***Project Report for MindMapper Game***

***Complete Software Documentation for   
MindMapper***

**Prepared by**

**Jesse Palasz, Aleksandar Ilic, Ishaq Khan, Fnu Nousheerwan**

**for use in CS 440**

**at the**

**University of Illinois Chicago**

**February 2019**

**Table of Contents**

[List of Figures](#_3znysh7) 7

[List of Tables](#_2et92p0) 8

[I](#_tyjcwt) Project Description 8

[1](#_3dy6vkm) Project Overview 8

[2](#_1t3h5sf) The Purpose of the Project 8

[2a](#_4d34og8) The User Business or Background of the Project Effort 8

[2b](#_2s8eyo1) Goals of the Project 8

[2c](#_17dp8vu) Measurement 9

[3](#_3rdcrjn) The Scope of the Work 9

[3a](#_26in1rg) The Current Situation 9

[3b](#_lnxbz9) The Context of the Work 10

[3c](#_35nkun2) Work Partitioning 10-11

[3d](#_1ksv4uv) Competing Products 11

[4](#_44sinio) The Scope of the Product 12

[4a](#_2jxsxqh) Scenario Diagram(s) 12

[4b](#_z337ya) Product Scenario List 12

[4c](#_3j2qqm3) Individual Product Scenarios 13

[5](#_1y810tw) Stakeholders 13

[5a](#_4i7ojhp) The Client 13

[5b](#_2xcytpi) The Customer 13

[5c](#_1ci93xb) Hands-On Users of the Product 13

[5d](#_3whwml4) Maintenance Users and Service Technicians 14

[5e](#_2bn6wsx) Other Stakeholders 14

[5f](#_qsh70q) User Participation 14

[5g](#_3as4poj) Priorities Assigned to Users 14

[6](#_1pxezwc) Mandated Constraints 15

[6a](#_49x2ik5) Solution Constraints 15

[6b](#_2p2csry) Implementation Environment of the Current System 15

[6c](#_147n2zr) Partner or Collaborative Applications 16

[6d](#_23ckvvd) Off-the-Shelf Software 16

[6e](#_ihv636) Anticipated Workplace Environment 16

[6f](#_32hioqz) Schedule Constraints 16

[6g](#_1hmsyys) Budget Constraints 16

[7](#_41mghml) Naming Conventions and Definitions 16

[7a](#_2grqrue) Definitions of Key Terms 16

[7b](#_vx1227) UML and Other Notation Used in This Document 17

[7c](#_3fwokq0) Data Dictionary for Any Included Models 18

[8](#_1v1yuxt) Relevant Facts and Assumptions 18

[8a](#_4f1mdlm) Facts 18

[8b](#_2u6wntf) Assumptions 18

[II](#_19c6y18) Requirements 19

1Product Use Cases 19

1[a](#_28h4qwu) Use Case Diagrams 19

1[b](#_1mrcu09) Product Use Case List 19

1[c](#_46r0co2) Individual Product Use Cases 20

2Functional Requirements 21

3 [Data Requirements](#_111kx3o) 22

4Performance Requirements 22

4[a](#_206ipza) Speed and Latency Requirements 22

4[b](#_4k668n3) Precision or Accuracy Requirements 22

4[c](#_2zbgiuw) Capacity Requirements 23

5Dependability Requirements 23

5[a](#_3ygebqi) Reliability Requirements 23

5[b](#_2dlolyb) Availability Requirements 23

5[c](#_sqyw64) Robustness or Fault-Tolerance Requirements 24

5[d](#_3cqmetx) Safety-Critical Requirements 24

6Maintainability and Supportability Requirements 24

6[a](#_4bvk7pj) Maintenance Requirements 24

6[b](#_2r0uhxc) Supportability Requirements 25

6[c](#_1664s55) Adaptability Requirements 51

6[d](#_3q5sasy) Scalability or Extensibility Requirements 25

6[e](#_25b2l0r) Longevity Requirements 26

7Security Requirements 26

7[a](#_34g0dwd) Access Requirements 27

7[b](#_1jlao46) Integrity Requirements 27

7[c](#_43ky6rz) Privacy Requirements 27

7[d](#_2iq8gzs) Audit Requirements 27

7[e](#_xvir7l) Immunity Requirements 28

8Usability and Humanity Requirements 28

8[a](#_1x0gk37) Ease of Use Requirements 28

8[b](#_4h042r0) Personalization and Internationalization Requirements 28

8[c](#_2w5ecyt) Learning Requirements 28

8[d](#_1baon6m) Understandability and Politeness Requirements 29

8[e](#_3vac5uf) Accessibility Requirements 29

8[f](#_2afmg28) User Documentation Requirements 29

8[g](#_pkwqa1) Training Requirements 29

9Look and Feel Requirements 29

9[a](#_1opuj5n) Appearance Requirements 29

9[b](#_48pi1tg) Style Requirements 29

[10](#_2nusc19) Operational and Environmental Requirements 30

[10a](#_1302m92) Expected Physical Environment 30

[10b](#_3mzq4wv) Requirements for Interfacing with Adjacent Systems 30

[10c](#_2250f4o) Productization Requirements 30

[10d](#_haapch) Release Requirements 30

11Cultural and Political Requirements 30

[11a](#_1gf8i83) Cultural Requirements 30

[11b](#_40ew0vw) Political Requirements 30

12Legal Requirements 31

12[a](#_upglbi) Compliance Requirements 31

12[b](#_3ep43zb) Standards Requirements 31

13Requirements Acceptance Tests 31

13[a](#_2szc72q) Requirements – Test Correspondence Summary 31

13[b](#_3s49zyc) Acceptance Test Descriptions 31

[III](#_279ka65) Design 32

[22](#_meukdy) Design Goals 32

[23](#_36ei31r) Current System Design 33

[24](#_1ljsd9k) Proposed System Design 33

[24a](#_45jfvxd) Initial System Analysis and Class Identification 33

[24b](#_2koq656) Dynamic Modelling of Use-Cases 34

[24c](#_3jtnz0s) Proposed System Architecture 34

[24d](#_1yyy98l) Initial Subsystem Decomposition 35

[25](#_4iylrwe) Additional Design Considerations 36

[25a](#_2y3w247) Hardware / Software Mapping 36

[25b](#_1d96cc0) Persistent Data Management 36

[25c](#_3x8tuzt) Access Control and Security 36

[25d](#_2ce457m) Global Software Control 37

[25e](#_rjefff) Boundary Conditions 37

[25f](#_3bj1y38) User Interface 37

[25g](#_1qoc8b1) Application of Design Patterns 37

[26](#_4anzqyu) Final System Design 38

[27](#_2pta16n) Object Design 38

[27a](#_14ykbeg) Packages 38

[27b](#_3oy7u29) Subsystem I 38

[27c](#_243i4a2) Subsystem II 39

[27d](#_j8sehv) etc. 39

[IV](#_338fx5o) Project Issues 39

[28](#_1idq7dh) Open Issues 39

[29](#_42ddq1a) Off-the-Shelf Solutions 39

[29a](#_2hio093) Ready-Made Products 39

[29b](#_wnyagw) Reusable Components 39

[29c](#_3gnlt4p) Products That Can Be Copied 39

[30](#_1vsw3ci) New Problems 40

[30a](#_4fsjm0b) Effects on the Current Environment 40

[30b](#_2uxtw84) Effects on the Installed Systems 40

[30c](#_1a346fx) Potential User Problems 40

[30d](#_3u2rp3q) Limitations in the Anticipated Implementation Environment That May Inhibit the New Product 40

[30e](#_2981zbj) Follow-Up Problems 41

[31](#_odc9jc) Migration to the New Product 41

[31a](#_38czs75) Requirements for Migration to the New Product 41

[31b](#_1nia2ey) Data That Has to Be Modified or Translated for the New System 41

[32](#_47hxl2r) Risks 41

[33](#_2mn7vak) Costs 41

[34](#_11si5id) Waiting Room 42

[35](#_3ls5o66) Ideas for Solutions 42

[36](#_20xfydz) Project Retrospective 42

[V](#_302dr9l) Glossary 42

[VI](#_1f7o1he) References / Bibliography 42

[VII](#_3z7bk57) Index 42

### List of Figures

[Figure 2 - Sample Use Case Diagram from Bruegge & DuToit ( modified ) 37](#_nmf14n)

[Figure 3 - Sample Use Case Diagram from Robertson and Robertson 37](#_37m2jsg)

### List of Tables

[*Table 2 - Requirements - Acceptance Tests Correspondence* 75](#_184mhaj)

# Project Description

## Project Overview

The product to be developed is a map-based game using real world data. The player would be shown a demo which will demonstrate a pattern on the map with a starting point and an ending point. The player would then have to replicate the exact same pattern in order to advance up in the game. The patterns would-be real-world locations integrated into the game via google maps API.

## The Purpose of the Project

### The User Business or Background of the Project Effort

Map-based games are becoming increasingly popular as the location and navigation-based technologies are advancing with the growth in the information and technology sector. Map-based games create an entirely unique medium of interaction between the player and the game that no other game can replicate. However, map-based games have mostly been developed for recreational purposes and there has been no significant development as to how we can make map-related games for educational purposes. This game will focus on exploring the educational aspect of the map-based games and how we can use it to make the quality of life better for the people who utilize this tool.

### Goals of the Project

The primary focus of this project is to help young generation understand the value of practical knowledge who have become increasingly dependent on technology to such an extent that they cannot navigate from one place to another without using maps. This game will help players strengthen their muscle memory and will also improve their information retention span, as it will require players to mimic unique patterns for which the player will have to pay close attention to the game. The game will have multiple domains e.g. a player who wants to get familiar with his neighborhood will be given challenges and patterns that will help him become more knowledgeable about his surroundings and it can be extremely helpful in the case of an emergency.

### Measurement

Different statistical tools can be used to measure the progress in the different areas that the game focuses on addressing as highlighted in the above paragraphs. The educational value can be measured by looking at the number of people who have shown interest in playing the game, and the players feedback will be an important resource to analyze whether the game is actually helping the target audience or not.

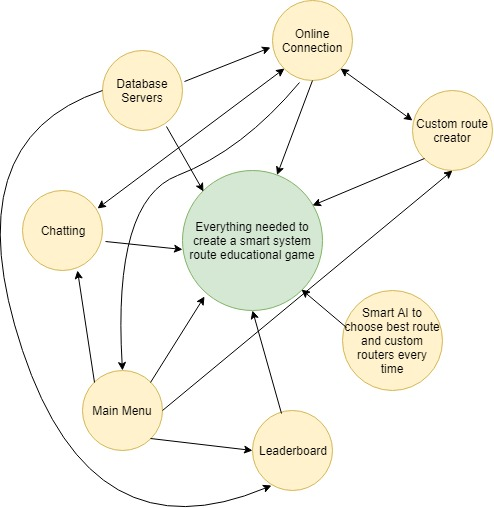
## The Scope of the Work

The product is a location based game with the intention of improving one's memory and problem solving skills, furthermore it should be able to implement VR such as oculus rift.

**The Current Situation:**

The possibilities of patterns are infinite. But if a player were to play in the same locality the patterns may become dull therefore a constant player must set out to further more different locations to get more challenging and thought provoking patterns. If groups were to play in the same locality there shouldn’t be any issues as they would just have similar patterns and may run into each other at worst. But if the user would want to play somewhere more desolate then that option will also be available. The game will also be storing what patterns were done so if a user wants to do an old pattern then that option is also available.

### The Context of the Work



### Work Partitioning

**Business Event List**

**Event Name Input and Output Summary**

|  |  |  |
| --- | --- | --- |
| Create an Account | Input this information into a database | A user can customize a player to whatever name that user would prefer |
| Enter a waiting stage | the account created in previous step will be stored in the database and entered into a waiting stage | In this waiting stage players will be able to see where other players are playing and could decide where they’d like to start |
| Start a game | A pattern will be generated based on the location choice of the player | Once the game has begun the information of the player will be sent to a game engine which will then decide on what other patterns to create, depending on the skill level of the player, in order to really challenge the players. |
|  |  |  |
| Select a different location | Input: From player  Output: the players next pattern will be in a different location | This allows the player to learn more directions and slowly but surely familiarize a player to many different streets and eventually even cities |
| View all the locations a player has been | Input: player object or database | This will allow a player to check which cities or streets they have done already and which ones still need to be |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*. .*

### Competing Products

.

In general there is absolutely no competition with this product, the only thing that can even be considered competition are apps that have mind challenging games. To say it simply the competition is playing checkers and we've moved on to chess. But this product that we plan to develop is real world based and requires you to actually engage with your location and surroundings.

## The Scope of the Product

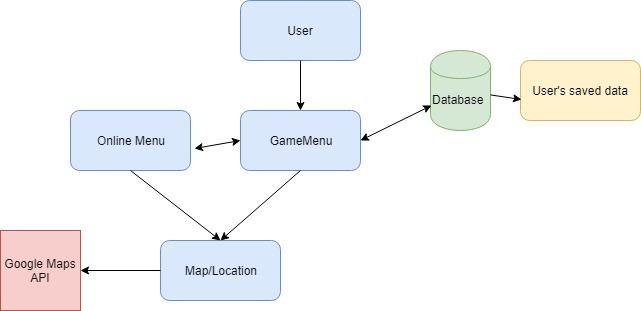
The purpose for this product is to create an environment where a player must remember not only the surroundings but also locations, and patterns to ultimately get users to not be so dependent on GPS to go from one place to another. Players should generally be 13 years of age and up or they should be accompanied by a guardian as we are not liable for kidnappings that may occur.

This Software will allow users to performore the following:

Single player - In this mode players will be able to go on solo walks or runs through cities of their choice.

Multiplayer - In this mode players will be able to go on walks or runs through cities of their choice only this time their paths will diverge at some point and converge at the end

### Scenario Diagram(s)



### Product Scenario List

1. Single player mode
2. Multiplayer mode

### Individual Product Scenarios

Single player mode - the player will choose single player from the main menu and then the player will be prompt with where they’d like to play.

When the game finishes all the places visited will be stored in a leaderboard as well as inside the player object.

Multiplayer mode - the player will choose multiplayer from the main menu and then the player will be prompt with where they’d like to play.

Now the player will be given a location where other players are playing and the user will be able to message them thru this game.

When the game finishes all the places visited will be stored in a leaderboard as well as inside the player object.

## Stakeholders

### The Client

Client for this project is the Chicago’s tour guide company that wants to educate it’s employees in order to improve the quality of services provided to their own clients. By using our map-based game tour companies can improve their employees directions giving and knowledge about the Chicago’s grid system. Also, school can utilize our map-based game to improve students memory performance.

### The Customer

Game customers are employees of the tour guide companies, and students enrolled in classes that the have instructors using the map-based game. Any user that can play

### Hands-On Users of the Product

User category: Age 6+

User role: User roles is to play the map based game and memorize direction. Multiplayer mode might be available in later versions.

Subject matter experience: User should be able to have a knowledge of using the computer.

Technological experience: User should be able to have a knowledge of using the computer.

Other user characteristics: User should be able to improve memory and knowledge of city and become better in navigating through the city and learning the Chicago grid system.

### Maintenance Users and Service Technicians

Maintenance users should be experienced working with databases, Google API’s and have some Java experience.

### Other Stakeholders

Transportation companies that require their employees to have a better knowledge of city and become better in giving directions. Same goes for the hospitality companies for example doormans and concierges that are giving tourists directions. Also, all other users/people that want to get better at knowing the city and get better at memorizing.

### User Participation

User needs to have a computer and knowledge on how to use it. This app will become available on mobile devices which can get us a greater audience/user list.

### Priorities Assigned to Users

Key users: Age 6+

Secondary users: People that want to get better at knowing the city and get better at memorizing.

## Mandated Constraints

### Solution Constraints

Description: Client shall be able to see the progress in improving the memory skills after moving up to a different game level.

Rationale: The product will have to track the users performance and use a leadership board to compare its results to other users.

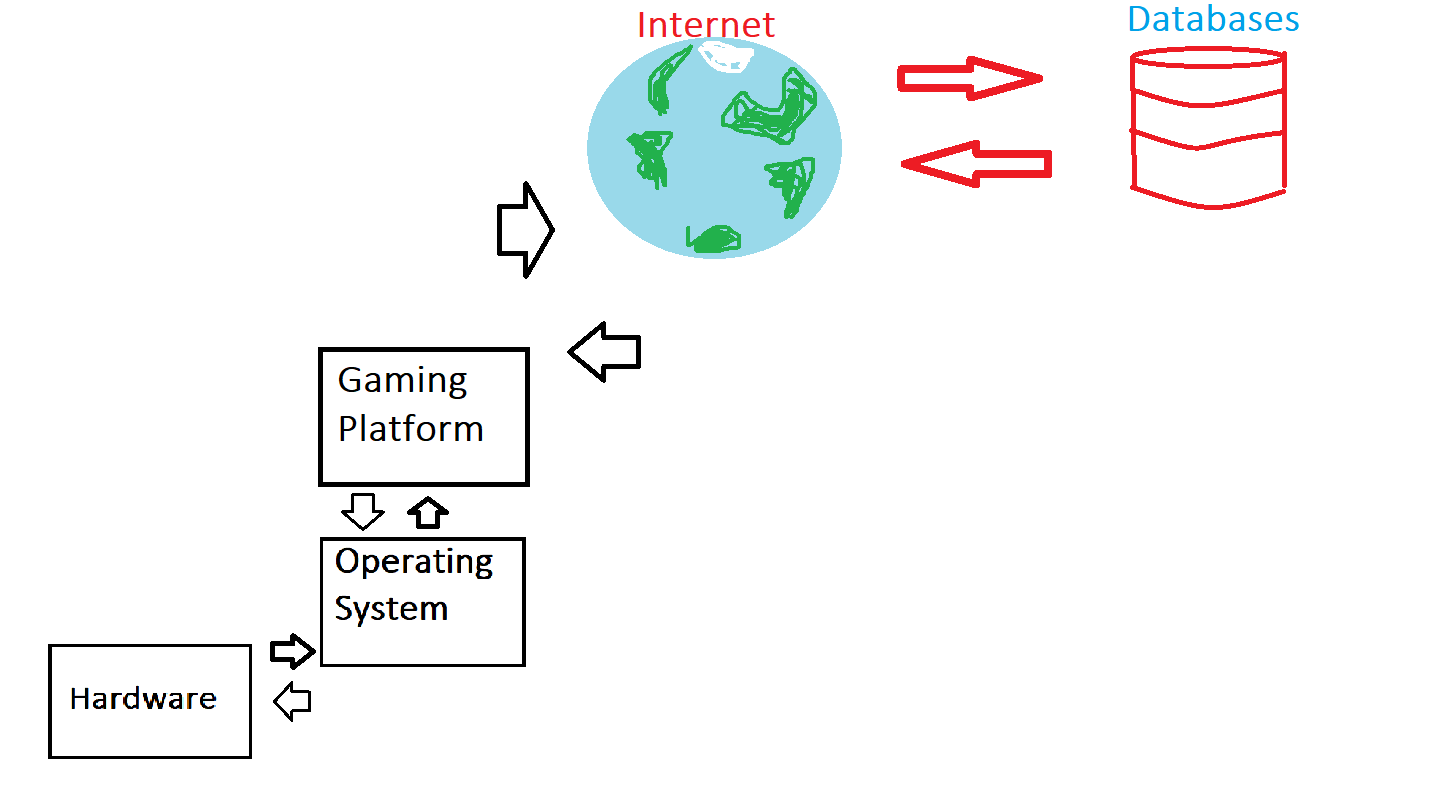
Fit criterion: Dashboard will be created with data that presents the performance over the period of time.

Description: Client shall be able to select between single or multiplayer mode.

Rationale: Client can choose between two different modes.

Fit criterion: Client will have an option in the user interface to select the appropriate mode.

### Implementation Environment of the Current System

**

### Partner or Collaborative Applications

Servers: This product will require servers that can allow multiplayer mode and downloading the Google maps API data.

Gaming Platform: PC or smartphone (in the future).

Operating system: Windows, iOS, Android (coming soon)

Software: The product will require a software solution that will allow users to install/play the game.

### Off-the-Shelf Software

Content: This product does not require OFT (off-the shelf) software.

Motivation: Both commercial and free product use. Commercial use allows for extended explanations and more detailed maps, while free product use will have a limited map and multiplayer 30 days free trial.

### Anticipated Workplace Environment

Anticipated Workplace Environment for this platform requires an internet connection and a System Running Windows 7+ and Mac OS 10.5+.

### Schedule Constraints

In the processing of creating are no deadline limitations at this point.

### Budget Constraints

There are no budget constraints at this point since student from CS 440 will work on developing this app in the future semesters at no cost.

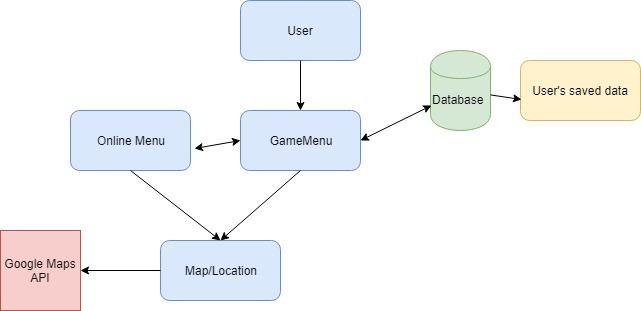
### 7 Definitions of Key Terms

**7a** [**Definitions of Key Terms**](#_2grqrue)

Google Maps API: Used to get the real world locations and so the game can choose a route to follow.

UML: Unified Modeling Language. It is a way to construct the artifact of the software system.

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



### 

### 7c Data Dictionary for Any Included Models

Database: Contains the players information(name, score, rank).

Google API: Used to store players location and to keep getting updated location.

## 8 Relevant Facts and Assumptions

### 8a Facts

The game uses google maps API.

The player must follow the same route the system has taken to get to the destination.

If player doesn’t follow same route, but ends up at the same destination, then the player gets a certain amount of points.

The game is intended to make the player learn the streets of Chicago.

### 8b Assumptions

User may be able to compete against a friend/random player to see who gets to the destination faster.

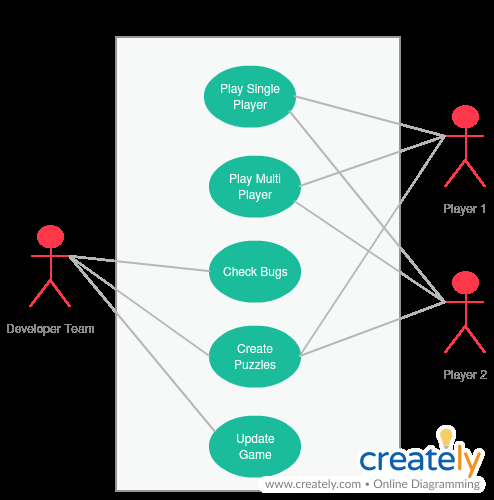
Once a certain amount of points have been earned, the player may progress to harder routes and get a new reward.

Player can down the game offline to be played without internet. If it is played offline, the system downloads a current map during the downloading phase. This map will not change unless a new map is updated.

# Requirements

## Product Use Cases

### Use Case Diagrams



**Figure 1 - Example of Main Menu Use Cases**

### Product Use Case List

1. Start single player (online)

2. Start single player (Local)

3. Start Menu

4. Exit game

### Individual Product Use Cases

|  |  |
| --- | --- |
| Use case name | Play single player |
| Actors | User |
| Sequence of events | 1. User will select online mode  2. User will be taken to an online game waiting lobby  3. Once User is ready to play the user will select start game  4. User will be in the game and will have the options to continue playing or save and quit |

|  |  |
| --- | --- |
| Use case name | Play single player (Local) |
| Actors | User |
| Sequence of events | 1. User will select Local mode  2. User will be taken to a local game waiting lobby  3. Once User is ready to play the user will select start game  4. User will be in the game and will have the options to continue playing or save and quit |

|  |  |
| --- | --- |
| Use case name | Main menu |
| Actors | User |
| Sequence of events | 1. User will be at main menu once the user launches the program  2. User will go to menu where the user will select whether to play local, online or to check the users stored data.  3. User will also be be able go to a skin store to purchase in game enhancements |

|  |  |
| --- | --- |
| Use case name | Exit |
| Actors | User |
| Sequence of events | 1. User selects exit from game  2. The program will automatically save everything  3. User will return to main menu where the user can Exit from program. |

## Functional Requirements

**Description:** The system should allow the user to create an account if they are not logged in. If they login, and have an account they can go straight to logging in.

**Rationale:** This allows new users to create an account easily without any problems. Also if the player already has an account they can login quickly.

**Fit Criterion:** This option will be available when the player first launches the game. After that the game will know if the user has created an account.

## Data Requirements

**Description:** There will be different types of databases that store different types of information. One database will store the players account information, and everything the character owns/has. The second database will store all the routes the system has in place. Also this database stores all the custom routes the players have created.

**Rationale:** The reason for multiple databases is so that if something goes wrong in 1 database, the others are safe. It keeps all the data separated incase of bugs or roll backs.

**Fit Criterion:** This requirement is met when all the databases are setup and running.

## Performance Requirements

### Speed and Latency Requirements

**Description:** When starting the game, it shall open in less than 20 seconds. The data will be loaded while the player is in the main menu. In case the player enters a game, there will be a short loading screen if the loading is incomplete.

**Rationale:** Players do not like to wait a long time for a game to load. Having a fast loading time will allow the players to quickly navigate the game.

**Fit Criterion:** The requirement is met when in every case the game is loaded, it loads in under 20 seconds.

### Precision or Accuracy Requirements

**Description:** The servers should be ready to be used and be ready to host all the users at least 97% or more of the time. Also, the game physics should be very precise on keeping the game loaded and having accurate routes to different locations.

**Rationale:** If the game is unable to host players then this is not good for the game. Players will quit. It is important for the game to be accurate.

**Fit Criterion:** The requirement is met when the game is able to host all the users at least 97% of the time or more.

### Capacity Requirements

**Description:** At the start, the servers should be able to host up to 1000 players. This will be the testing phase for a small group of schools. After this, it should be able to host over 50,000 players at a time. There should be no limit to database storage.

**Rationale:** The starting phase is a stress test. If servers are running smooth at 1000 players, then it is ready to release globally. If it is not meeting running requirements, then an upgrade to the servers is mandatory.

**Fit Criterion:** The requirement is met when the servers are running with over 50,000 players with no problems.

## Dependability Requirements

### Reliability Requirements

**Description:** The product should not crash if it encounters a problem that it cannot solve. Either if it is inside the actual game, or in main menu. If it does, system needs to save before exiting.

**Rationale:** This requirement is needed because if the game crashes while the player is playing the game, we don’t want the player to lose their data.

**Fit Criterion:** The requirement will be met if in the rare occasion the game crashes. The system saves everything the player has done until the point of crash. If player relogs, the data was saved.

**Acceptance Tests:** A test would be to play the game, and forcefully crash the game. If data was saved, test was successful.

### Availability Requirements

**Description:** The servers should be up most of the time. At least 97% or greater.

**Rationale:** Reason for some downtime is for maintenance and updates.

**Fit Criterion:** We know it will meet this requirement if we keep the server up 97% of the time and we plan the maintenance accordingly.

**Acceptance Tests:** Make sure servers are up 97%.

**Description:** Players should have access to the game 24/7, unless the game is in maintenance or update.

**Rationale:** Requirement is needed because if a game is unavailable, players will not play. People should be able to play anytime they feel like it.

**Fit Criterion:** Meets the requirement if we see the game is on most of the time.

**Acceptance Tests:** Make sure game is on most of the time.

### Robustness or Fault-Tolerance Requirements

**Description:** If the player who is playing the game loses connection to internet. It will disconnect player if they are playing online and place them in single player.

**Rationale:** Players should still be able to play the game if connection is lost. When online, data is saved and placed in single player. So they can continue to play.

**Fit Criterion:** Player is online and is disconnected, if player is placed in single player and data is saved for online, then we have met the requirement.

### Safety-Critical Requirements

**Description:** No data should be modified that should not be modified. Do not modify, delete, or corrupt any data other than products allocated data storage.

**Rationale:** Don’t want anyone or anything modifying data that should not be modified. If this data is modified, deleted, or corrupted this is not good for the system. We will lose data and will need to roll back our data in order to recover.

**Fit Criterion:** Make sure nothing is modified, deleted, or corrupted.

## Maintainability and Supportability Requirements

### Maintenance Requirements

**Description:** The game will require weekly minor maintenance to make sure everything is running properly. Also every few weeks there will be an update to the game.

**Rationale:** Every week the developers will be required to make sure everything is running properly and smooth. Reason for new updates is to keep the fresh and give the players more things to do.

**Fit Criterion:** The requirement will be met when the product is released and weekly the game is receiving maintenances and updates.

### Supportability Requirements

**Description:** The game will have an option at the main menu that takes the player to a new window that tells the player everything about the game. (A manual)

**Rationale:** Every new player needs to know how the game works and if the player has a question they can look it up in the manual.

**Fit Criterion:** The support requirement will be met when the manual is created in the main menu of the game.

### Adaptability Requirements

**Description:** The product is expected to run on all computer systems (Windows, Mac, Linux).

**Rationale:** The game should not restrict any player from playing if they have a different machine. Therefore the game should be compatible to be run on any machine.

**Fit Criterion:** Requirement shall be met when the system is able to be ran on all systems.

**Description:** The product will be placed on steam so any player can download and play it.

**Rationale:** Steam is a great place for players to download games. It is easy to download and understand how it works.

**Fit Criterion:** Requirement is met when the game is placed on steam.

### Scalability or Extensibility Requirements

**Description:** At launch day of game, the expected number of users is around 1000. Within the years the game is up and being updated, the expected number of users will increase to 10000.

**Rationale:** Reason for only 1000 players is because there will only be a handful of schools that will get hands on. These students will test and see if they like it. Once this is completed, the game will be released to public in a few weeks. This is when the player base should increase.

**Fit Criterion:** Requirement shall be met when the amount of players reaches the requirement.

### Longevity Requirements

**Description:** Company will hire professionals who will deal with players inside the game. They will communicate with developers about what the players ask. They will also take suggestions, any bugs found, and create events for players.

**Rationale:** The game needs to have an active community with professional staff assisting players. Players will need someone to talk to if they encounter bugs/problems and suggest any suggestions. The events for players are created so the game feels unique and for players to get extra rewards.

**Fit Criterion:** Requirement is met when community professionals are hired and assisting players in their day to day playtime. Once events are successfully being started.

## Security Requirements

### Access Requirements

**Description:** Only the user has access to their personal details and items.

**Rationale:** Don’t want any random players accessing any other players details, other than developers. This will create problems of players losing items/data.

**Fit Criterion:** Requirement is met when system safety requirements are added into the game. Allow no possible way for player to access data of other players.

**Description:** The developers will also have access to add players data to ensure everything is correct.

**Rationale:** If the player has a problem, the developers will be able to access the data and fix the problem.

**Fit Criterion:** Requirement is met when only the developers have access to data.

### Integrity Requirements

**Description:** The product shall protect itself from hackers accessing private information and manipulating the system.

**Rationale:** Can not allow hackers from accessing private information that will lead to breaking the game. If hackers access/change data this will affect every player.

**Fit Criterion:** Requirement will be met if updates to security are consistent and blocking hackers from accessing anything they should not be.

### Privacy Requirements

**Description:** The product will notify players if there are any changes to the policy.

**Rationale:** If the game changes its policy, the players need to be aware of what has changed.

**Fit Criterion:** Requirement will be met when the player is notified if any policy has changed.

**Description:** The product shall only reveal private information in accordance with the privacy laws and and the games information policy.

**Rationale:** Don’t want to break any private policy laws. Therefore the system will only reveal private information according to these laws.

**Fit Criterion:** Requirement is met when none of the private information is released.

### Audit Requirements

**Description:** The product will save all chat discussions between players, save server logs, and save all errors that players encounter.

**Rationale:** If there is a problem that is encountered between players or between the player and the system, the developers will be able to look back at the logs and solve the problem.

**Fit Criterion:** The requirement shall be met when the system will automatically save chat logs, server logs, and errors.

### Immunity Requirements

**Description:** The product will check the system data once player logs on.

**Rationale:** This is to prevent any corrupt data when player logs on. This will allow the system to catch the problem before player has logged on.

**Fit Criterion:** The requirement shall be met when data is successfully checked before player logs on.

## Usability and Humanity Requirements

### Ease of Use Requirements

This game has a user interface that is made to be easy to use, easy to learn and easy to navigate through. This game has a start screen easy to navigate and easy to learn how to operate.

The gameplay and options in the game are easy to learn and they are self-explanatory which makes the game suited for all ages.

Interviews with users showed that most of users about eighty seven percent were able to navigate easy through the game interface while other used said that the game instructions weren’t clear.

### Personalization and Internationalization Requirements

Game shall have a language selection and the game will support English and

Spanish since those are the two most used languages in North America.

### Learning Requirements

Instructions that are used in the game are made clearly and there are no specific learning requirements. User should be able to read in English and Spanish and users should be able to memorize the streets names.

### Understandability and Politeness Requirements Accessibility Requirements

This game will be accessible for color blind people. The game will not display the facts on how the program and its methods are implemented. Language, images and sounds will be all ages appropriate and they will be easy to understand and not be specific to certain cultures.

### User Documentation Requirements

### This game will have an option that explains users how the game is played, there will be also a short video in addition to written manual.

### Training Requirements

This game will have a video that explains users how to play in addition to interactive tutorial and written instructions.

## Look and Feel Requirements

### Appearance Requirements

This gave will be appropriate for all users older than 5 years.

.

### Style Requirements

This game will include day and night time lighting option in addition to different color themes that will change the color of buttons.

## Operational and Environmental Requirements

### Expected Physical Environment

The user/player can play this game anywhere considering the fact that they have access to internet and their system meets the minimum requirements. Although the user can technically play this game in any setting of his choice but it is better to be in a quiet and peaceful space as the game requires sheer focus and memorization.

### Requirements for Interfacing with Adjacent Systems

In near future the game might also be released on other gaming platforms like PlayStations, Xbox, Nintendo Switch. So, all those kids who don’t have a personal mobile phone can experience the game on standardized gaming consoles.

### Productization Requirements

The game should not take more than 1gb of space with all the features enabled and working on the mobile phone. The user will be required to download the game from the compatible app store.

### Release Requirements

Every new update/version upgrade will ensure data security and smooth transition to the upgraded version. Any bugs in the upgrade will be fixed as soon as possible after the initial response of the players about the new upgrade gets received by the company. Every new update will come with a feedback form that the player can fill out to convey his concerns regarding the new version and what improvements the game can make in the future upgrades.

## Cultural and Political Requirements

### Cultural Requirements

The game will not be offensive to any religious or ethnic group and will respect all the societal and cultural norms. The game will present all the information accurately as per the required standards and there will be no unnecessary addition. The game will support multiple languages e.g. English, Japanese, French, Spanish, Germal, Arabic, Hindi etc.

### Political Requirements

The game will be neutral from the perspective of all the political entities within the organization and those that are external to the company. The game will have a layered security framework that will only provide limited access to the people who don’t have the authority to handle sensitive information.

## Legal Requirements

### Compliance Requirements

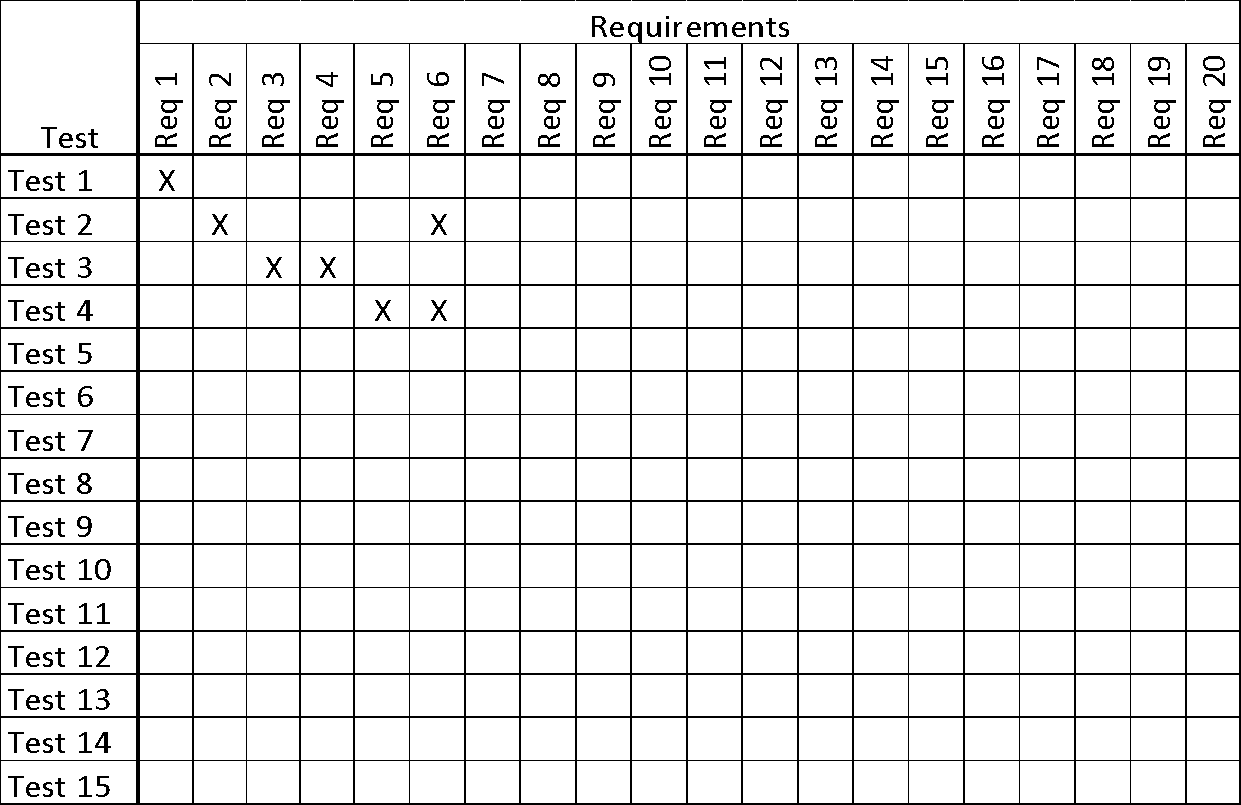
All personal information that the user provide will be protected under the data protection laws. Proper security infrastructure will be installed to protect any data theft of the sensitive information.

### Standards Requirements

The game will comply with ESRB ( Entertainment Software Rating Board) Everyone rating standard.

## Requirements Acceptance Tests

### Requirements – Test Correspondence Summary

**

***Table 1 - Requirements - Acceptance Tests Correspondence***

### Acceptance Test Descriptions

# Design

## Design Goals

There were a couple aspects of the design for our game that we believed the user needed. Some of these design goals are listed below:

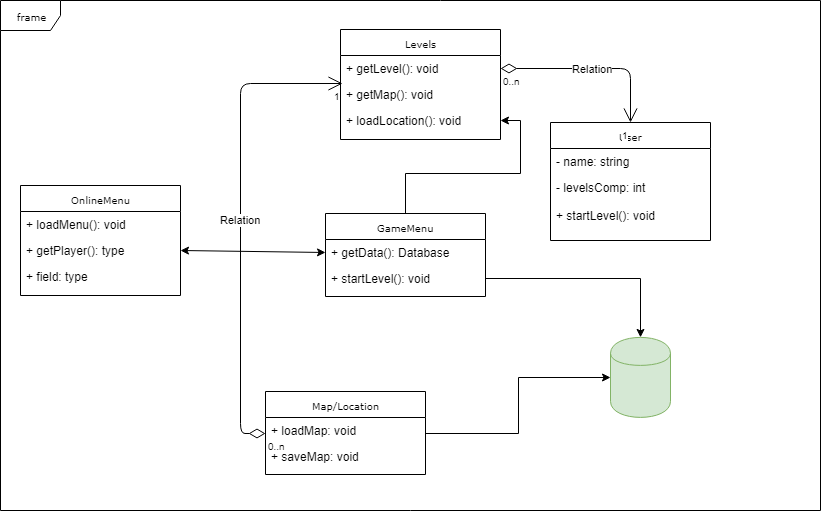
* The system should be flawless, the game shouldn’t crash without a reason. In the case that the system does crash, then no data should be lost. Instead it should be recoverable from any state. If some unexpected hardware issues occur during the players multiplayer or singleplayer session, then everything they have done should be saved and the appropriate rewards are given out. When any user crashes, then all the data is sent as a report to the developers for further analysis.
* Every level created automatically or manually through some program has to be completed first by the developer or player who created the level. This will stop impossible levels from being created and people being stuck on a stage.

## Current System Design

The system is to be designed and playable on windows, mac, and mobile.

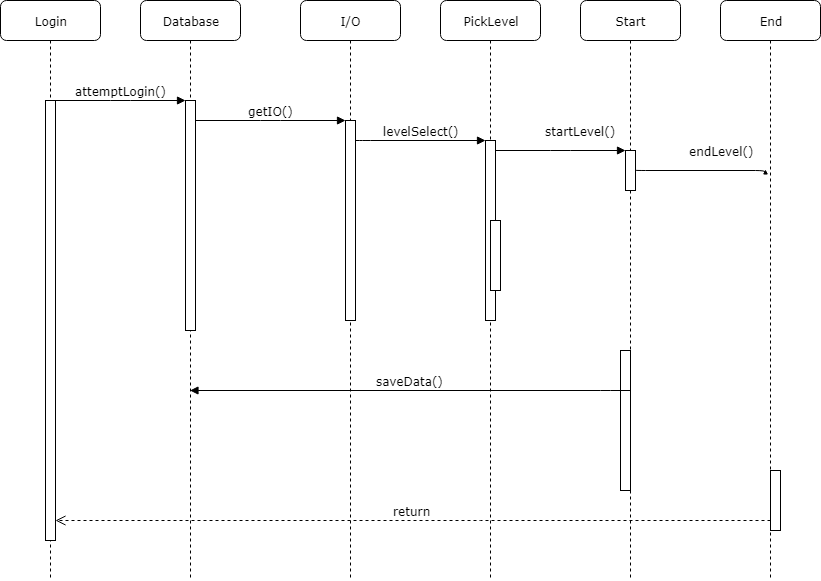
## Proposed System Design

### Initial System Analysis and Class Identification



This simple class diagram just shows the basic model of the game. We have an online menu and regular menu. Each Menu access specific levels and gets the map/location from the database.

### Dynamic Modelling of Use-Cases

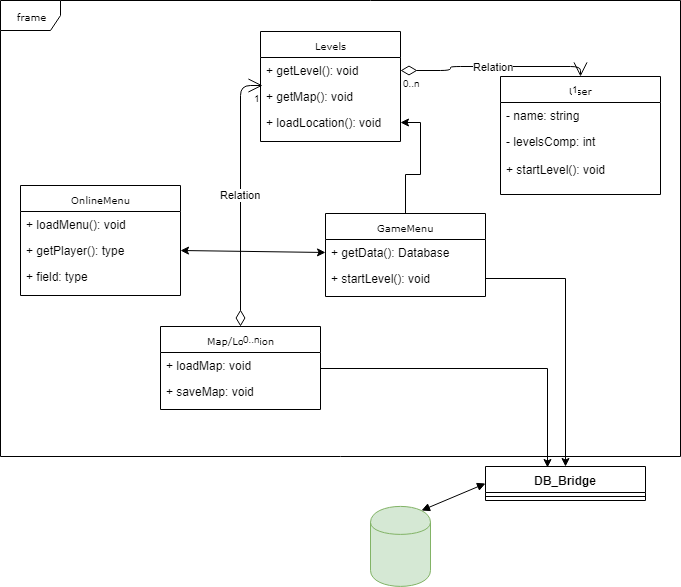


This dynamic diagram shows the prototype life cycle of the game. It is subject to change.

### Proposed System Architecture

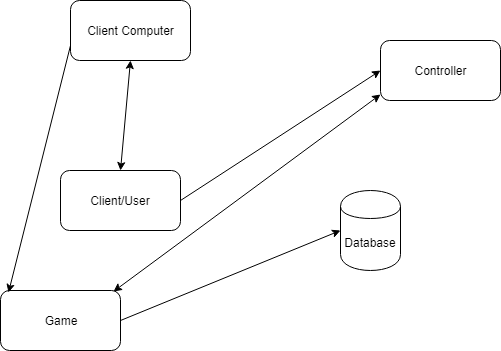
The software architecture to be applied to this project is client-server communication. With database access. This allows the server to create unique levels for each player (client).

### Initial Subsystem Decomposition



## Additional Design Considerations

### Hardware / Software Mapping

**

The client is connected with his computer that then is connected to the game. The game listens to the controller that is used by the client/user. The game is only able to access the database so no one else will be able to access it.

### Persistent Data Management

All of the data that the game has will be stored in a SQL database. There will be different tables to count for different types of data. There will be a user/player table, to store the players information and everything they obtained. There will also be a table that stores the maps and locations. Every pattern that is created.

### Access Control and Security

Access to the database should only be accessible by developers and selected people. All password information will be encrypted directly to the SQL. All of the data should be stored for a few weeks incase of a rollback or system malfunction. *.*

### Global Software Control

This game is designed for players to learn directions and where to go. Some complex levels will be designed to be long and there will be tons of different directions to go. So the system needs to figure out from where the player is currently, to what are the other possible directions.

### Boundary Conditions

If the game is experiencing glitches or buffs, or if the servers go into maintenance or update. Then the team responsible must inform the user ahead of time about the start and end time for the downtime. Before the shutdown, all data should be stored into the database as to not lose any data. This is also covered if the system shuts down unexpectedly.

### User Interface

The UI for the game will be simple. There will be a single screen and the player can choose either to switch to multiplayer screen or single player. Below is a detailed list:

Multiplayer screen

* Options: Show the different options the player has.
* HowToPlay: Brief description of how to play the game
* Help: Takes the player to a help page
* Switch Single player: Switches to single player

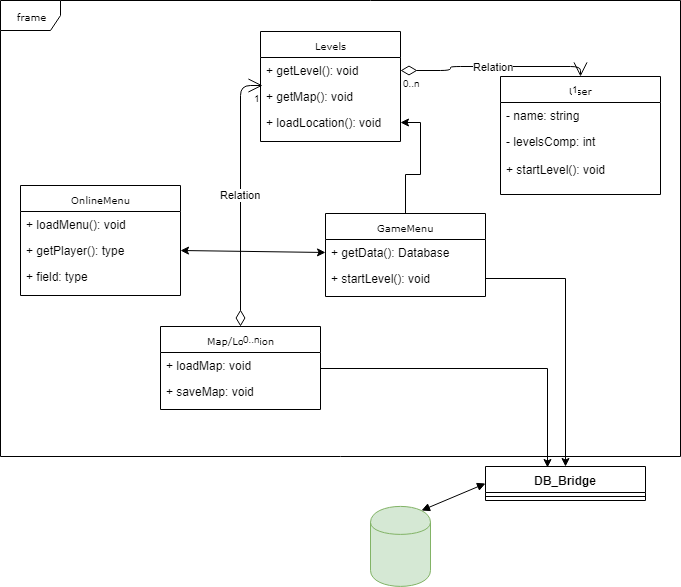
Single player screen

* Same options as multiplayer but also the ability to switch to multiplayer.

### Application of Design Patterns

N/A

## Final System Design



## Object Design

### Packages

Each package should have documentation on which classes and subclasses belong to it. Also the use of each class.

### Subsystem I

There will be a subsystem 1 which will be used for levels created manually, aka the level creator subsystem.

### Subsystem II

The 2nd subsystem is designed for every level designed in the game. It will have all of the levels associated with it.

### etc.

N/A

# Project Issues

## Open Issues

Presently the game is integrated with google maps API and its relying on the data provided by the api. However, in the near future if google planned on commercializing the API for all the developers than the cost of developing the product would significantly increase.

The other concern is how to reduce the server uptime during software updates as any major update to the server will requires us to shutdown the server for an extended period of time.

## Off-the-Shelf Solutions

### Ready-Made Products

In case of API restrictions we can look for other map development companies that can meet our demands or we can integrate other developer friendly API’s to counter this issue.

In order to tackle the server updates we will use MySQL database to manage all our data and ensure no significant error effects the overall functioning of the application during the update. Furthermore, we are planning on using git for version control.

### Reusable Components

The overall design of the database will remain unchanged during the major updates. We can use many open source frameworks and libraries for the new release of the product.

### Products That Can Be Copied

Map-based games are becoming increasingly popular with the advances in the navigation and real time location technologies. We plan on incorporating creative ideas from games like pokemon Go and many more into our product to make it more fun and interesting.

As our game depends on unique patterns we can analyze different pattern dependent games in the market to understand how we can improve our pattern generation process.

## New Problems

### Effects on the Current Environment

Presently, there are no map-based games in the market that provide real time interaction between the maps and the user through which the user can learn alot about his neighborhood and surroundings. The introduction of our game will introduce a unique genre of apps that involves user interaction with the maps.

### Effects on the Installed Systems

Our game will rely heavily on the Java programming language for the core functionalities and therefore any significant updates to the Java inbuilt libraries can affect the overall functioning of our product. However, I think the change would not be too drastic and we will be able to handle any significant updates.

### Potential User Problems

Some of the potential user problems are being unable to load a map. In this case, the user will be unable to see the actual map only his location can be visible. Another potential problem can be being able to see the map, but unable to see the location. In this case, user will have reload the app, check for permission to see if they’re allowed. Next potential issue would be the network lag when user transforms for wifi to wifi, or from cellular connection to wifi connection.

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

The only limitation it poses is the speed, which will vary depending on whichever environments you use. This game will also rely heavily on google maps API, therefore issues may arise if google maps API crashes.

### Follow-Up Problems

Although our product is new in the market and one of its kind but still we will be in competition with other map based games as this new trend can have a significant impact on their revenue. Secondly, we also have to make sure that we carefully follow all the copyright laws to avoid any kind of legal trouble.

## Migration to the New Product*.*

### Requirements for Migration to the New Product

If the game is close to what we expect the playerbase to be then the servers will need to be updated to be able to support the increase in data created. Also addition databases will be needed and developers to work on bug fixes.

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

If there are a lot of users then we would need to redo our algorithms to increase the performance of the system and to reduce lag. If more players start to play, then it will start to lag.

## Risks

Some of the risks that we’ve considered as possible problems are as follows:

● Server costs may be much larger than expected

● Game might not be as popular as we expect

● Legal backlash from competitors

● scalability problems

● low replayability factor

## Costs

Estimated cost of the project will be around 1 million dollars and it will take up to 12 months to complete. In regards to resources this project will require 14 software developers, 2 project managers, 1 salesperson, CFO, and CEO. Lost opportunities might be losing market share due to Google’s new platform that’s similar in nature to our development project.

Approximate number of business events would be 4. Team plans on doing 4 major events and 14 smaller events these events will provide the place where the product can be showed off. These events would require funding as well, since there will be an vendors charge to have your products showcased in different gatherings and shows.

## Waiting Room

Some ideas that we had for the waiting room that weren’t really relevant to the cycle of the game is to have a shop where players can buy items to upgrade their character. You would earn coins from completing stages and then exchange these coins in the shop.

## Ideas for Solutions

To solve some of the problems, try to learn more about the la :Dnguage your coding in. See if you understand how graphical things work. Ask team mates questions if you are stuck. It is better to ask a question and solve a problem faster than get stumped and try to solve a problem for hours. Just remember you got this! There is always a solution!

## Project Retrospective

What worked well while we worked on this project was splitting up the jobs. If working in a group of 4, split up in teams of 2 and each take a different section of the game. Each work together and if help is needed ask the other team members.

The levels created for testing should be passable.

# Glossary

API - application programming interface

# References / Bibliography

|  |  |
| --- | --- |
| [1] | Robertson and Robertson, Mastering the Requirements Process. |
| [2] | A. Silberschatz, P. B. Galvin and G. Gagne, Operating System Concepts, Ninth ed., Wiley, 2013. |
| [3] | J. Bell, "Underwater Archaeological Survey Report Template: A Sample Document for Generating Consistent Professional Reports," Underwater Archaeological Society of Chicago, Chicago, 2012. |
| [4] | M. Fowler, UML Distilled, Third Edition, Boston: Pearson Education, 2004. |

# Index

Design 61, 63

Requirements 35, 51, 58

Test 64, 65