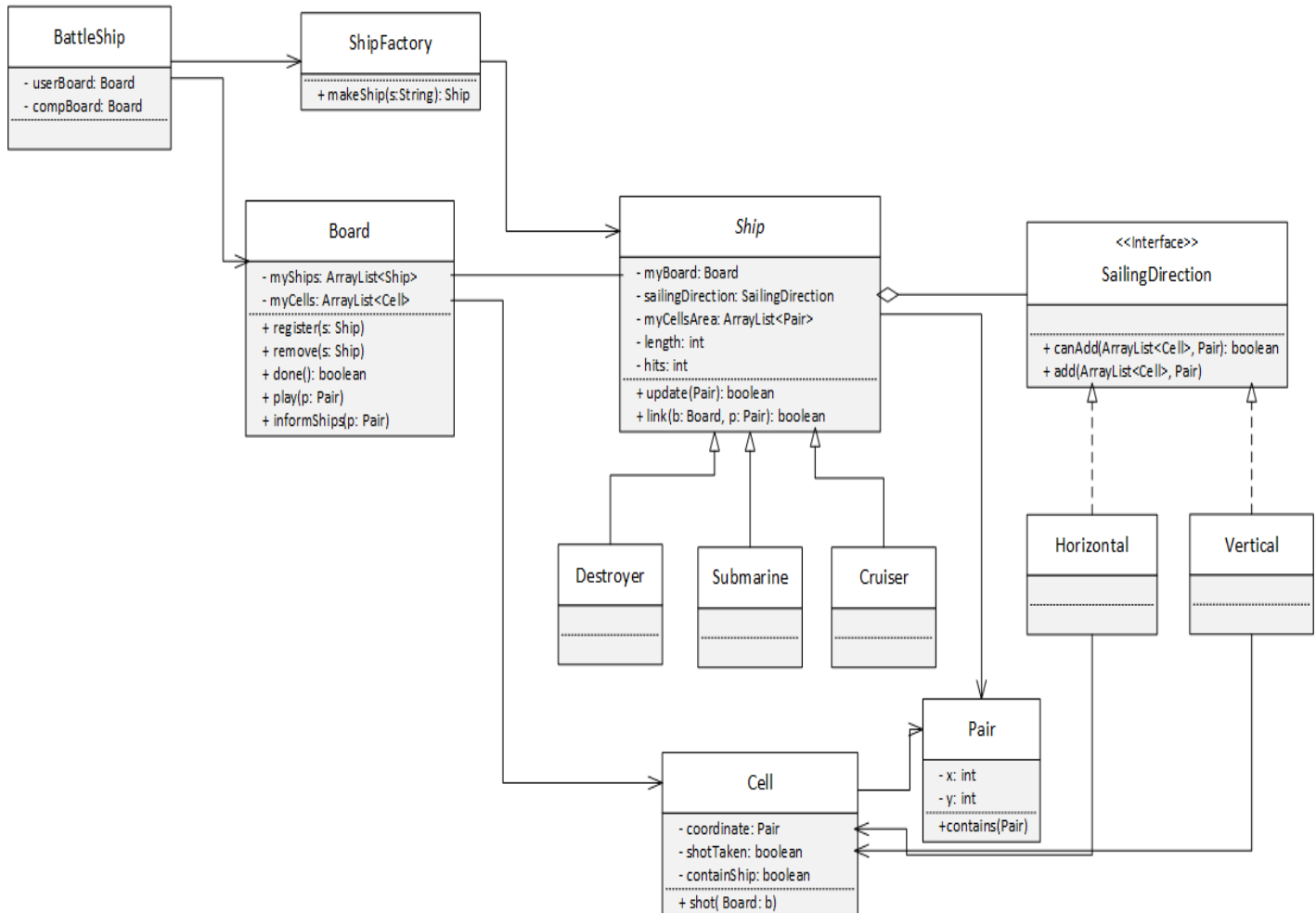
Class ShipFactory provides a makeShip() method for various inherited Ship child object creation. Object BattleShip uses makeShip() method with the type (string) to specify the kind of Ship class it request to be created.

#### **2. Strategy Design Pattern**

Ship class encapsulate SailingDirection interface to use different behaviors (horizontal/vertical) at run time for add() method. At object creation for Ship class the sailing direction is passed as string argument to the Ship constructor to instantiate Horizontal or Vertical sub class it composite to get appropriate add() method behavior. Therefore the appropriate add() method polymorphic, invoked at run time.

### 3. Observer Design Pattern

Class Board plays as subject in observer design pattern and maintains a list of Ships objects, which play as observers. Ship class register with Board class by calling Board class's register() method at the start of the program. As the battleship game progress and both players take shots at different coordinate, whenever an object Cell takes a hit at coordinate and has containShip field true, it informs the subject (Board)

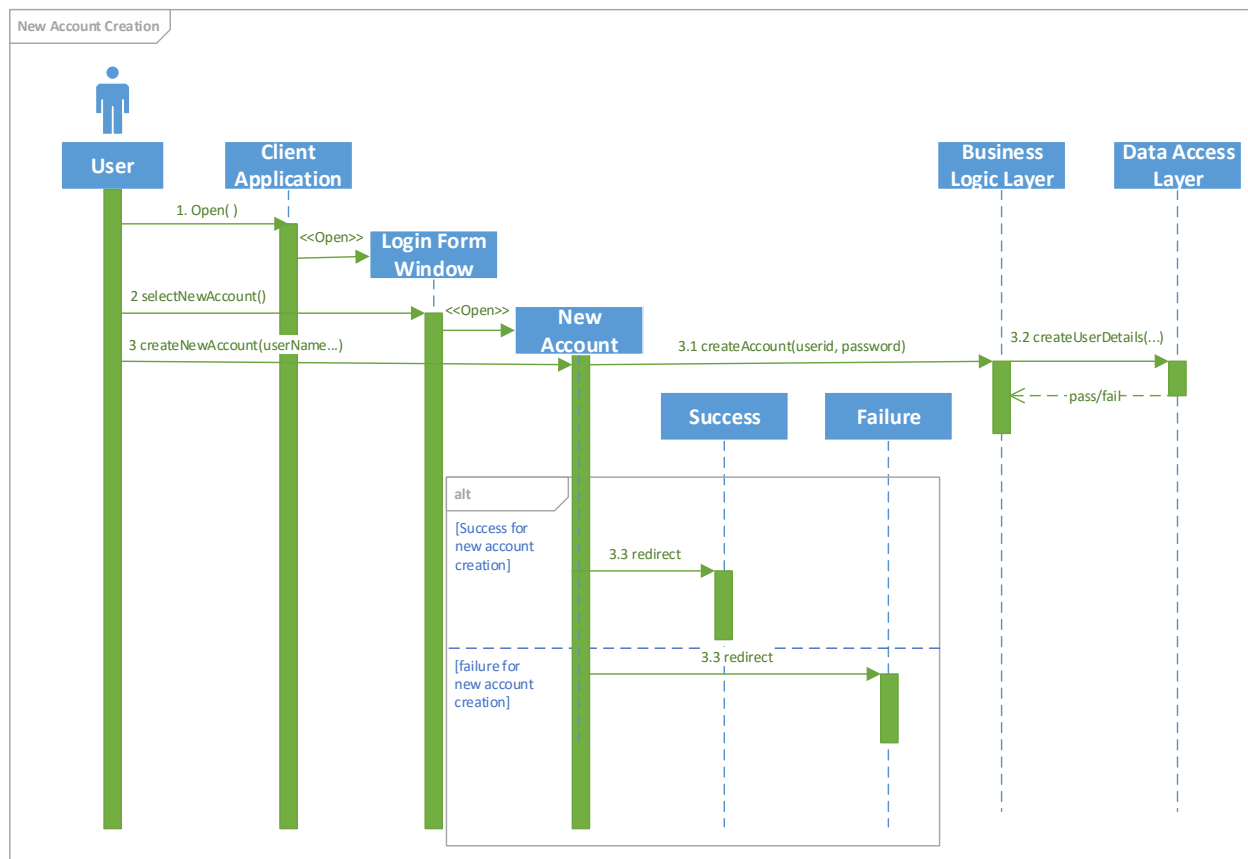


**Figure 8 UML class diagram**

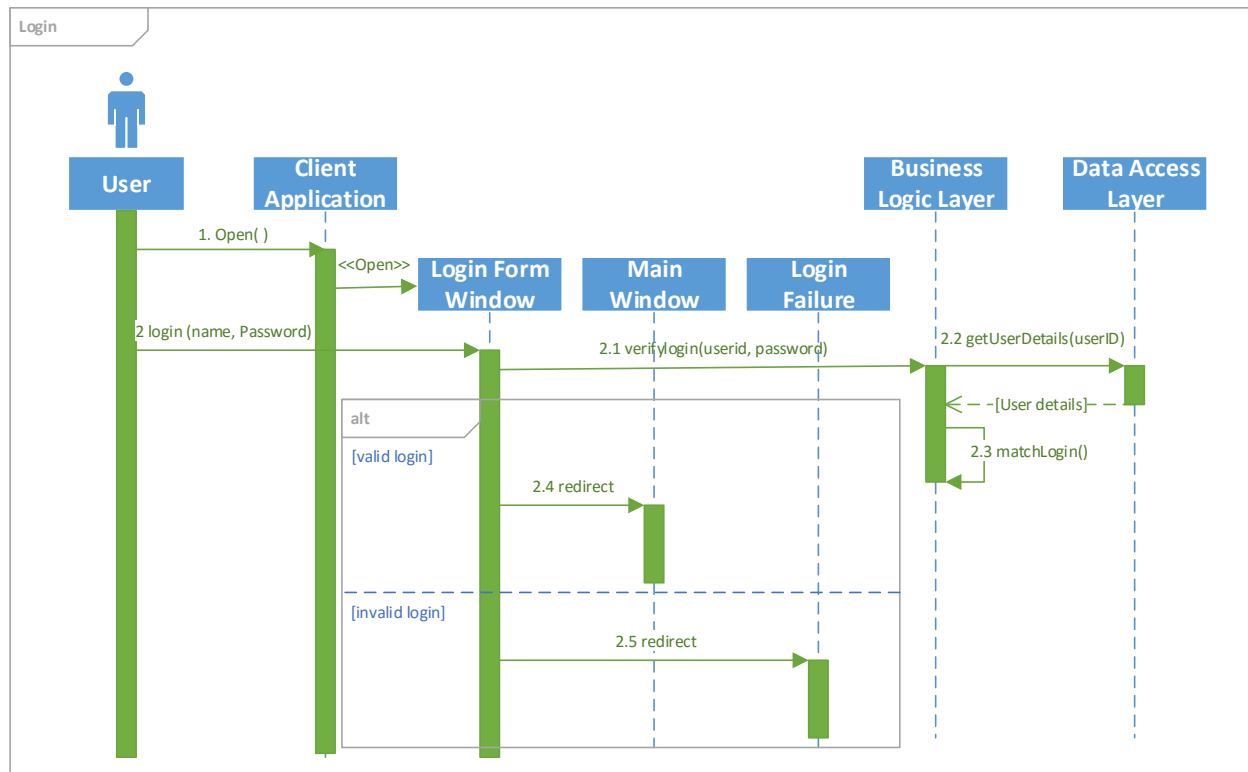
to update all the observers (Ship) with the x, y coordinates. Board class use Ship class's update() method to notify all the ships in its list about the coordinate that took the hit. In return, Ship class check if it contains a reference to that coordinates and decrement its hits field accordingly. When Ship class takes a hit and finds that hit field is equal to zero (initially every ship has hits = shipSize), it notify the subject (Board) to remove itself from Board's list that contains all the observer (Ship) as it has sunk.

## 23c Dynamic Model

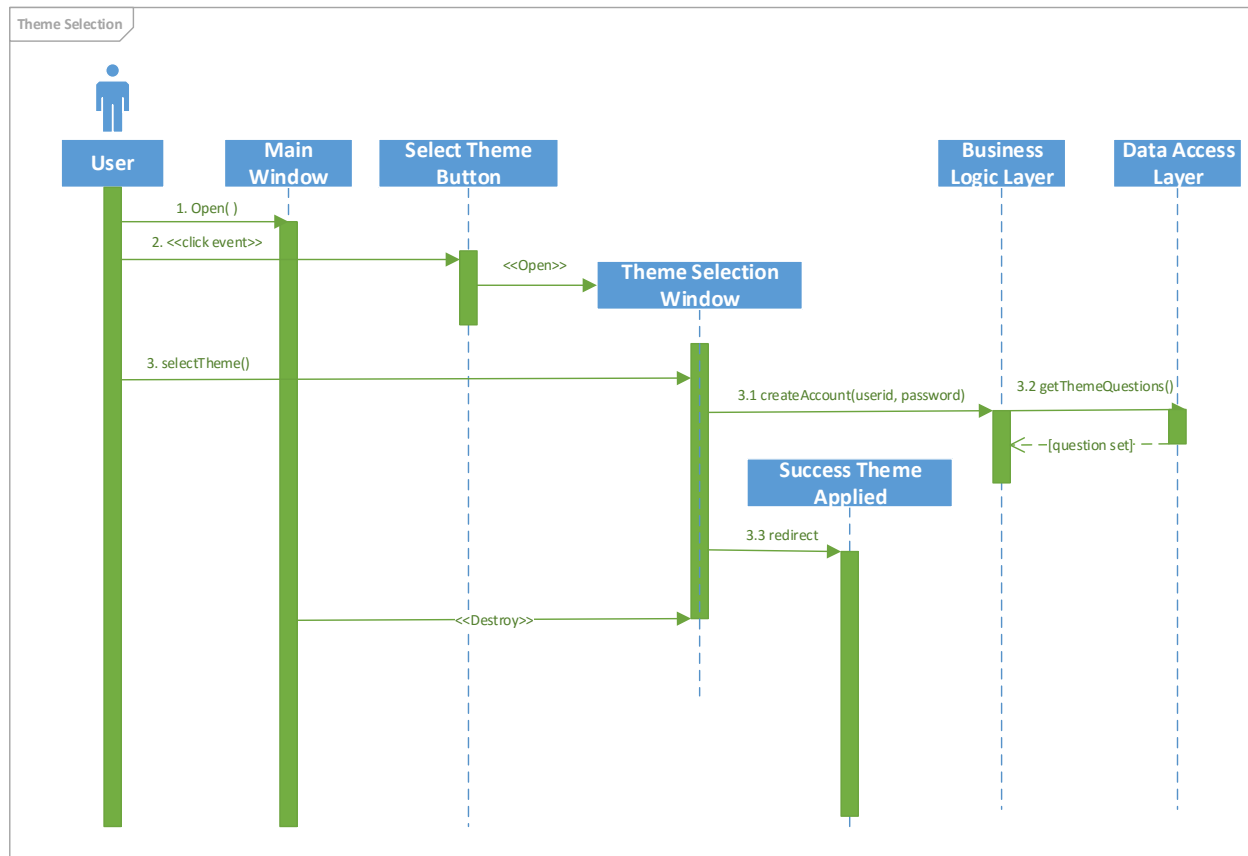
The sequence diagrams for each use-case are diagramed below.



**Figure 9 New Account Creation**

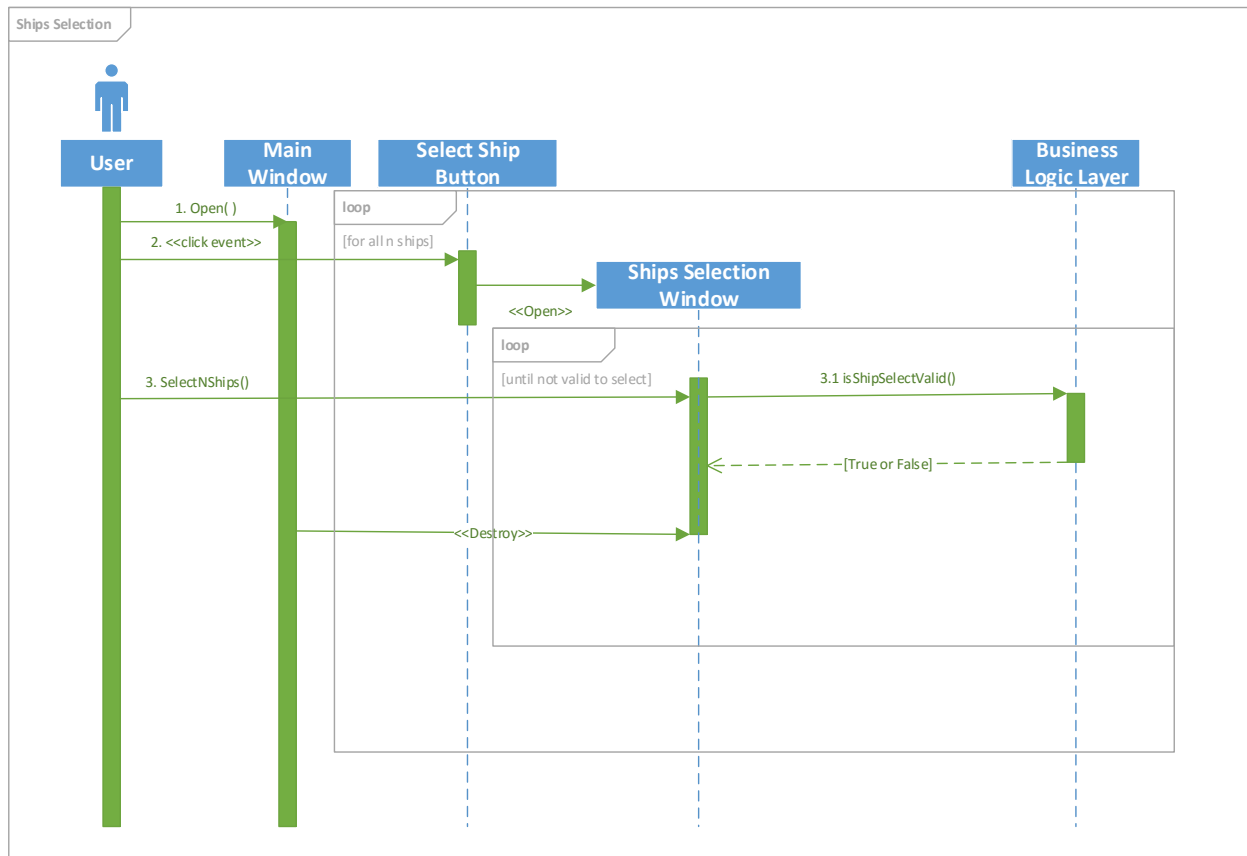


**Figure 10 Login Use Case**

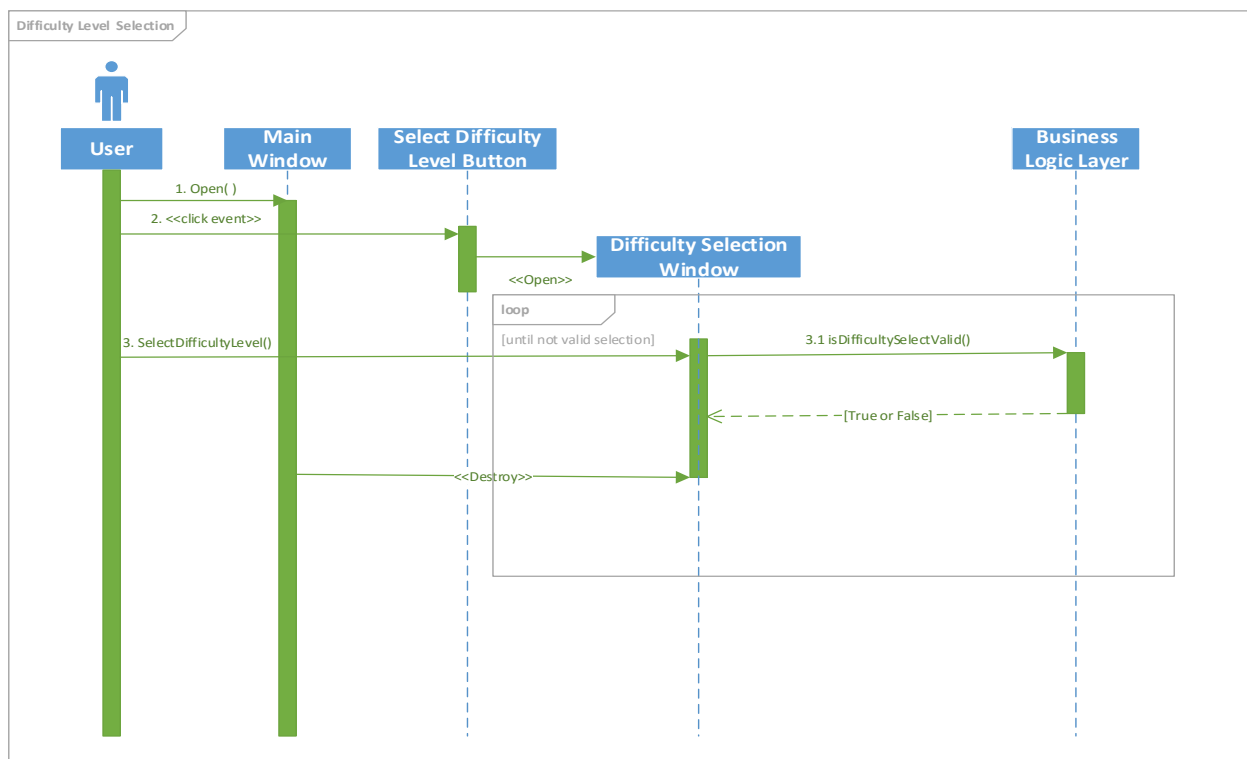


**Figure 11 Theme Selection**

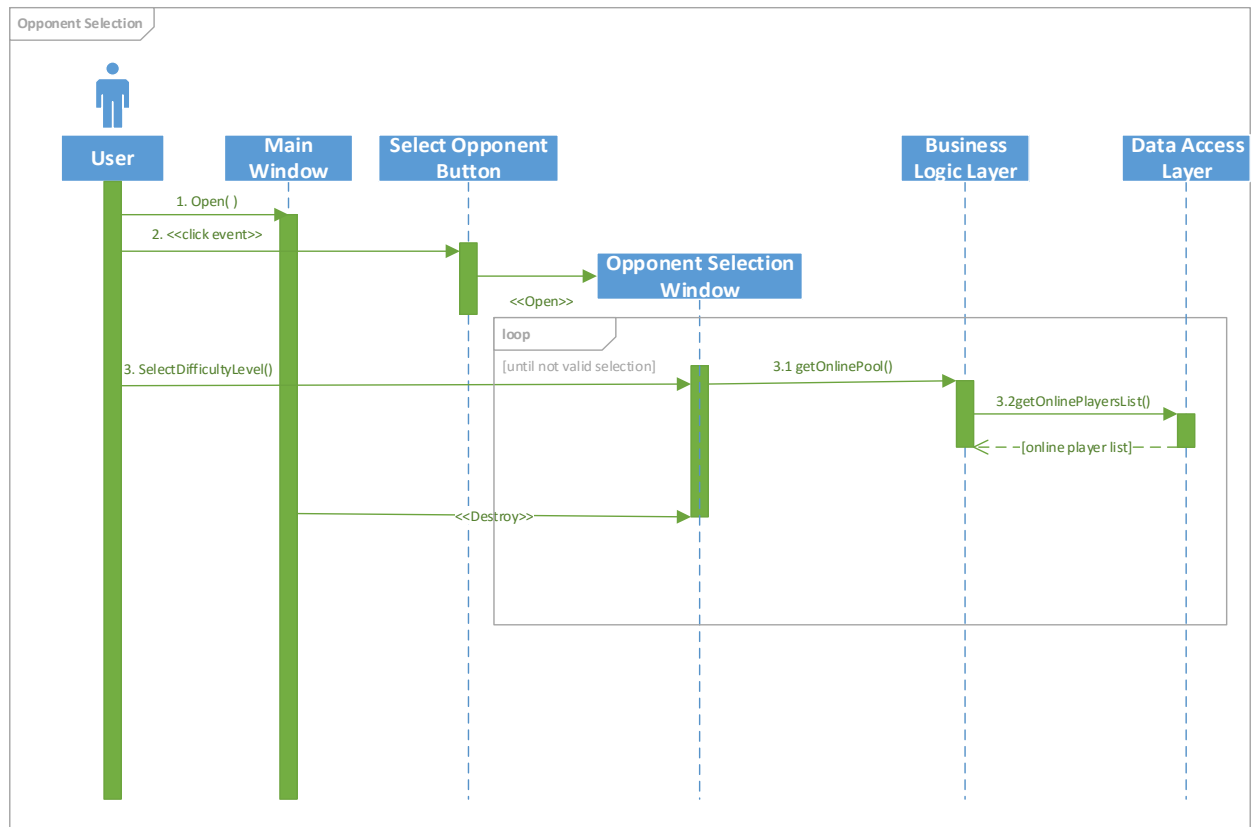




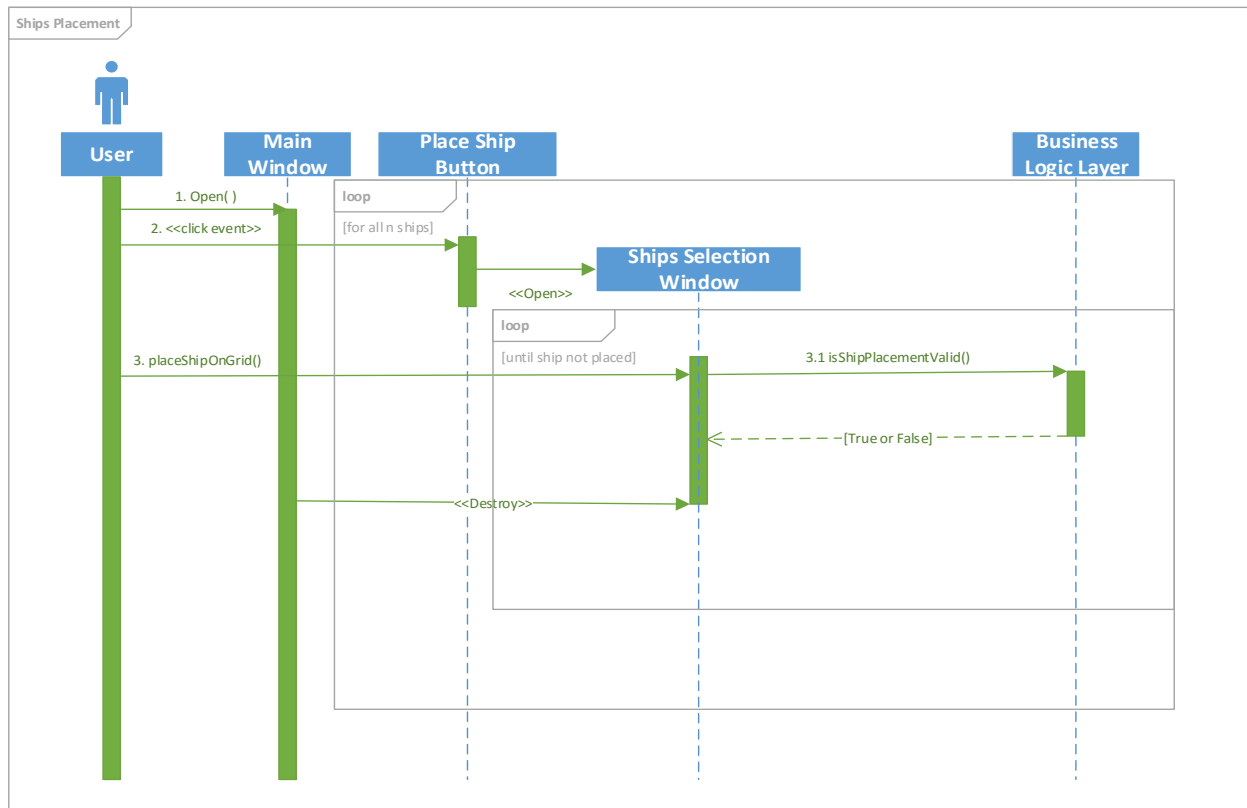
**Figure 12 Ships Selection**



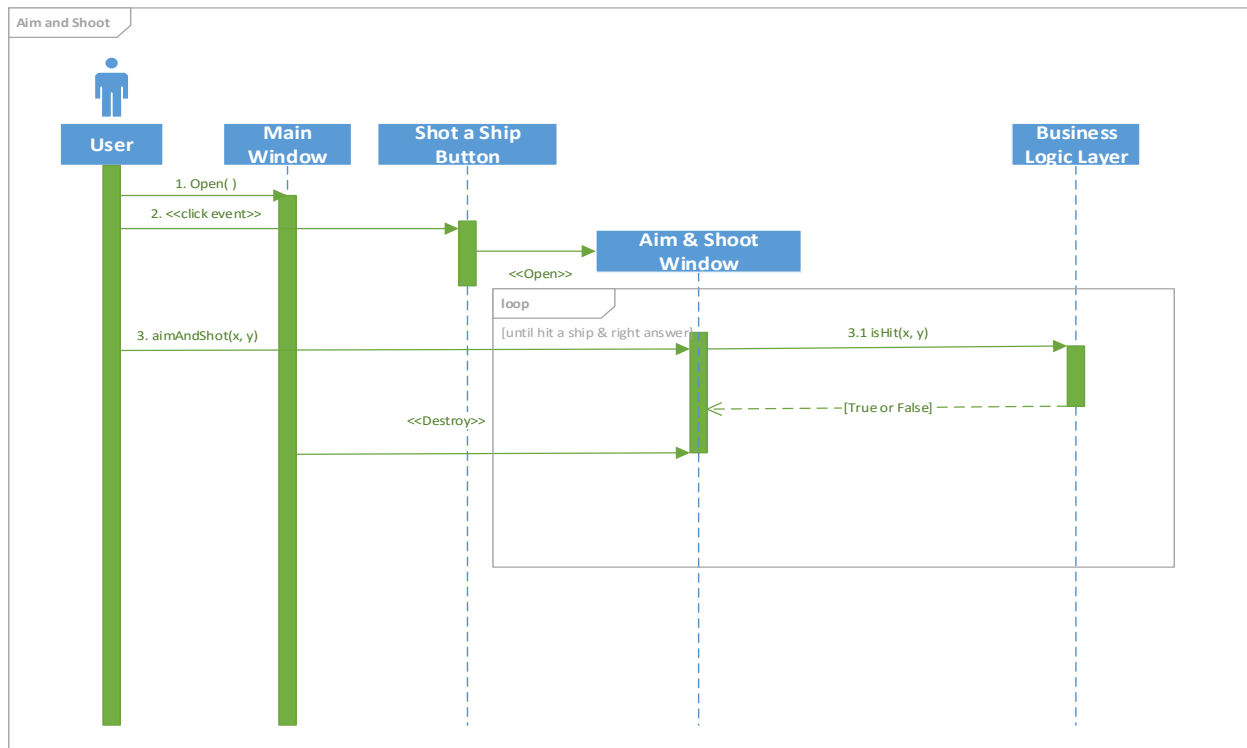
**Figure 13 Select Difficulty Level**



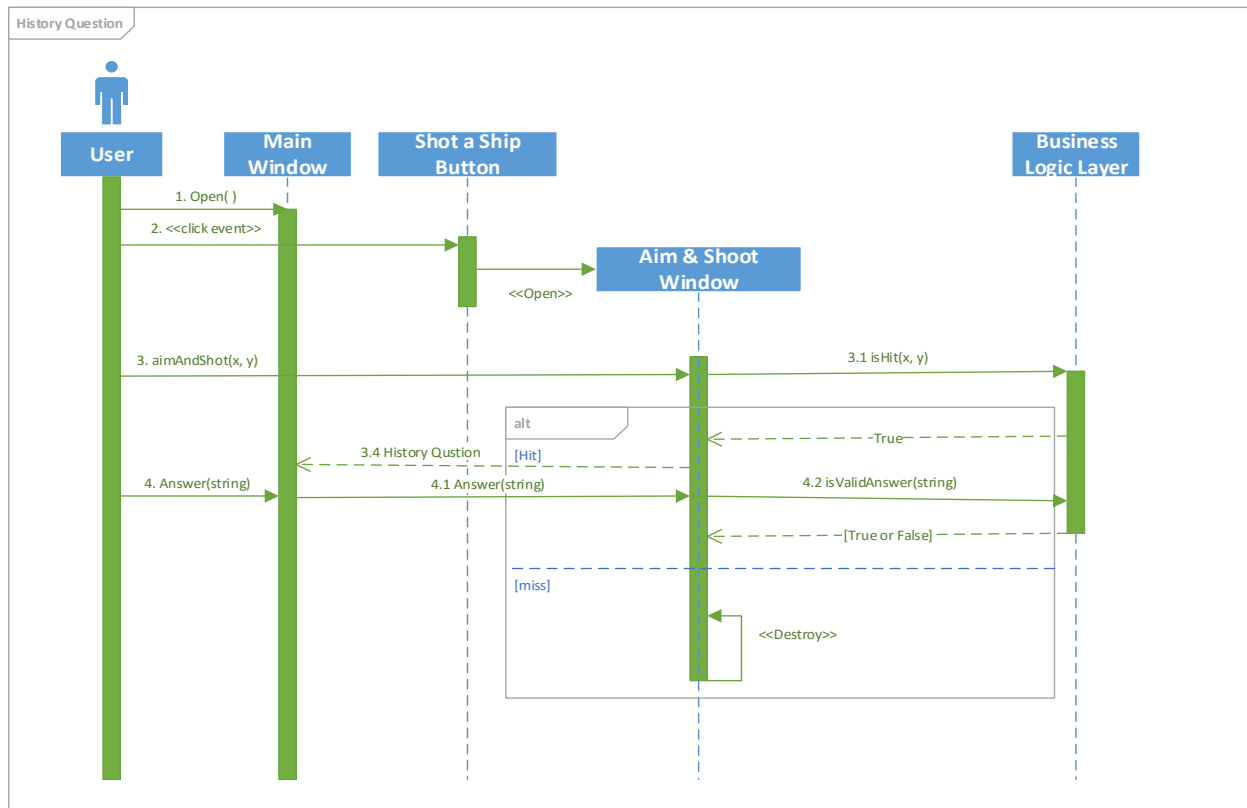
**Figure 14 Opponent Selection**



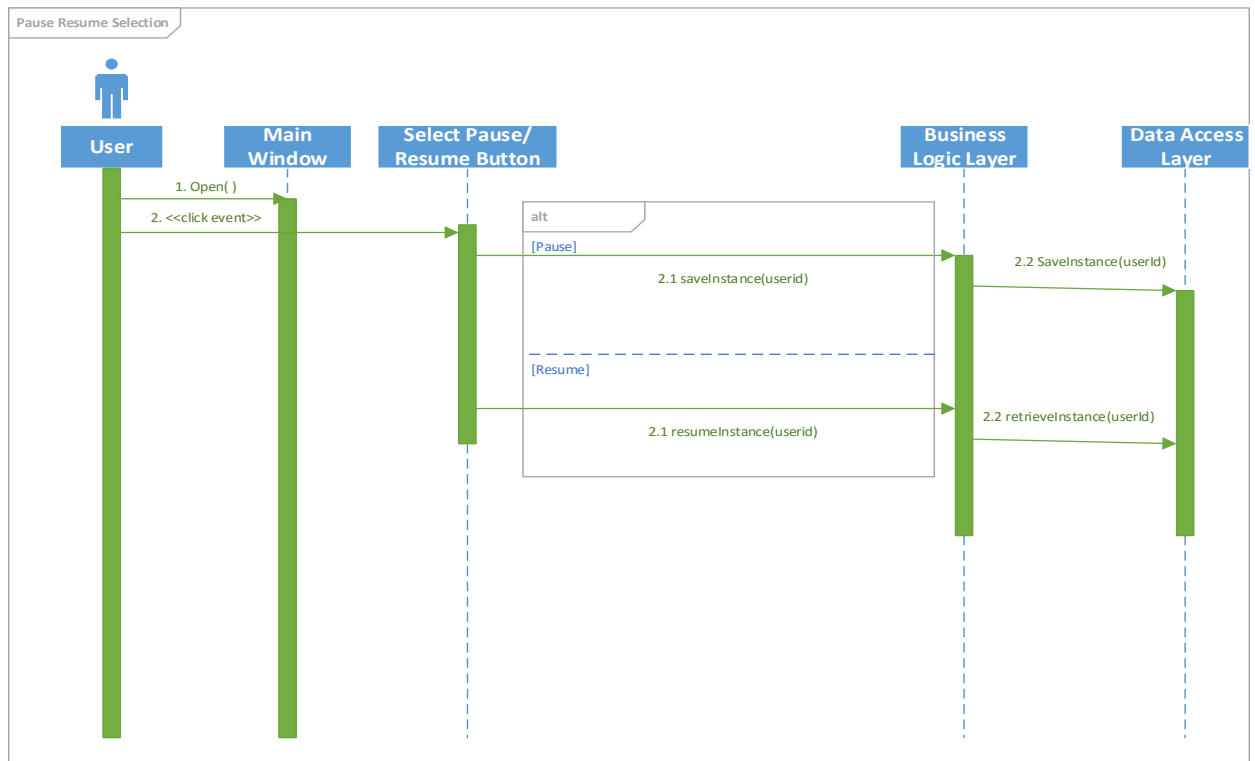
**Figure 15 Ships Placement**



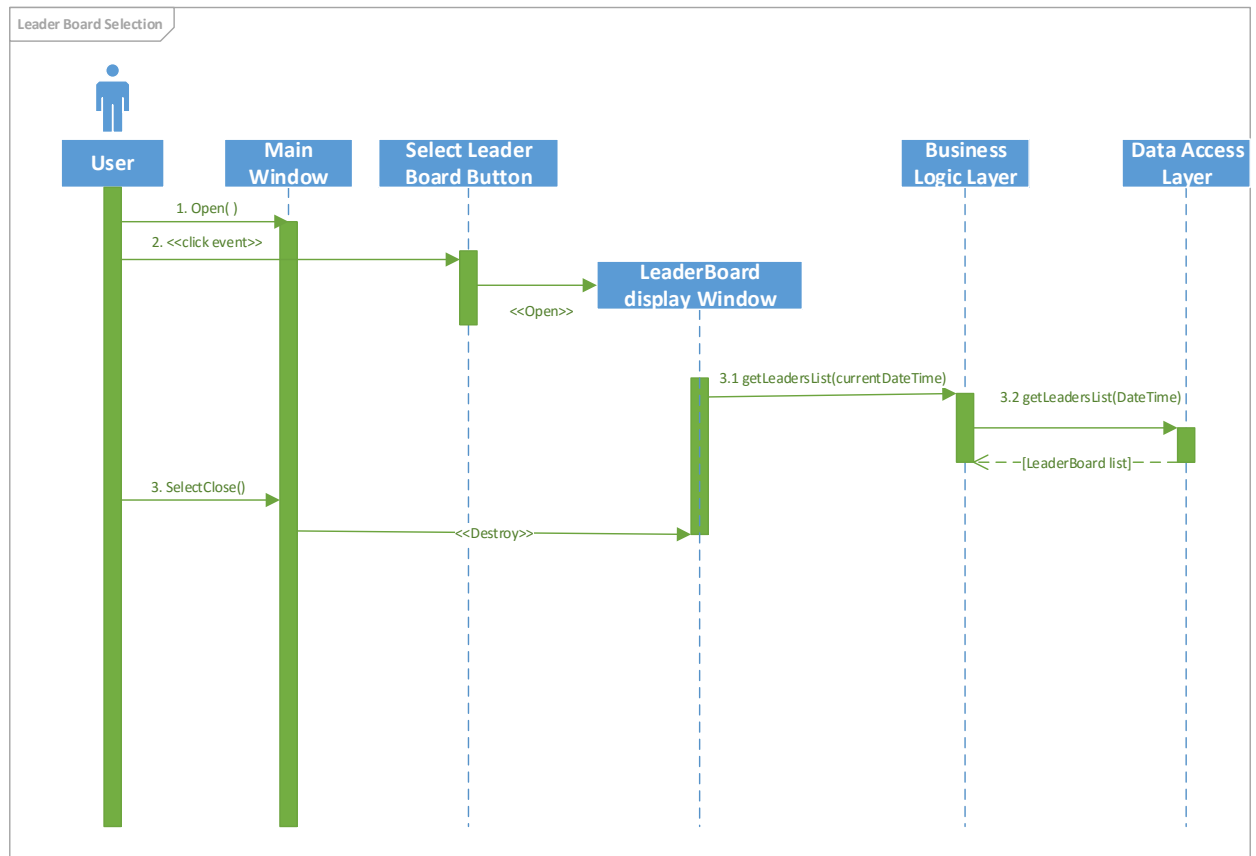
**Figure 16 Aim and Shoot**



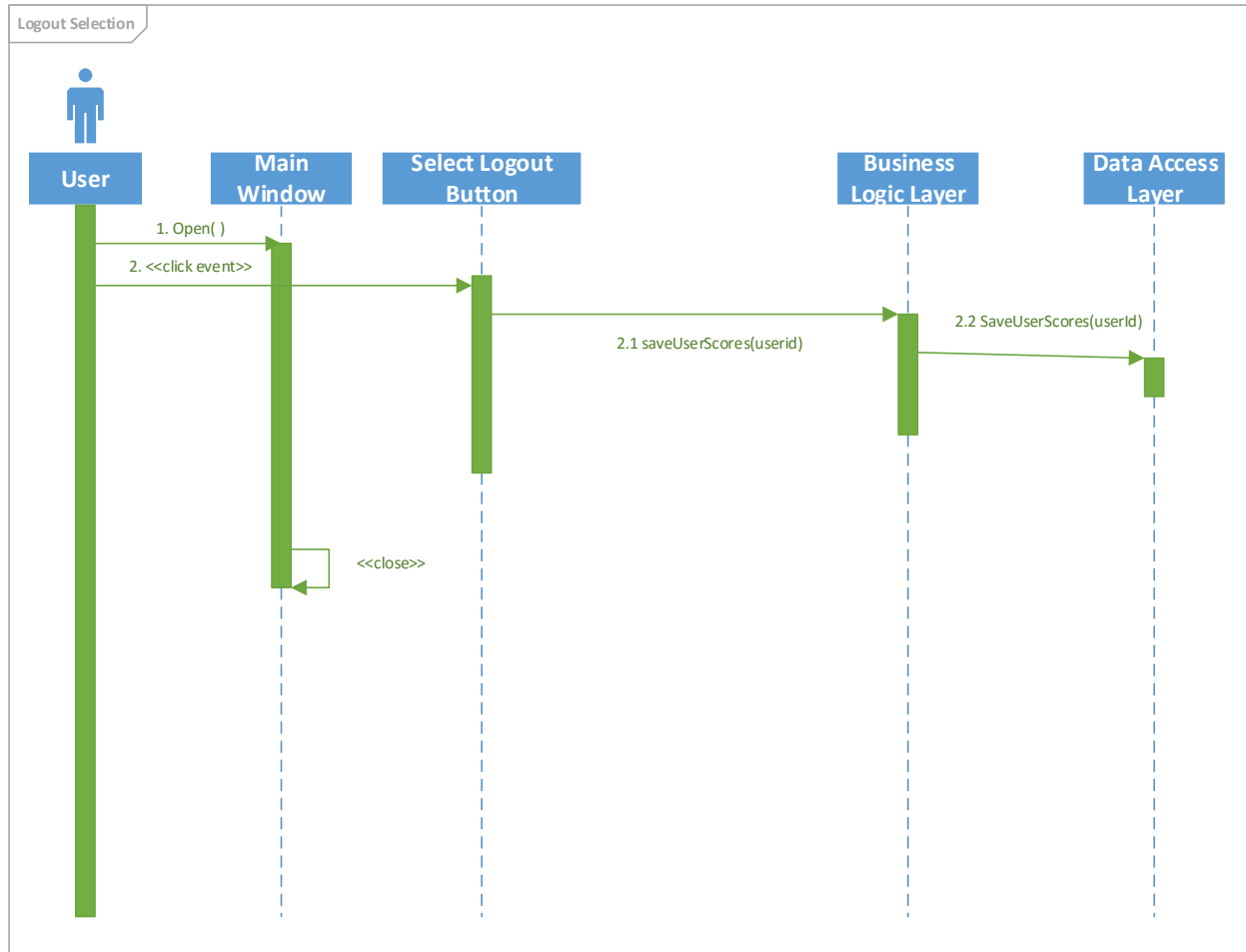
**Figure 17 History Question**



**Figure 18 Pause / Resume Selection**



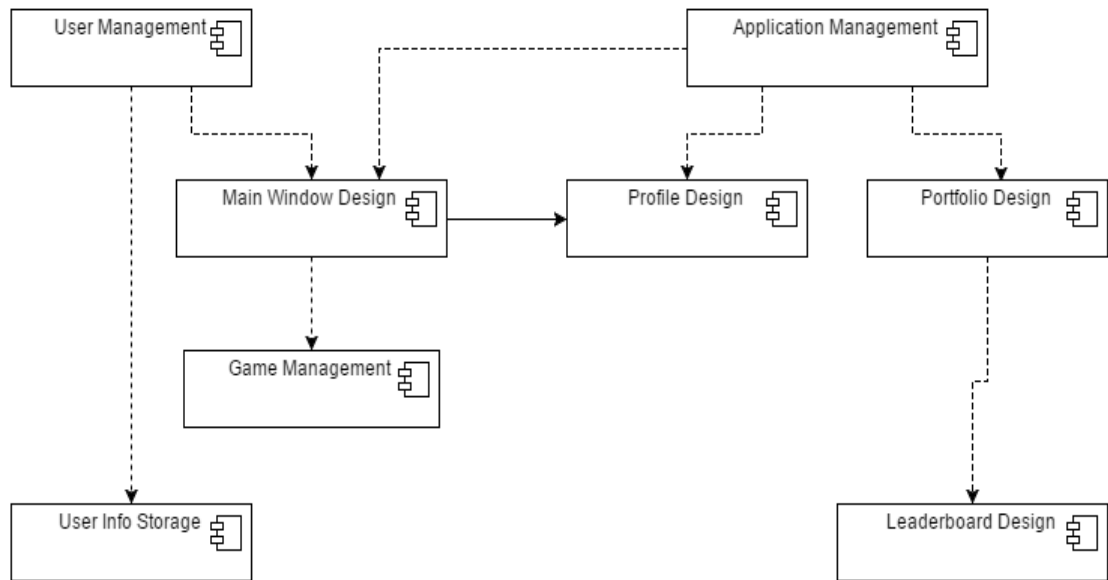
**Figure 19 Display Leader Board**



**Figure 20 Logout Selection**

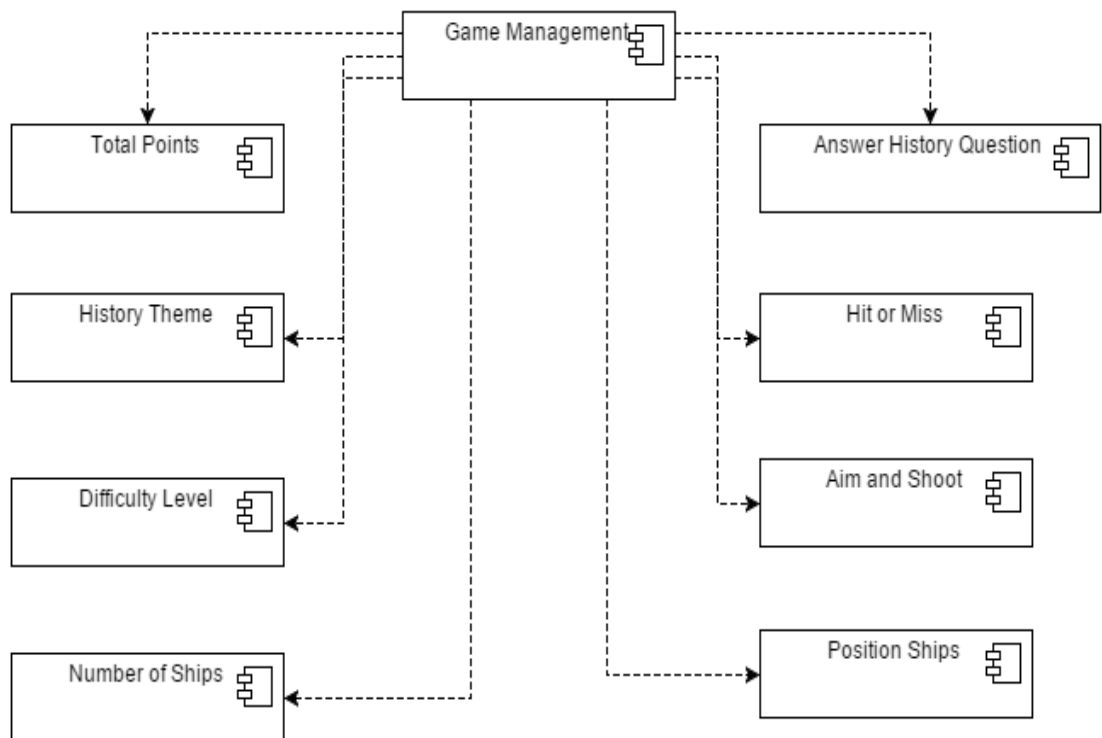
### 23d Subsystem Decomposition

The system decomposition has been divided into two separate parts. The first part is the game organization part and game play design. The game organization part consists of backend setup and design of the game features. It is summarized in the component diagram below:



**Figure 21 Game Organization Subsystem**

The other system describes the features of the actual game play. This includes the features that are at the disposal of the users during the game. Each of the subsystem can be assigned to small group of programmers or an individual programmer. The game play subsystem is summarized with a component diagram below:



**Figure 22 Game Play Design Subsystem**

## 23e Hardware / software mapping

The Age of Battleships is built upon a web server. The server will be responsible for storing the information of the user and the total number of wins and losses based on a particular historical theme.

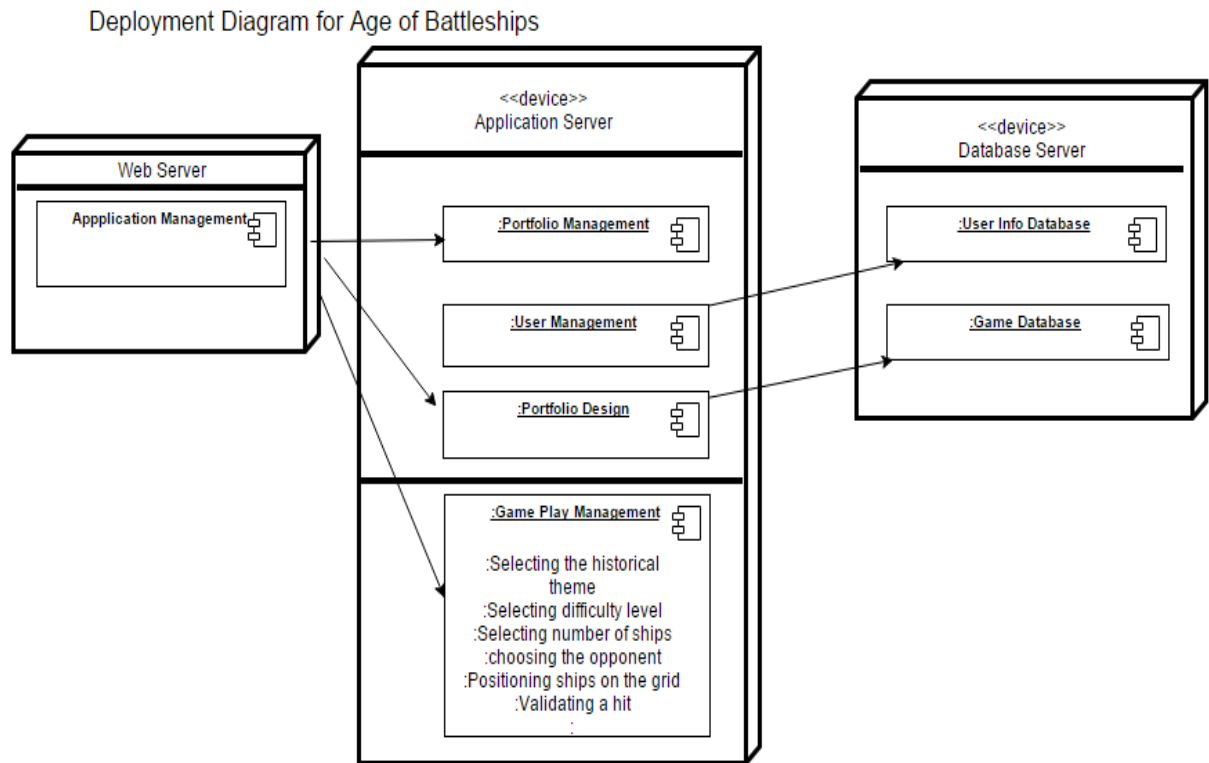


Figure 23 Deployment Diagram

## 23f Data Dictionary

### Miss

When a user shoots a shot on the opponent's ship and it misses the opponent's ship it's called a miss. It is denoted as white dot in the game to the corresponding cell of the opponent's grid.

### Hit

When a user shoots a shot on the opponent's ship and it hits the opponent's ship it's called a hit. But the hit is not validated unless and until the user answers a particular question related to the historical event the user has selected. It is denoted as yellow dot.



**Validated hit**

When a user shoots a shot on the opponent's ship and it hits the opponent's ship it's called a hit. After hitting the user will be prompted a question related to historical event the user has selected. If he answers it correctly then the hit is validated, but if he answers it wrongly then it's considered a miss. A validated hit in the game is denoted by a red dot.

**Knockout (KO)**

If player Roger destroys all of the opponent's ships, it means that the player Roger has KO the opponent.

**Clean Sweep**

A clean sweep is a situation in which a player destroys all the enemy ships with none of the player ships getting destroyed in the battle.

**Battle Report**

At the end of the game each player will be presented with a battle report which depicts the total number of ships destroyed by the player along with the Accuracy rate

Accuracy Rate= (Total number of validated hits / Total number of shots fired)\*100

**Career Report**

A player can view his career report, which includes total number of games won by the player, total number of questions answered, total number of ships destroyed and his ranking on the leaderboard.

**23g Persistent Data management**

The persistent data includes the login information of the users that is entered into the database once the user signs up for the game. This persistent data is stored in the database. The database that will be used is a relational database. The database will be kept at a remote location and will only be accessible to game administrators.

Another persistent data that exists for this game is the career report of the players, which includes the total number of games won by the player, total number of questions answered, total number of ships destroyed and the player's ranking on the leaderboard.

## 23h Access control and security

### Access Control Matrix:

Objects \ Actors	Game Database	User Database	Profile/ Portfolio	Leaderboard	Question Sets
Player			<<Create>>	viewAnytime()	
Game Administrator	Update()	Update()		viewAnytime()	Update()

## 23iGlobal software control

The control flow that will be selected for Age Of Battleships will be the event-driven control flow paradigm. Once the user requests some operation by clicking on a button, an event will become available and it will be dispatched to an appropriate object, based on the information associated with the event. This can include clicking on the “Login” button. Once the user clicks on the “Login” button, an event will be fired and an appropriate object will be dispatched which will result in a Login form appearing to the screen. This will be the case for almost all of the features of the game. Once the user clicks something, an appropriate event will be fired, resulting in a new window appearing or a new action taking place.

## 23jBoundary conditions

The boundary conditions of the system specifies how the system is started, initialized and shut down and we need to define how we deal with major failures such as data corruption and network outages.

**Startup and Shutdown:** Each game is initialized and started as the player connects to the game server. The player can terminate the game instance by choosing the Exit option from the menu or closing the session forcibly by clicking the cross button on the top right corner of the application.

**Exception Handling:** The Age Of Battleships game is network dependent. Hence, we have to take care of network outages. So, in case of any such failure, the system should be able to save the state of the game for a particular player and waits for the player to reconnect within 10 seconds. If the player does not connect within 10 seconds, then the game is dropped. If the player connects within 10 seconds, the player should be able to restore to its previous state of game from where he left. We need to handle these exceptions carefully, since our whole game is network

dependent for most of its resources. Subsystem responsible for database management should be able to perform consistently checks and repair corrupted data, as necessary.

## 24 Subsystem services

- **Application Management:** Application management consists of maintaining the application and providing latest updates and prompting the user when an update is available so that the users have latest version of the game with them.
- **User Management:** The user management consists of keeping all the features of the game up to date for the users. The new user must be able to register for the game and the returning users must be able to login by supplying their username and password.
- **Main Window Design:** The welcome window shall contain the sign-in, the login and the exit buttons for the users.
- **Profile Design:** The users shall be given to view their profile as well as access their portfolio after they login or signup.
- **Portfolio Design:** The portfolio shall be defined in a way that allows user to view the total points he has earned and total number of wins and losses based on a particular history theme.
- **Portfolio Management:** The portfolio should consist of options to position ships by the user anywhere in the grid at the beginning of the game. If a player hits an opponent ship then he can validate the hit by answering a particular question related to the historical theme that the user chose in the beginning. Depending upon the difficulty level a player can also move his ships if his opponent answers a particular history question incorrectly multiple times
- **Game Management:** Game management makes sure all the features of the game are working properly at all times. The features include selecting the history theme, selecting the difficulty level and the number of ships, choosing the opponent, positioning the ships in the grid, validating a hit in case the player hits opponent's ship, keeping track of leaders through leaderboard etc.

## 25 User Interface

Below is the preliminary design for the user interface.

The Welcome Screen:



The image shows a login interface for a game titled "Age Of Battleships". The title is centered at the top in a large, bold, black font. Below the title, there are two input fields: "Username" and "Password", each with a corresponding label to its left. Below the password field is a "Sign In" button. Further down, there is a link "New to game ?" and a "Sign Up" button below it. The entire interface is set against a light gray background with a subtle gradient.

**Figure 24 Login User Interface**

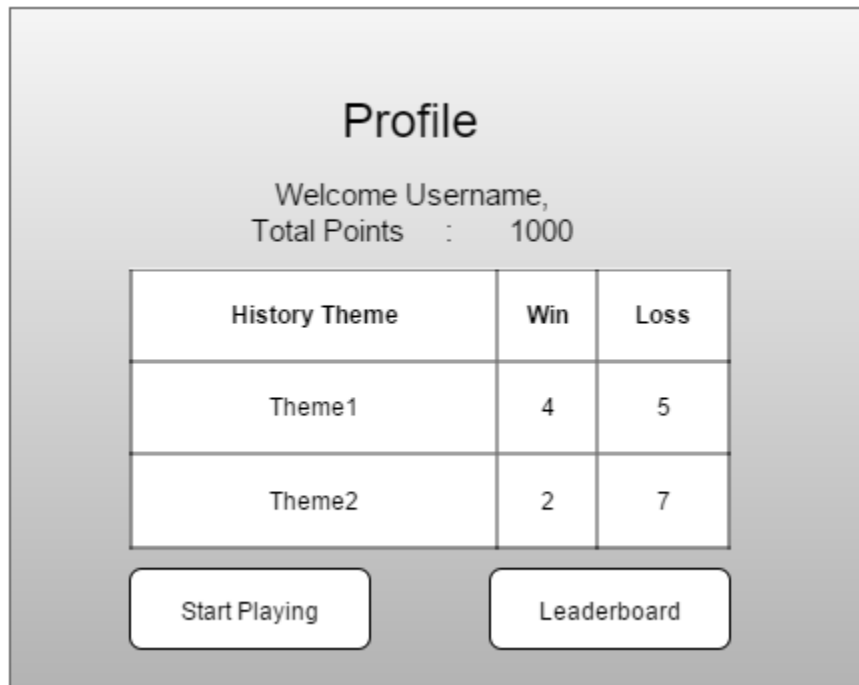
The following image shows the sign up screen after the user clicks “Sign Up”.



The image shows a sign-up interface. The title "Sign Up" is centered at the top in a large, bold, black font. Below the title, there are three input fields: "Username", "Email", and "Choose Password", each with a corresponding label to its left. Below the "Choose Password" field, there are two buttons: "Sign Up" and "Exit". The entire interface is set against a light gray background with a subtle gradient.

**Figure 25 Sign Up User Interface**

The following screen appears after the user logs in.



The Profile screen displays a welcome message and a table of game history. Below the table are two buttons: 'Start Playing' and 'Leaderboard'.

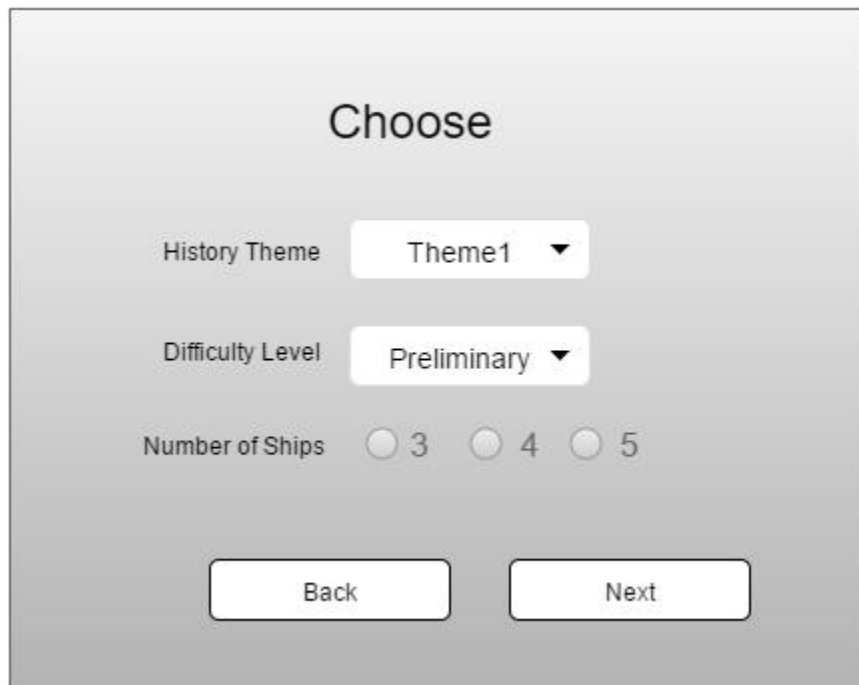
**Profile**

Welcome Username,  
Total Points : 1000

History Theme	Win	Loss
Theme1	4	5
Theme2	2	7

**Figure 26 Profile User Interface**

The following screen appears after the user clicks “Start Playing”



The Choose screen allows the user to select game options. It includes dropdown menus for 'History Theme' and 'Difficulty Level', and radio buttons for 'Number of Ships'. At the bottom are 'Back' and 'Next' buttons.

**Choose**

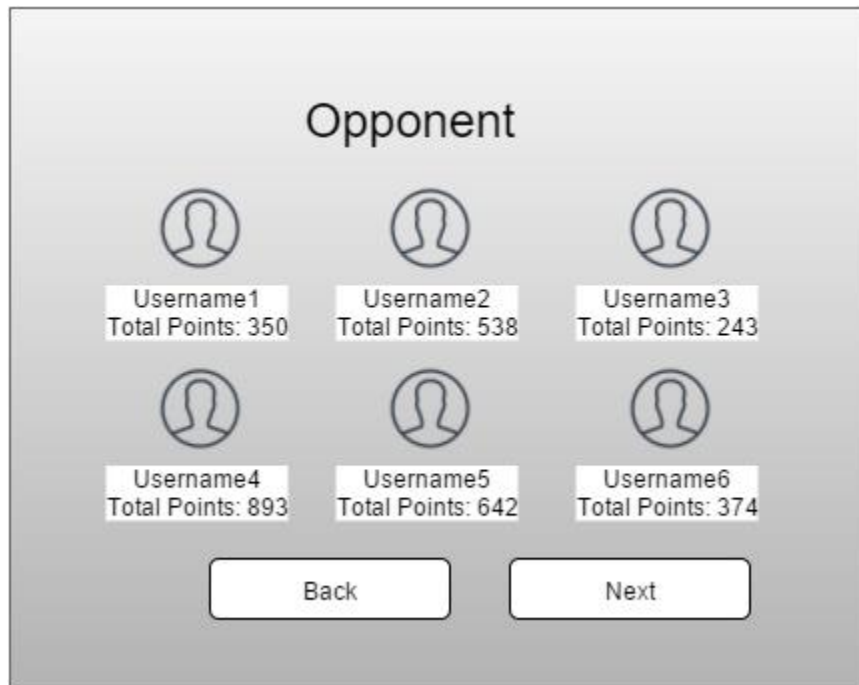
History Theme

Difficulty Level

Number of Ships ☐ 3 ☐ 4 ☐ 5

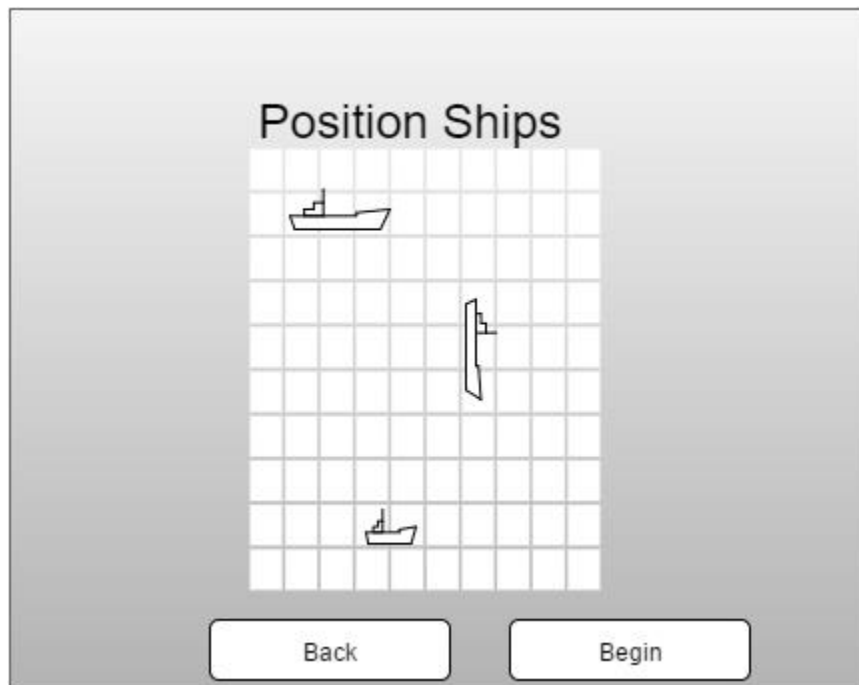
**Figure 27 Choose Options User Interface**

The following screen appears after the user clicks “Next”.



**Figure 28 Select Opponent User Interface**

The following screen appears after the user clicks “Next”.



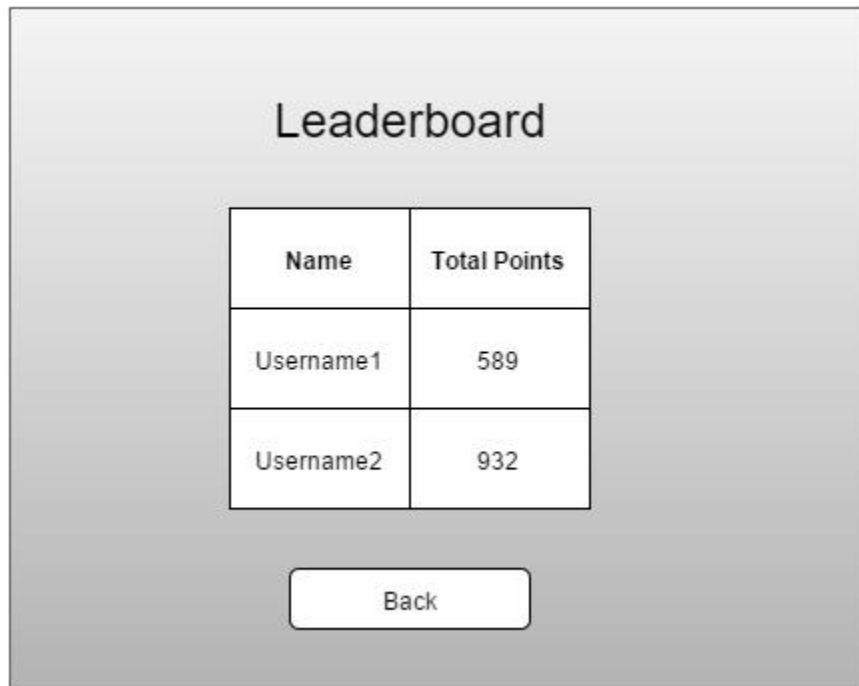
**Figure 29 Position Ships User Interface**

The following screen appears after the user clicks “Begin”.



**Figure 30 Game Board User Interface**

The following screen appears when the user clicks “Leaderboard” from the Profile page.



**Figure 31 Leaderboard User Interface**

## **26 Object Design**

### **26a Object Design trade-offs**

- **Changes vs Modification:** Reusability is not the main concern for designing this project but open to changes and close to modifications. Therefore the object design and design pattern embrace to the change in this project. Any changes in the project will not force too much modification to another class or layer in the multi-tier design.
- **Functionality vs Usability:** It is very important to have wide range of users for this game. Therefore, the game will have plain usage. The system should not be too complex to play. It means the functionality of the system will be basic.
- **Space vs Speed:** Space is notably inexpensive and as such is more expandable than speed. It is more important that the system is responsive and doesn't take too much time to perform internal operation. Therefore the design of this project emphasis on speed.



## 26b Interface Documentation guidelines

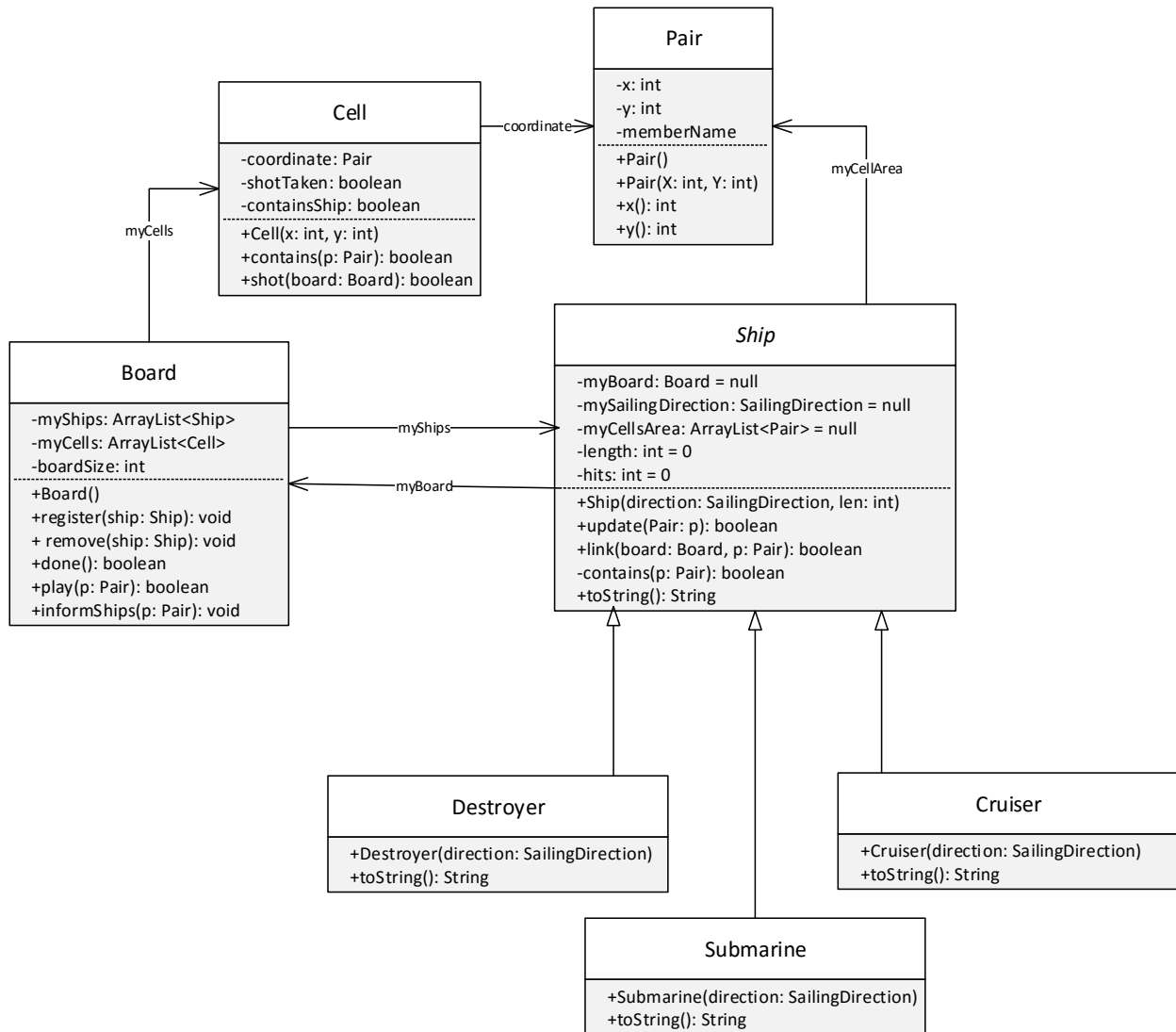
Identifier Type	Rules for Naming	Example
Packages	The prefix of a unique package name is desired and will contain all lower case but the first letter.	Package Ageofbattleships;
Classes	Class name should be nouns in mixed camel case. The first letter should be upper case follow by every noun to start with first letter upper case.	Class Battleship;
Interfaces	Interface naming convention should adhere the same naming convention from classes.	Interface SailingDirection;
Methods	Methods should having verb naming convention, so the receiver of the method has clear understanding for the purpose of the method. Naming verb should not be ambiguous.	IsHit(); IsValid();
Class member field or variables.	All member fields and variables will start with lower case letter but global variables. Camel case convention should also be followed in naming conventions for variables. Common and ambiguous names i.e. x, y, z should be avoided at all cost.	String firstName;
Constants	All constant variables local and/or global should be all upper case separated by underscore '_'. underscore '_'.	Const long MAX_SHIPS = 10;

## **26c Packages**

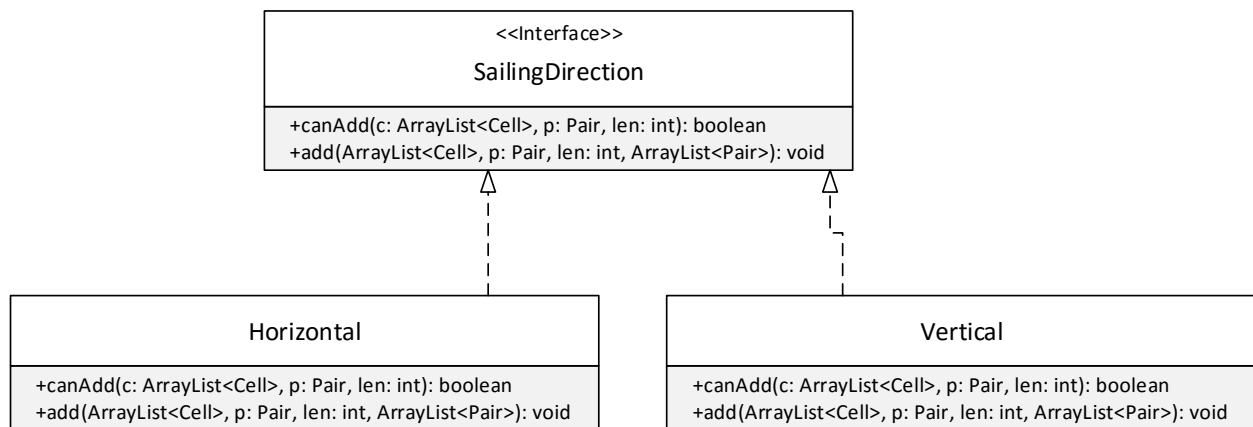
The project will contain one package name across the system. Developers' team are at freedom to choose meaningful project + company they work for as part of their package signature.

## **26d Class Interfaces**

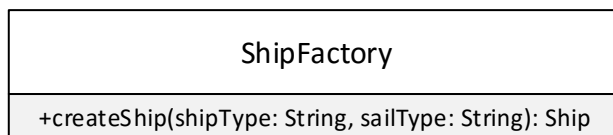
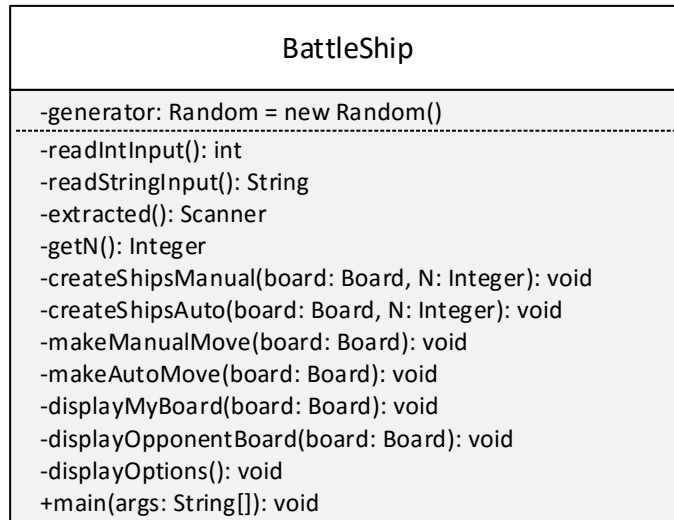
- BattleShip
- ShipFactory
- Board
- Cell
- Ship
- Pair
- SailingDirection



**Figure 32 Class Diagram (i)**



**Figure 33 Class Diagram (ii)**



**Figure 34 Class Diagram (iii)**

## IV Test Plans

### 27 Features to be tested / not to be tested

Age of Battleships must be thoroughly tested to ensure that it operates at a high standard. The following is a list of general features that will be fully tested before the final release.

- New user sign up
- Returning user login
- Choosing a historic event
- Choosing types of ships
- Choosing a difficulty level
- Moving ships
- Displaying correct coordinates of the ship in the screen
- Attacking opponent ships

- Displaying correct result of attacking opponent ships
- Displaying history question in the correct time
- Displaying correct history question relevant to the event chosen at the beginning of the game
- Answering the history question
- Applying the bonus

Features not to be tested are those that reside outside of our control, namely to do with the accuracy of the historic data stored in the database.

## **28 Pass/Fail Criteria**

A test will be considered a pass if and only if the actual results of the test match the expected results specified in the associated test case. Any deviation from the expected results will be considered a failure of that test. A feature or method will pass when it has been subjected to all associated test cases and passes at least 99 percent of the time. Anything less will call for debugging of the associated code.

## **29 Approach**

The testing approach for this project will be efficient. Testing will occur during all stages of development in parallel with other development tasks.

All developers are expected to perform white box path testing and unit testing, and generate associated reports, for any code they submit to the project.

A fellow developer must then inspect submitted code.

Once all code associated with a certain feature has gone through these first steps, integration testing may begin for the future.

Then the features will be black box tested by the associated test cases below.

As features are completed they will then also be run through integration testing to insure they work together.

At various points the User Interface will be subjected to usability testing using the Wizard of Oz strategy, where the ‘Wizard’ plays the part of the AgeOfBattleship database. This is to ensure the interface is user friendly and easy to understand.

### **30 Suspension and resumption**

If at any point in the schedule any part of the test code does not pass the fit criteria, the development team's concern developer must be notified immediately. If the developer is not present, the manager of the development team must be notified. No buggy part of the code shall ever be accepted. Developer should be informed regarding the test case and detail report of pass and test cases along with testing environment. The feature in question must be suspended with immediate action until the development team fixes the bug and code is re-tested.

### **31 Testing materials (hardware / software requirements)**

The game shall be tested with various hardware and software platforms for a reality check for various probable customer platforms.

#### Hardware:

- Dell PowerEdge Servers
- HP Server DL380p and ML350p
- Microsoft Surface Pro 3 and 4
- Apple MacBook Pro
- Amazon AWS Servers
- Microsoft Azure Cloud Servers

#### Software:

- VMWare virtual machines on MAC and Windows
- Parallels virtual machine on MAC
- OS-X Maverick
- OS-X Yosemite
- OS-X El Capitan
- Windows 7 Pro
- Windows 8.1 Pro
- Windows 10 Pro
- Ubuntu 14.04 LTS

- Linux Mint 17.2 Rafaela

### 32 Test cases

Test Name	Entry Condition	Flow of Events	Exit condition
registerUser_Complete	User is on the home screen and is a new user	1. User clicks sign up. 2. User fills in all required fields and clicks done.	User's information is added to the user database and the user is brought to their portfolio page.
registerUser_Incomplete	User is on the home screen and is a new user	1. User clicks sign up. 2. User leaves one or more required fields blank and clicks done.	A dialogue highlighting the missing information is displayed. Info is not added to the database.
registerUser_Duplicate	User is on the home screen and already has an account.	1. User clicks sign up. 2. User fills in all required fields and clicks done.	A dialogue appears explaining that there is already an account with that information. The user is prompted to login.
loginAccount_Correct	User is on the home screen and has an account.	1. User clicks Login. 2. User enters a valid username and password.	The user is logged in and brought to their portfolio page.
loginAccount_Incorrect	User is on the home screen	1. User clicks Login. 2. User enters invalid username and password.	The user is prompted that they have entered an invalid username or password
chooseHistoryTheme	User is logged in.	1. User clicks Play. 2. User is given a list of history themes. 3. User selects any one theme and clicks Next.	The history theme selects by the user is recorded and user is brought to difficulty level selection page.
chooseDifficultyLevel	User is logged in and has selected the history theme.	User selects a difficulty level among Preliminary, Intermediate and Advanced and clicks Next.	Difficulty level selection recorded and the user is brought to opponent selection page.
chooseOpponent_AI	User is logged in, selected	User chooses to play against AI	User selection is recorded and

	history theme and difficulty level.		brought to ship selection page.
chooseOpponent_User	User is logged in, selected history theme and difficulty level.	User chooses to play against another user.	User provided an option to select a player from online pool and then brought to ship selection page.
chooseNumberOfShips	User is logged in, selected history theme, difficulty level and opponent.	User selects any number among 3,4,5.	User has selected number of ships and is brought to the page where user can position the ships.
positionShips	User is logged in, selected history theme, difficulty level, opponent and number of ships.	User positions the ships in the grid.	Position of each ships are recorded and game starts.
aimAndShoot_Hit	The game has started.	User aims at a point in the grid and shoots. User Hits opponent's ship.	User asked to answer a history question related to the theme selected before the start of the game. If answered correctly, hit is validated and opponent's ship is damaged. Otherwise, players swap turns.
aimAndShoot_Miss	The game has started	User aims at a point in the grid and shoots. User did not hit opponent's ship	Players swap turns.
displayLeaderboard_New	User is logged in.	User clicks leaderboard.	The all time leaderboard is displayed.
displayLeaderBoard_year	User is logged in and on the all-time leaderboard page.	User clicks on 'Year' button.	The players who have gained more points in the last year is displayed.
displayLeaderBoard_month	User is logged in and on the all-time leaderboard page.	User clicks on 'Month' button.	The players who have gained more points in the last month is displayed.
displayLeaderBoard_week	User is logged in	User clicks on 'Week'	The players who



	and on the all time leaderboard page.	button.	have gained more points in the last week is displayed.
LogoutAccount	User is logged in.	User clicks the 'Logout' button.	User is logged out and returned to the home screen.

### 33 Testing schedule

<b>Path Testing</b>	Throughout Development
<b>Inspection</b>	Throughout Development
<b>Unit Testing</b>	Duration :1 Week, Begins: 3 Weeks before each release
<b>Integration Testing 1</b>	Duration :1 Week, Begins: 2 Weeks before each release
<b>Usability Testing</b>	Duration : 3 Days, Begins: 3 Weeks before each release
<b>Integration Testing 2</b>	Duration :1 Week, Begins: 1 Weeks before each release

## V Project Issues

### 34 Open Issues

We are continuing to search for other viable sources that we can use to obtain the game history information. Currently, we have decided to obtain the information from the following websites: classzone.com, authentichistory.com, historylearningsite.co.uk, and history.com. However, we are by no means limited to these websites; we believe that there many other sources which provide much more historic information; therefore, these websites are just to start with.

### 35 Off-the-Shelf Solutions

#### 35a Ready-Made Products

Since ready-made products provide lower costs, minimal problems and unparalleled technical support, it would be wise to consider going that route for some software of hardware products during the development of this game. One of the products that can be taken from the market are the type of databases that will be used for this game. Two databases are needed, one for the user information and one for the game information. Microsoft SQL Server, MySQL and SQLite are among the few

candidates for the databases that can be used off the market. It will be the decision of the development team to decide which product to eventually use.

### **35b Reusable Components**

Instead of having to recreate hypothetical data and hypothetical historic information, as mentioned before, we will be using real historic information obtained ideally from the websites mentioned before. This will be a major help for the development team, as they can place more of their focus on the design of the game, instead of having to worry about creating and modifying data throughout. Getting data from the websites mentioned above or other existing source will also require less maintainability compare to self-produced hypothetical data, which will need to be updated after short period of time. In addition, developers will not need to worry about updating the historic information since most available historic information is static.

### **35c Products That Can Be Copied**

As it turns out, there are quite a few other products on the market that are similar to the Age of Battleships game. One such product is the original battleships game. However, the original battleships game is very basic and does not provide the features we will provide which includes moving ships, blocking positions, being educational in history, awarding bonuses, and many other features. Our game will surely change the way people think when playing the game, and it will change the way people formulate strategies. Users with a wider background in history will be the most successful.

## **36 New Problems**

### **36a Effects on the Current Environment**

Since this is a new desktop application, and as such, the current environment will depend on the development team and how they choose to develop this game.

### **36b Effects on the Installed Systems**

The game will have to conform to the current popular operating systems, Windows, Linux, and Macintosh. Although it is up to the programming team, we suggest the usage of Java programming language due to its popularity and portability.

### **36c Potential User Problems**

We hope to design the software to be fun and worth playing, and we hope to make it educational so users can benefit while having fun. Theoretically a user could become

addicted to playing; therefore, we recommend limiting time spent on the software to 2 hours a day.

### **36d Limitations in the Anticipated Implementation Environment That May Inhibit the New Product**

The software is limited by the individual users' hardware. However, since most users in today's age have computers running on Intel Core i3 or equivalent processors as a minimum, and the Age of Battleships is not a graphics intensive game, then the great majority of users should be able to run the game with no limitations. This game does not require a dedicated graphics card, the main processor's capabilities should suffice.

### **36e Follow-Up Problems**

If the average number of logged in users exceed 49,000, then there may need to be expansions done to the user and game database in order to house a larger number of users and run a larger number of games. This could also result in increased server traffic which will require upgrading the server parts to more powerful hardware.

## **37 Tasks**

### **37a Project Planning**

Software Development Life (SDLC) is used in the development of game. It is a structure imposed on the development of a software product. There are several models for such life cycle, each describing approaches to a variety of tasks or activities that takes place during the process. Some people consider a life-cycle model a more general term and software development process a more specific term. For example there are many specific software development processes that 'fit' the spiral life-cycle model. An international standard for software life cycle processes aims to be the standard that defines all the tasks required for developing and maintaining software, details of the lifecycle and approach that will be used to deliver the product. The new product will be developed using SDLC. However some circumstances are unique to a particular product and will necessitate changes to the life cycle. While these considerations are not product requirements, they are needed if the product is to be successfully developed.

### **37b Planning of the Development Phases**

The development of the Age of Battleships is planned in the following ways. First would be the requirements phase in which functional and non-functional requirements would be gathered followed by the analysis phase and design phase. In implementation the coding would be done. It must be taken care that the game should be able to deploy most of the Functional Requirements.

## **38 Migration to the New Product**

### **38a Requirements for Migration to the New Product**

On availability the users who will move over to the new application will not face a lot of migration problems in terms of operating systems. The game will be compatible with windows, Mac OSX and Linux, hence users that already have these devices will be able to run the application efficiently.

One of the main constraints will be that the users will face a roadblock because of the additional hardware that is required and the users with old systems will need to upgrade their system.

We suggest some minimum requirements for smooth operation of the application:

- The minimum RAM required to run the application is at least 2GB.
- At least 1GB of free space should be available on the hard disk.
- The minimum screen resolution for the system should be 800 x 600.

### **38b Data That Has to Be Modified or Translated for the New System**

There are no such specific data transitional requirements for implementation of the game as the game will be in accordance of the existing data resources available on the application. There would be no importing of data from any other previous applications.

However there is an exception in the case of new product updates, the application might require specific data based updates but this will be an automated process and developers creating updates will be required to ensure consistency from internal testing before the updated release is available in the market.

## **39 Risks**

Age of the Battleships will involve risks—namely, the risk of inaccurate metrics, inadequate measurement, inaccurate cost estimating, low quality and low productivity of the software. Risk is only a bad thing if the risks are ignored and they become problems. Risk management entails assessing which risks are most likely to apply to the project, deciding a course of action if they become problems, and monitoring projects to give early warnings of risks becoming problems. Following points will provide a transitory but comprehensive illustration to such causes.

- Inaccurate metrics: inaccurate information regarding history reference to question answer of portion of the game can result in user losing their confidence in the software. We have to constantly remember the fact the core reason for this project is

developing a game that can be used as a learning tool for students and history enthusiasts.

- Inadequate measurement: It is important to understand the target audience for this game. If the question sets are too hard for the students that they are constantly giving wrong answers and keep losing the game, probability is high that they will not want to play this game again in the future. It is important to understand the target audience and also deploy algorithms within software product to encounter such issue to relax the questions' toughness.
- Inaccurate cost estimating: Providing an inaccurate cost to the eventual client regardless it is high or low can deeply harm the project before even it begins. It is important that the current market price analysis should be done to get better estimation to provide more accurate quote to the customer, who is willing to pay for this project.
- Low quality: Low quality software can lead to bad end user experience and eventually bad reviews and disaster for the project to grow more customer base. It is important that this project provide ease of use user interface, means for the customer to provide feedback regarding their experience and opt in option for the user to provide stack dump on every crash to determine the cause. Such issues must be taken seriously and provide solution as soon as possible.
- Low productivity: It is also important to provide something new i.e. feature or change in user interface design to the customer. Humans tend to get bored from using one thing over and over, therefore it is important to bring high productivity for this project and add something fresh and thrilling for the end users for every new release.

## 40 Costs

Age of Battleships will necessitate the following cost factors and it is important to remember this is not hard coded limit set for the project management to not to analysis other various factors since changes are well expected at the time of development of this project:

- Number of input and output flows on the work context
- Number of business events
- Number of product use cases
- Number of functional requirements
- Number of nonfunctional requirements
- Number of requirements constraints
- Number of function points

## 41 Waiting Room

Other requirements that we would like to be implemented in the future releases include:

- Include more animation for the missile attack, sinking of the ships etc, to make the game more appealing.
- Include Challenging feature where the user can challenge any other user even if that user is not currently online.
- Include Experience points which shows how much experience a player has in a specific topic.
- Include the feature of showing the accuracy of hits a player has.

## 42 Ideas for Solutions

In order to make the game more engaging and interesting, the game can be made in such a way that the damage made to the ship can differ based on the difficulty level of the questions that are answered so that it gives extra weightage to the players who answer difficult questions.

A chat feature can also be included which enables user to share and discuss about various topics they come across.

## 43 Project Retrospective

This development project is being developed using the Waterfall Software Development Life Cycle. It was very helpful to divide the report into separate sections so that each section focused on one part of the report, such as requirements, design and testing report. The communication between the group always challenging at the beginning, but we were able to coordinate times to accommodate everyone's schedule. It is essential that the team coding this project also find proper ways of communication that work for everyone in the group early in the development stage. Team members should also openly think about any problem and brainstorm for the solution.

## VI Glossary

The glossary defines terms that may not be familiar to all readers. This is especially important if the document is expected to reach a wide and varied audience, such as school children. The glossary may be placed at either the beginning or the end of the document.

**Flotsam:** Any part of a ship or its cargo found floating on the water, whether it was deliberately or accidentally lost by its original owners.

**Jetsam:** Any part of a ship or its cargo that is deliberately cast off ( jettisoned ) by its original owners, generally in order to lighten the ship, whether it floats or sinks.

## VII References / Bibliography

- [1] "<http://www.historyandheadlines.com/april-7-1945-yamato-biggest-baddest-battleship-barraged-bombs-like-boss/>," 28 10 2015. [Online]. Available: <http://www.historyandheadlines.com/april-7-1945-yamato-biggest-baddest-battleship-barraged-bombs-like-boss/>. [Accessed 28 10 2015].
- [2] J. Bell, "Underwater Archaeological Survey Report Template: A Sample Document for Generating Consistent Professional Reports," Underwater Archaeological Society of Chicago, Chicago, 2012.
- [3] A. Silberschatz, P. B. Galvin and G. Gagne, Operating System Concepts, Ninth ed., Wiley, 2013.
- [4] Robertson and Robertson, Mastering the Requirements Process.
- [5] M. Fowler, UML Distilled, Third Edition, Boston: Pearson Education, 2004.

## VIII Index

This section provides an index to the report. The sample below was generated using the “Mark Entry” and “Insert Index” items from the “Index” section on the “References” tab, and can be automatically updated by right clicking on the table below and selecting “Update Field”. To remove marked entries from the document, toggle the display of hidden paragraph marks ( the paragraph button on the “Home” tab ), and remove the tags shown with XE in { curly braces. }

Design .....	61, 63	Test.....	64, 65
Requirements .....	35, 51, 58		