Software Engineering CS 440
Project Report

Beat the Profs
(Beat up em game)

GROUP 13

Hardik Prajapati
Junior Luna
Keith Thomas
Gowdhaman Sadhasivam

University of Illinois Chicago

Fall 2014

Table of Contents

List of Tables

List of Figures

I. Project Description

1. Project Overview

Beat the Prof is a side scrolling beat-'em-up game with comic art style. The game is a single player game where the user fights enemy from different environment to environment. At each environment the game starts with fighting bunch of graduate assistance, and research assistance, and at the end there will be a boss fight with the professor, each professor will have a special attribute like robot helper, assembly boost, and other features.

2. The Purpose of the Project

2a. The User Business or Background of the Project Effort

The story is that you're a UIC student who started freshman year as a Computer Science major. You eavesdrop on one of the professor's meeting and find out they are planning on creating a Cyborg to rule humankind. You tell your classmates but they won't believe you as you are new and have very little experience.

Our motivation for the game is CS department of UIC. We have spend a great deal of time, and learned so much in the couple of years. The motivation of the game is to stop the Professor from building a Cyborg that is intended to build to rule humans.

2b. Goals of the Project

Beat the Profs is a fast paced action game in old school style in new platform. The purpose of Beat the Profs is to have fun, and experience the UIC's CS department. The objective of the project is want to make a fun game that UIC CS student can relate as they play.

2c. Measurement

This game project want to make a fun game that all student of Computer Science will experience throughout their career at UIC. We will have a rating system for our game for people who download and play. The success of the game is measured by ratings given by students.

3. The Scope of the Work

The environment or clients we would be looking for is at Game Companies that can adapt our game into different platforms. Apps are the new fab and money makers so if our game can gain recognition in the mobile world, our app can success indefinitely.

3a. The Current Situation

The motivation of this game is for users (mostly college students) to enjoy some down time when needed from a long day of attending classes. We don't want to just target one demographic out there but we feel that with word of mouth, ads, and true enjoyment, we can lead a success app. We won't be changing the industry with this app but we will be taking advantage that mobile usage is at an all time high. Knowing what users want before they even know they want it is a huge payoff and the app world is so versatile with ideas that almost any applications can make it in just a blink of the eye.

3b. The Context of the Work

We need to understand the current status of the gaming industry to filter out the games that are more popular depending on metrics such as ratings and downloads. In addition to this, we need to find demographics while analysing gaming industry. There is need to identify strategies to make developers to build the game effectively. This can be done by looking into recent trends with developers, their level of comfort with certain programming languages and system platforms. Analysis about easy to use Application Programming Interfaces and Software Development Kits along with online support community needs to done.

3c. Work Partitioning

Game Event List

Event Name	Input and Output	Summary
Track User Growth	User Tracked(I) Low User(O)	Track number of users growth, if there is slow rate of growth , apply promotional strategies like social media and email
Update Software Development Kit	Update SDK(I)	Developers SDK should be updated regularly to provide more functionalities for developers
Monitor server health	Server Health Monitored(I) Server Unable to handle(O)	Servers are monitored and if there is need for more servers, install them
Track 3rd party developer interest	Developer Interest Tracked(I) Developer Interest Tracked(O)	The game product rely on 3rd party game developers. We have to consider them during certain events and give them incentives

Table 1 Game Event List

3d. Competing Products

Apps come and go. There is always the next big things when it comes to games. The lifespan of a game is very short because developers always come up with new ideas and new features. Users want new games to engage with because they become tired with the old ones therefore new games need to be developed for the consumers.

The motivation will consist of the following:

- Creating a new app for consumers to engage with,
- Creating more options for users to engage with,
- Implementing new features that consumes have not seen.

4. The Scope of the Product

Included in game's scope are variety of subsystem.

- Music
- Sound effects
- Main menu
- Settings
- Storyline
- Boss fights
- Comic style art
- Highscore
- Lives

- Health

4a. Scenario Diagram(s)

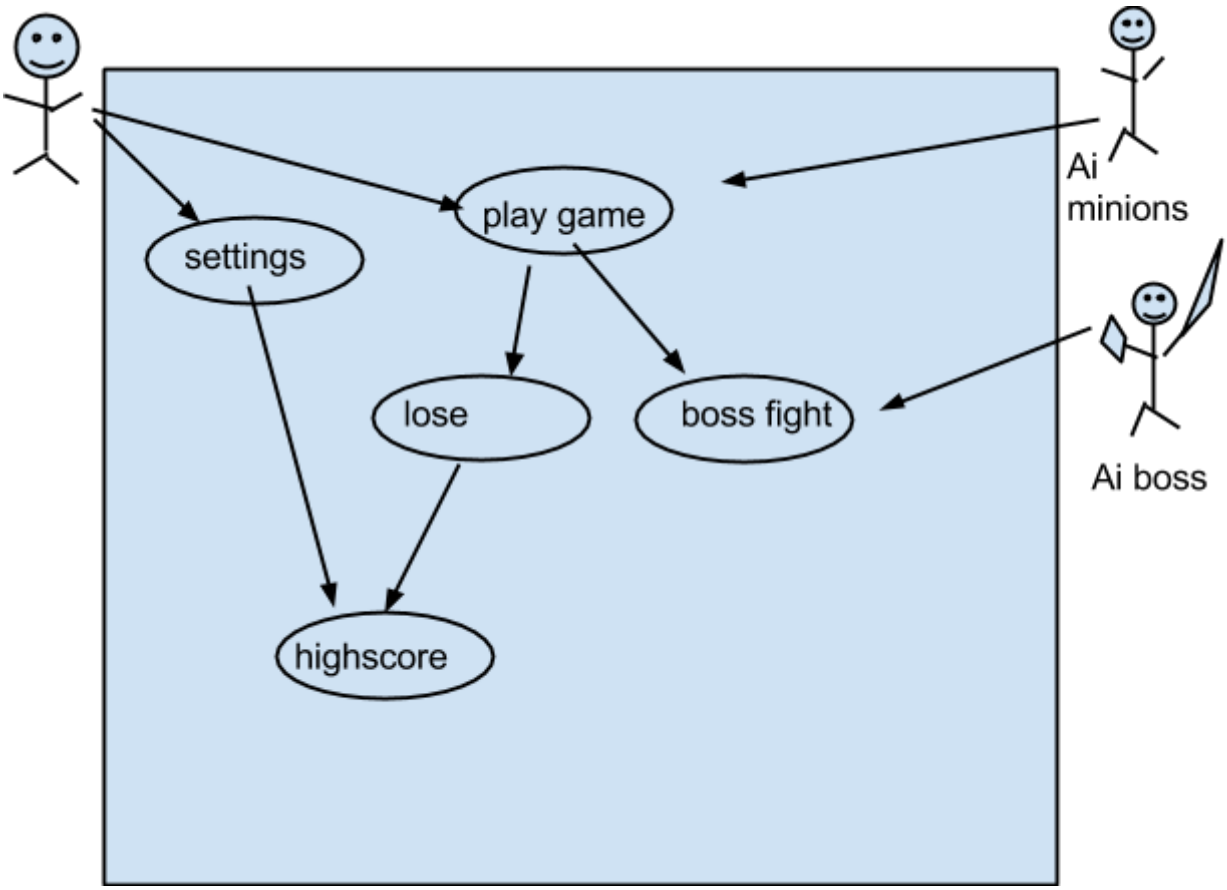

Figure 1 Scenario Diagram

4b. Product Scenario List

The product scenario list is quite simply a list of the product scenarios that will appear in the next section. It is a good idea to either number or name each scenario for later reference, and it can also be a good idea to organize the list so that related scenarios appear together. (Depending on the naming / numbering scheme, they can be grouped into sections and subsections, etc.)

1. Game Starts up
    a. Loads story
    b. Loads level
    c. Loads sound
    d. Louds AI
2. User plays game
    a. the user moves with keyboard or joystick input left right up down
        2.a.1. goes to the right of screen until enemy shows up
            2.a.1.1. fights minions
        2.a.2. fights boss
            2.a.2.1. loads new level and harder ai go to 2.a
        2.a.3. user dies

2.a.3.1. game over

2.a.3.1.1. display highscore

4c. Individual Product Scenarios

Harry has some experience with games as he plays them on his free time. Harry is a freshman, and just started UIC.

1. Game boots up

   Harry decides he wants to play Beat the Prof, to kill some spare time. He is a UIC student, and knows few some stuff about the CS department. He downloads the a game from the playstore using his android phone, and loads it up. He notices the the game company logo, and name pop up and fades out. Then he see a game intro with the story come up He then press any button and see the game menu come up. He hover over the start game and gets a waiting screen.

2. Gameplay

   Harry notices the game is similar to old school games he played and picks up the game fast. Some old school game he played are double dragon, and the gameplay is very similar to this game. He moves to the right screen and starts punching and kicking the weak enemy's. He gets hit two times and has one more until his character dies. He makes to the boss level. Here he rushes into and finds that bosses have special attributes and loses one of his live. He has two more lives and this time he is careful in dodging the boss's attack while doing damage to it. He beats the boss. and produces to the next level. He beats level two as well and has one more life left. In level three he gets to the boss fight and dies.

3. Gameover

   Harry gets frustrated and see the highscore.

5. Stakeholders

5a. The Client

UIC department of Computer Science is our client. CS department funds this game project to compare student studies at UIC with each level of game. Beat the Profs game will give new perspective to study Computer Science with its unique style of play.

The game vendors of social networking sites like Facebook which can be used to distribute this game product to reach all computer science of UIC.

5b. The Customer

The customer of this game product is students from UIC CS department. Any students from the CS department will use this game product and experience the fun in learning Computer Science at UIC. Any students from other departments of UIC will also consider as customer.

5c. Hands-On Users of the Product

- User name/category: Students from Computer Science department of UIC.
- User role: User will play the game effectively by understanding game instructions and experience more fun in studying in Computer Science.
- Subject matter experience: This game product expects user to have basic usage of English language skill.
- Technological experience: This game product expects user to have basic knowledge of using computers like using keyboards and mouse controller.
- Other user characteristics:

Education: This game product expects user needs to have basic mathematical skill to understand about game score computed while user playing this game.
Age group: 14 years and above.

5d. Priorities Assigned to Users

- Key users: Students from Computer Science department of UIC
- Secondary users: Students from other departments of UIC.
- Unimportant users: People below age group.

5e. User Participation

User of this game product will move from one environment to another environment by completing each level of game. In order to design the environment storyboards for each level we will get information from environmental engineers. By meeting and discussion with environment engineers for less than one hour, we will be able to sketch different environments that can be included in this product.

5f. Maintenance Users and Service Technicians

Users with Computer Science knowledge background, programmers and computer engineers will play role of maintenance users.

6. Mandated Constraints

6a. Solution Constraints

Constraint 1

Description: The game will run on multiple operating systems.

Rationale: The user may use any devices that supports Unity plug-in. Single user may use different devices like PC, tablet and mobile devices with different operating systems.

Fit Criterion: The game will run on Windows, Mac, Linux and Android operating systems.

Constraint 2

Description: The game will be accessed by users who have English language proficiency

Rationale: Users have different language proficiency. If this game is not accessed by users who do not understand English, users cannot play the game.

Fit Criterion: The product must come with usage instructions described in English language.

6b. Implementation Environment of the Current System

To play the game, user's system should satisfy minimum platform configuration
CPU: 2 Ghz Dual Core Processor
RAM: 1Gb
Free disk space: 1Gb
Operating System: Windows, Mac, Android and Linux

6c. Partner or Collaborative Applications

This game product will not directly collaborate with other applications. However, Unity plug-in will be need in order to run the game.

6d. Off-the-Shelf Software

Adobe Photoshop: Images, story board and icons used in this software are designed using this software package.
Unity packages: The packages and plug-ins from Unity frameworks are used to develop customized functions and game engine.
Mono IDE: It is an open source Integrated Development Environment that can be used to develop this game product.

6e. Anticipated Workplace Environment

If the user play the game in personal computer, user should sit in proper position and keyboard and mouse should be in considerable distance to use. Monitor of the system should have proper resolution in order to display rendered images properly. If the user wants to experience the background of the game, then system should be connected to either speakers or headphones

6f. Schedule Constraints

The working demonstration of this game product should be presented within 3 months after the development team has received this requirement specification report. All the deliverables and packing of necessary gaming libraries should be completely achieved before the product has been set for release.

All future release and updates should be done via online where user does not need to check for regular updates. Any updates of the game product should be notified to the user and should get user permission before start updating. All major release should be done every quarter and minor release should be done on demand.

6g. Budget Constraints

This game product should be developed and released with low budget. The cost involve in development of this product should not be more than $57,600 (USD).

This budget is based on four developers working for three months, approximately sixty business days with 8 hours/day of work, with per hour wage of $30 / hour. All necessary packages development and testing should be done within this time period. Any delay in deliverables or release will be handled by developers and cost for their additional work will not be covered in their wages.

7. Naming Conventions and Definitions

7a. Definitions of Key Terms

Beat Em Up - Beat Them Up

CS - Computer Science

UIC - University of Illinois at Chicago

Content

Beat Em Up - An abbreviation for “Beat Them Up”. A colloquial term used for games that involve fighting.

CS - The department from which the game’s bosses (professors) originate from.

UIC - The setting of the game, where the meat of the game takes place.

Motivation

We chose “Beat Em Up” purposefully over a term such as “Fighter” or “Battler” because it invokes an almost cartoon-like connotation, and this game is meant to be very light-hearted.

Examples

Tekken or Viewtiful Joe are both examples in the mainstream videogame industry of a Beat Em Up.

Considerations

We use these all liberally because they are the focus point of the game, but will be sure to explain them in-depth to the client (the player), so that there isn't any data loss in the conversation.

7b. UML and Other Notation Used in This Document

Content

Arrows

Circles

Motivation

Arrows - In a previous example we used these to portray when an "actor" or character in the game would be in a specific scenario and which one that is.

Circles - Highlights a "feature" of the game, which is abstracted pretty far out, and could benefit from a more narrow definition.

Considerations

Arrows - Used primarily in section 4a.

Circles - Likewise used almost exclusively in section 4a.

Example

See section (4a).

7c. Data Dictionary for Any Included Models

Classes/Objects:

Player

Boss

Level

Weapon

These **classes** are essential to the infrastructure of the game, as to have a fighting game, you need at it's core a player and an enemy. The weapon is an add-on that exists to give the game more personality and complexity. The level exists for the same reason, lending more scenery to an otherwise one-dimensional experience.

For Example

The **weapon** and boss classes are the most dynamic. Some examples of the former (keeping with the school theme) would be a calculator, smart phone, VR headset, server rack, laptop, etc. The boss in this context is a CS professor, so they could be categorized with attributes such as difficulty, friendliness, strictness, etc.

Considerations

Create an **i**nstance of the player class to start the game off, and place him/her in an instance of the level class, giving a back reference to the player object. This player can then have any number of Weapon instances, so that would be best organized with a list or array.

8. Relevant Facts and Assumptions

8a. Facts

This game product will run in all operating systems such as Windows, Mac and Linux which downloads and supports Unity plug-in. The user interface of this game product will render as designed regardless of operating system that users use to play this game.

To provide better background experience, this game product shall support modern speakers or headphones that are connected with systems which player use to play the game.

As future enhancement, this product can be re-engineered to run on all mobile operating systems like Android and iOS. In addition to that, this game can be published with game vendors in social networking sites. Users will play this game either in mobile devices or online, this game product shall have expandability features to support any future enhancements.

8b. Assumptions

Following are the assumptions that are considered into account in developing this game product:

Developers have installed all necessary softwares such as Mono IDE, Visual Studio, testing tools to their computers. Any softwares and hardwares that are needed on part of this development must be in available to developers.

User has personal computer that meets minimum hardware requirements in order to run this game product.

User uses at least anyone of the operating systems like Windows 7/8/8.1, Mac OS and Linux in his/her personal computers.

User has internet connection and allows to download Unity plugin from internet if it is not already installed in user personal computers.

User has at least minimal English language proficiency to understand the game instructions. Additionally, user has basic Mathematical knowledge to understand his/her score points that are computed during his play.

II. Requirements

9. Product Use Cases

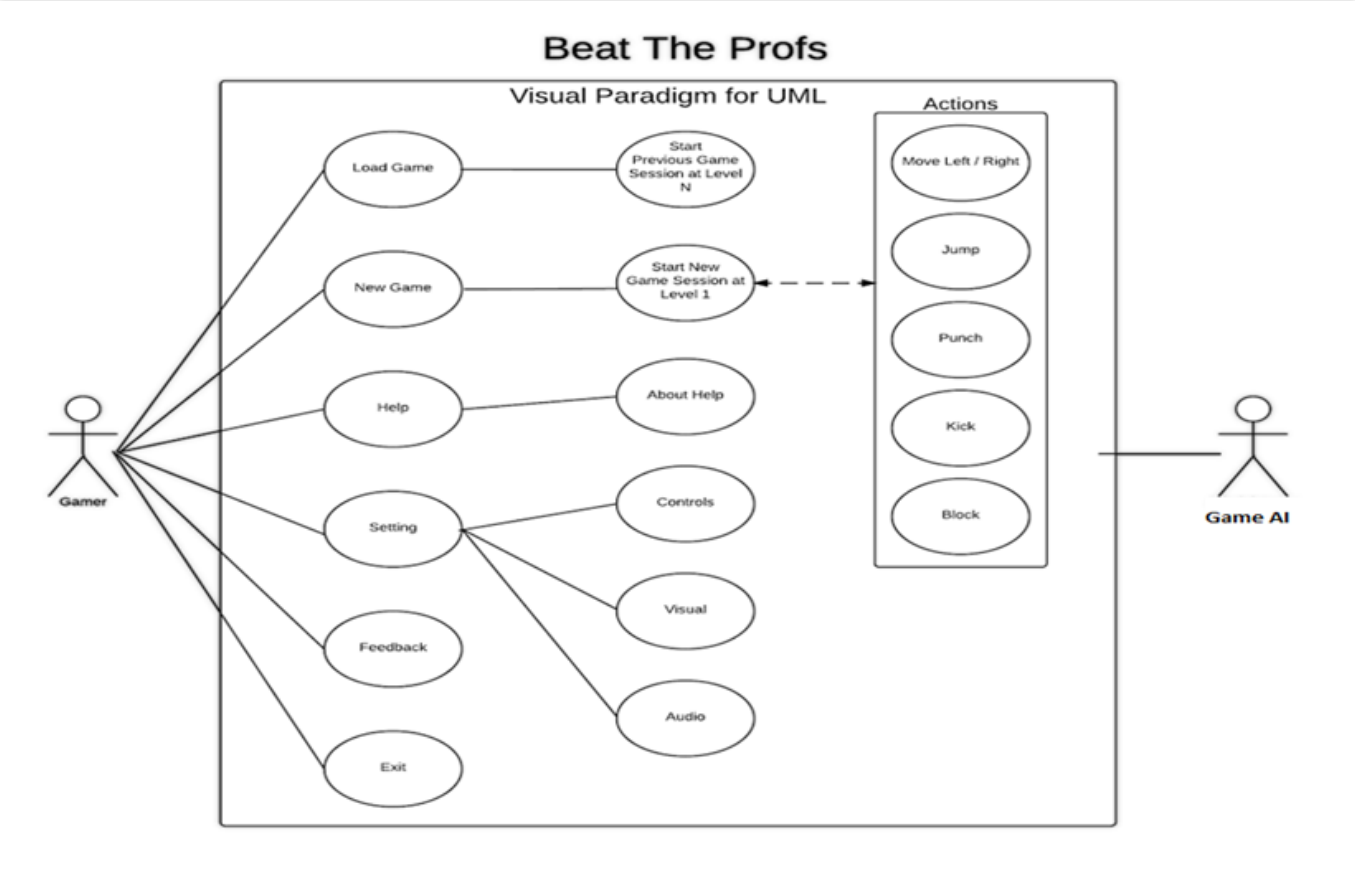

Figure 2 Use Case Diagram

9a. Load Game

Use Case Name	Load Game
Participating Actors	Player
Flow of Events	1 The application redirects user to the Main menu of the game when user starts the game. 2 Main menu has following options A Start Game B Load Game C Settings D Help E Feedback F Exit 3. From Main menu user click Load Game option to continue the game from previous play
Entry Conditions	The application redirects user to Main menu
Exit Conditions	The user played the game from last stored state
Quality Requirements	The selected action should be performed within 4 seconds after user requests it

Table 2 Load Game - Use case

9b. New Game

Use Case Name	New Game
Participating Actors	Player
Flow of Events	1 The application will show the Main menu when user load the application 2 The application starts a new game session at level 1 when user clicks New Game Option 3 The User will play the application until he /she eithers: a. Finish the current level and moves on the next level b. Finish the game in it's complete Has met the maximum deaths allow (3 lives)
Entry Conditions	The application should show the Main menu
Exit Conditions	The application will display score
Quality Requirements	The selected action should be performed within 4 seconds after user requests it

Table 3 New Game - Use case

9c. Help

Use Case Name	Help
Participating Actors	Player
Flow of Events	1 The application redirects user to the Help section when user click Help option that gives information about instructions for playing the game 2. From Main menu user click Help option to know about game play instructions and other settings information about the game
Entry Conditions	The application redirects user from Main menu
Exit Conditions	User did some actions and read information given in Help Section
Quality Requirements	The selected action should be performed within 2 seconds after user requests it

Table 4 Help - Use case

9d. Settings

Use Case Name	Settings
Participating Actors	Player
Flow of Events	1 The application redirects user to the Main menu of the game when user starts the game. 2 Main menu has following options A Start Game B Load Game C Settings D Help E Feedback F Exit 3. From Main menu user click Settings option to change the game controls, volume and visual controls

Entry Conditions	The application redirects user from Main menu to Settings option
Exit Conditions	User did changes to some settings and it is reflected in the game.
Quality Requirements	The selected action should be performed within 5 seconds after user requests it

Table 5 Settings - Use case

9.e Feedback

Use Case Name	Feedback
Participating Actors	Player
Flow of Events	1 The application redirects user to the Main menu of the game when user starts the game. 2. From Main menu user click Feedback option to give review about the application
Entry Conditions	The application redirects user from Main menu
Exit Conditions	User gave review and ratings about the application and it is posted to support team
Quality Requirements	The selected action should be performed within 4 seconds after user send feedback

Table 6 Feedback - Use case

9f. Exit Game

Use Case Name	Exit Game
Participating Actors	Player
Flow of Events	1 The application redirects user to the Main menu when game is over. 2 From Main menu user click Exit option to close the application
Entry Conditions	The application redirects user to Main menu when game is over
Exit Conditions	The application will save its current state and close it
Quality Requirements	The selected action should be performed within 4 seconds after user requests it

Table 7 Exit Game - Use case

10. Functional Requirements

- The Main Menu has to be the first thing the user has control over.
- The Setting will have sound, configure control, and mode
- Help screen will guide the user when he first starts the game, and needs help.
- Play will start the game and the story. The game will have a quick story intro where the user press a button to interact.
- After that the game begins and user will be able to move his character and start doing the actions.
- Each level will get progressively harder as the user moves throughout the game.
- The Feedback section will give our users the ability to reports necessary bugs if and when they do occur.
- There will be an exit screen that allows user to close the game

11. Performance Requirements

11a. Speed and Latency Requirements

- The application should boot up within less than 10 seconds
- The application should establish connection between player and game artificial intelligence.
- The application should generate proper story boards as per event triggered by player. The response time of storyboards rendering should be considerable minimal and should not introduce any latency
- The application should allow the player to customize their game settings and player key controls. The changes to game settings should be applied to the game within one second.

11b. Precision Requirements

- The application should calculate points accurately to let user to move to next game levels.
- The application should calculate valid hit and miss-hits during fight with opponent and its score is calculated as high score.

12. Dependability Requirements

12a. Reliability Requirements

- The game product should be closely audited by software engineers to ensure that it satisfies customer's expectations
- The game product should not fail more than once during multi-player mode.
- The game product should not have any affect on player machine where it runs during failure. If there would be any impact like latency, game product should notify the customer.

- The game product should have consistent and reliable performance that meets customer satisfaction.

12b. Availability Requirements

- The game product should be ready to run in device where it is installed upon customer start
- The application should not have downtime more than 2%. This downtime may be due to updates or security fixes.

12c. Robustness or Fault-Tolerance Requirements

- The game product should be able to run in single player mode even if the installed device is not connected to Internet
- During multi-player mode, the application should run efficiently even if there is slow network connection.
- The application should be able to run even if the installed device do not have any speakers or if the speakers are malfunctioned
- In case of failure the application should be restore its current state and able to continue from last state.

12d. Safety-Critical Requirements

- The application should not cause any visual impairment problems to the game player.
- The application should not consume more processing power of the installed device that produces abnormal heat
- The game product should be completely verified to ensure that it should not have any malicious code that cause security issues to customer's device where it is installed.

13. Maintainability and Supportability Requirements

13a. Maintenance Requirements

- Any new features will be added during on major release of the game product and new features will be updated to existing game product with less than one minute.
- All maintenance activities and fixes should be documented clearly for future references.
- Test cases should be developed for each new releases to verify that new updates works with existing product.

13b. Supportability Requirements

- Instructions section should be part of the game product that explains user controls to play the game and how to interact with the game.
- Settings section is another part of the application should allow the user to change the difficulty level of the application and volume controls.
- The application should have a feedback option that allows the player to send feedback if there are any flaws in the application and rate the game product.

- There should also be an online support for customer

13d. Adaptability Requirements

- The application should be compatible with all devices that satisfy minimum system requirements and it should be able to run in all operating systems like, Windows, iOS, Mac and Android.
- The application should be able to run in installed devices either with Wi-Fi or cellular data during multi-player mode.
- For future operating systems, necessary updates should be released no later than two months after new future operating systems is introduced.
- The game product might later be sold to other countries like India and European countries

13c. Scalability or Extensibility Requirements

- The application should be robust and enough to handle any loads given by user.

13d. Longevity Requirements

- The game product shall be expected to operate with maximum maintenance, and within the present budget for maximum of five years.
- The game product should be updated regularly to ensure the stability and responsiveness of the game.

14. Security Requirements

The two main security concerns are that of cheating in the game and pirated versions of the game. The first concern should be handled in a simple fashion, such as encrypting the save data of players. If you don't do that then it wouldn't be difficult for a player to modify the plaintext within the save file. The encryption used should be a version of SHA, as it's currently one of the most reliable encryption algorithms.

The second concern would require a very complicated system to combat. There are some third-party systems out there for this, but most players aren't very fond of these systems. It would be best to invest resources in other things such as features instead of this, as there will always be a pirate who finds their way around the system.

15. Usability and Humanity Requirements

15a. Ease of Use Requirements

- The user should be able to download the application easily from the iTunes Store or Android Play Store.
- The application should be easy to use for people of all ages from 6 years and above
- The application can be used by anyone who uses a smartphone or tablet device
- The application design would be intuitive to allow the user to use it with little to no help or remembrance from the previous use
- The application should provide rapid response for the event triggered by player

- The application should notify the player if an error occurs while using the device and provide feedback on how to revert the error.

15b. Personalization and Internationalization Requirements

- The player should be able to select the game level depending on his preferences.
- The player should be able to customize the audio and video controls and game controls.

15c. Learning Requirements

- As mentioned earlier in the ease of use requirements, any user who has experience in using modern touch based smart devices can use the application.
- The application should provide game instructions for new layer on how to play the game effectively to achieve high score.

15d. Understandability and Politeness Requirements

- The application should use symbols and words that are understandable by the user
- The application should provide an interface that would offer informative feedback in a polite context
- The game instructions should be in simple context that allows player to play the game at maximum level.

15e. Accessibility Requirements

- The application should be easy to use for people with common disabilities such as color-blindness, myopia and partially sighted users.
- The application should be easy to use for elderly people by using larger texts and icons where possible, which could be easily readable.

15f. User Documentation Requirements

- The application support website should provide the user with installation instructions to install the application from the Play Store or iTunes Store
- The Help screen should provide general information for player to use the game and how to download games from the Game-store.

16. Look and Feel Requirements

Considering the realities of this project, there won't be any artists on board the project. With this in mind, a minimalistic style should be adhered to. It wouldn't be a failure of a project if the weapons, for instance, were simple geometric shapes.

The feel of the game should be that of a classic run n' gun game such as Contra or Metal Slug. Again, time should be allotted to focusing on the feel of the game, much like the usability, and it's an issue that should be tackled early on, since it would be more difficult to change later on in the design.

17. Operational and Environmental Requirements

The video game will be used in both on the go or at home through mobile phones, and tablets. The Game will be formatted dynamically depending on the platform and device. but it varies from 4 in to 10 in. The user can be the game sitting, standing, or resting down on the bed.

The game can be downloaded from the iOS Apple store, and Google Play Store, this will make it convenient to get new members but also easy to market the game. To update the user will need WIFI or data connection. The developers will release the update every month, or a bug is found a patch that week will be pushed. The update will make sure the game is being ported to new devices, increase performance, change in gameplay, visual features, and multiplayer. Each update will be properly tested before its released.

18. Legal Requirements

18a. Compliance Requirements

The primary legal concern for this application is the use of current professors at their respected universities. This application will walk a fine line to keep the likeness of certain professors while also trying to give them their own persona as well.

18b. Standard Requirements

The game must meet ESRB PG13 game rating. The game's default will be in English, but other language will be available in later updates.

III. Design

19. System Design

19.a. Purpose of the System

Beat The Profs is a system that gears towards the gamer side of users. Our application is going to simple yet high-energetic system which will lead to high uses. Beat The Profs is going to be a game that can be played for a few minutes but will have the ability to last longer if the user desires. Some examples of this type of games are Dumb Ways to Die and Games Gone Wild which both are currently on both app stores.

19b. Design goals

The design goals will outline the desired qualities in our application Beat The Profs. This will act as a set of criteria when making design decisions. Our main purpose for our application is to develop a well-maintain, well-designed application with minimal bugs and crashes. We will achieve this by developing our program with Object-Oriented analysis and design.

- Adaptability: Java will provide a cross-platform portability. With its attributes, the player should not have to worry about operating system requirements. If order to full-fill this adaptability feature, which is very important for the system, we preferred to program the game in Java and will be sacrificing some performance advantages that other programming languages offer like C or C++.
- Efficiency: The system is going to be responsive and able to run with high performance. Responsiveness is going to be an important aspect in the system because with any delay, the user's experience will be at a minimal and at the risk of not interacting with the application.
- Reliability: The system will be bug-free and consistent within each level. The system should not crash with any input given by the user.
- Usability: The system will be designed so that user can easily interact with the system without any previous experiences. User-friendliness is an important aspect in the system but this also does not mean that the gameplay is made easier. If gameplay was made easier, we risk that the user will become uninterested sooner.

20. Current Software Architecture

There is no current software architecture exists. However, below is the proposed software architecture.

21. Proposed Software Architecture

21a. Overview

The architecture model that suites for the development of this game product is three tier architecture model. This model consists of three layers, interface layer, application layer and storage layer. The interface layer allows game player to interact with game. It displays information and provides user controls that allows player to play the game. Application layer deals with rules of the game and checks whether player uses respective controls while playing. The storage layer deals with storing user information that they enter such as player name and store high score details.

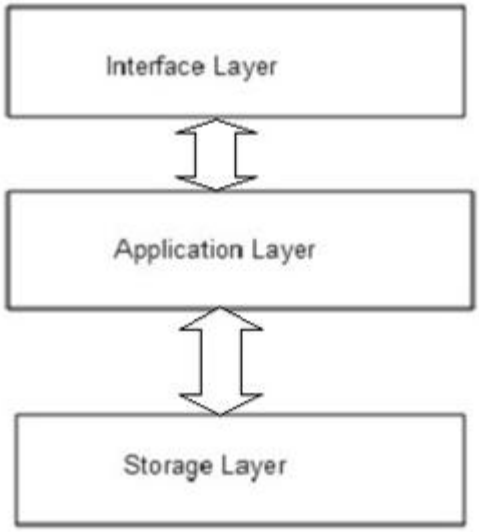

Figure 3 Three-tier Architecture

21b. Class Diagram

The below class diagram shows preliminary relationships between the class that are used to build the game product.

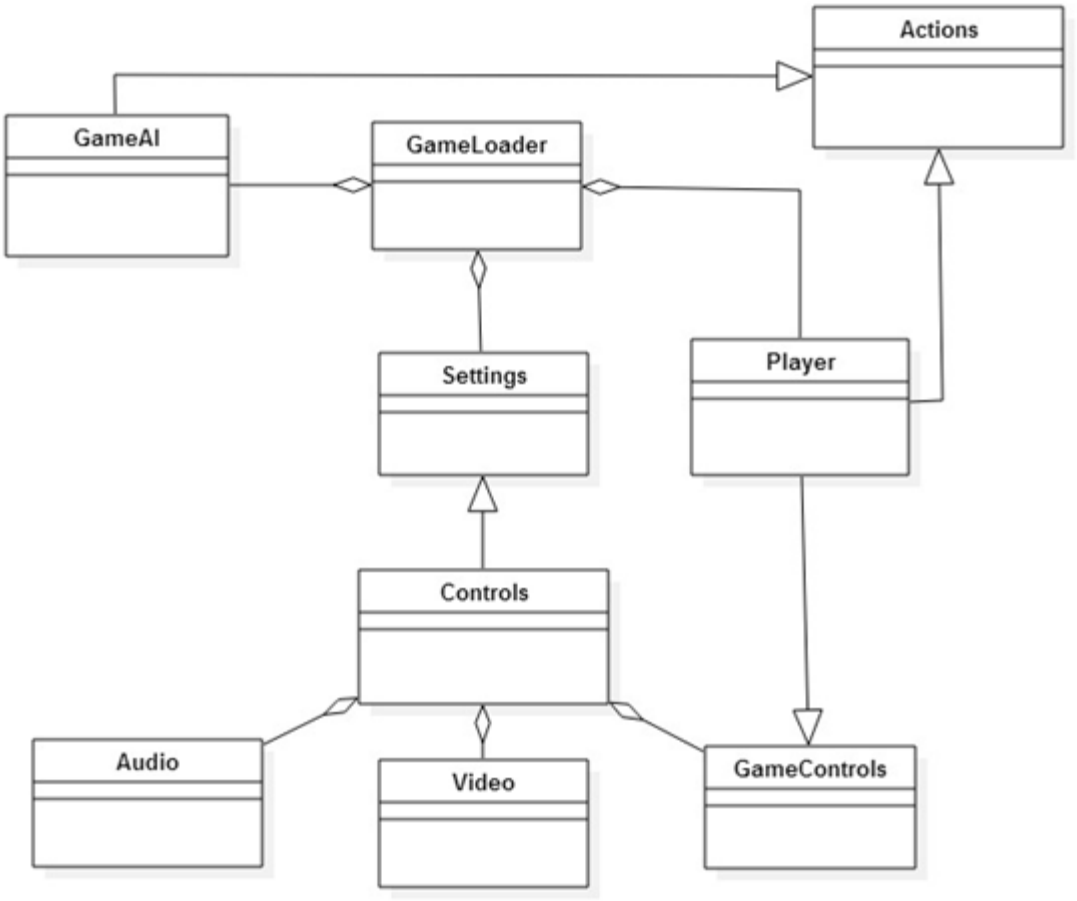

Figure 4 Class Diagram

21c. Dynamic Model

The sequence diagrams for the use cases are given below. Each sequence diagram corresponds to a use case discussed earlier.

- Sequence diagram for the use case Load - to allow the user to load the game

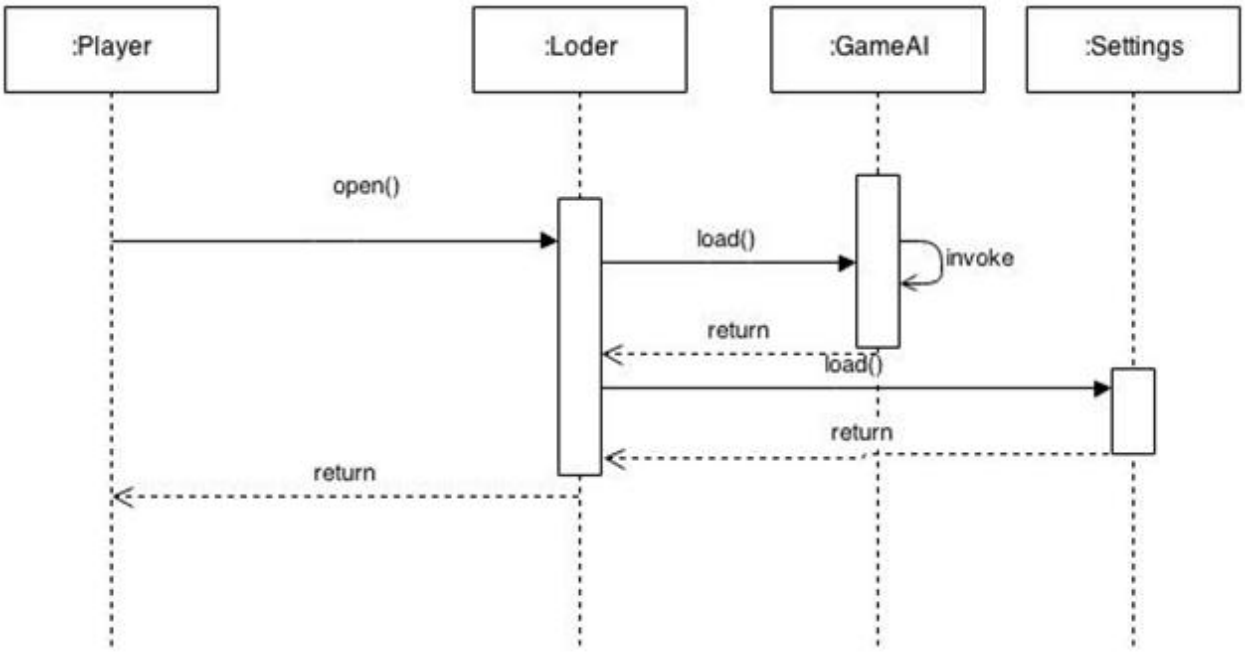

Figure 5 Load the game - Sequence Diagram

- Sequence Diagram for the use case - Settings to change the game settings

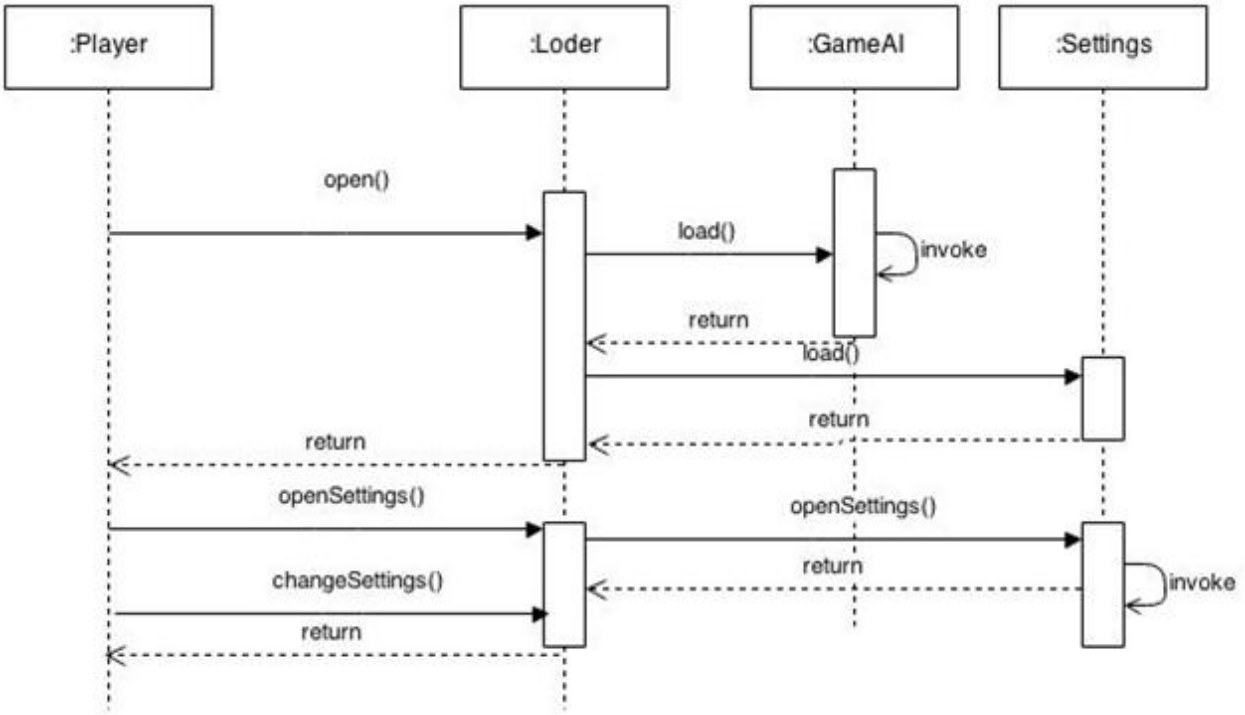

Figure 6 Change Game settings - Sequence Diagram

- Sequence diagram for the use cases Play, High Score, Help, Feedback and Exit

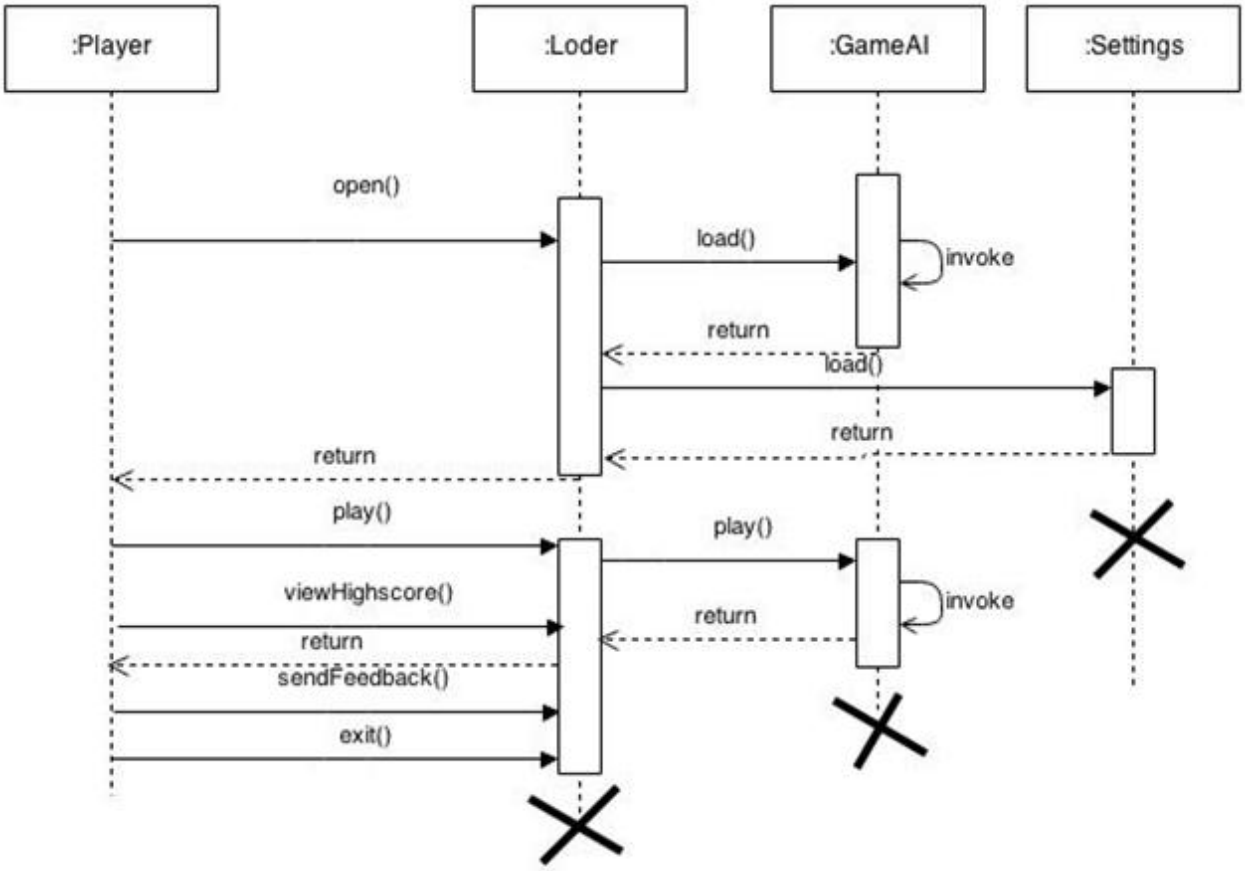

Figure 7 Play Game, View Score, Read Help and Send Feedback - Sequence diagram

21d. Subsystem Decomposition

The Subsystem Decomposition diagram shows components involved in the system. They are shown in UML packages. Below are the subsystem of this game product.

- Game Interface
- Game Loader
- Game AI
- Mini Game Database

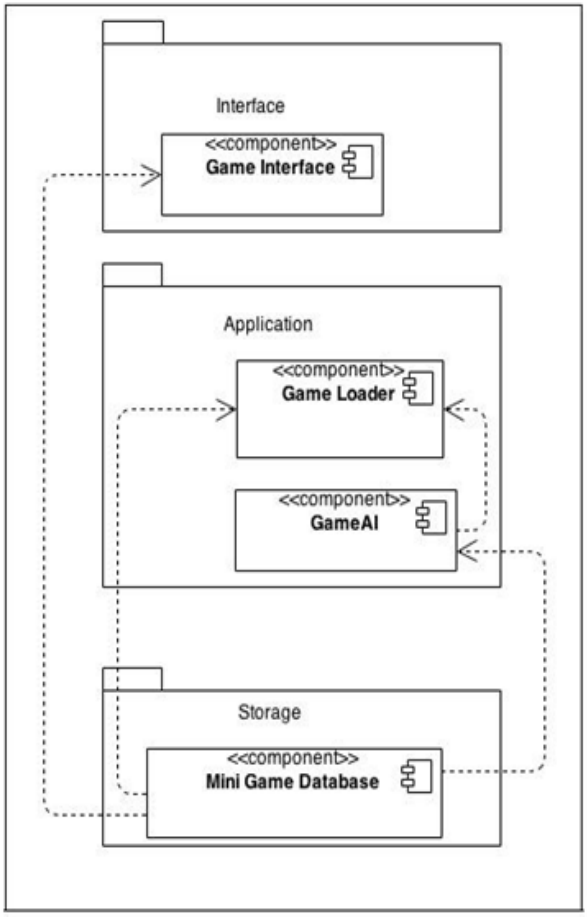

Figure 8 Subsystem Decomposition Diagram

21e. Hardware / software mapping

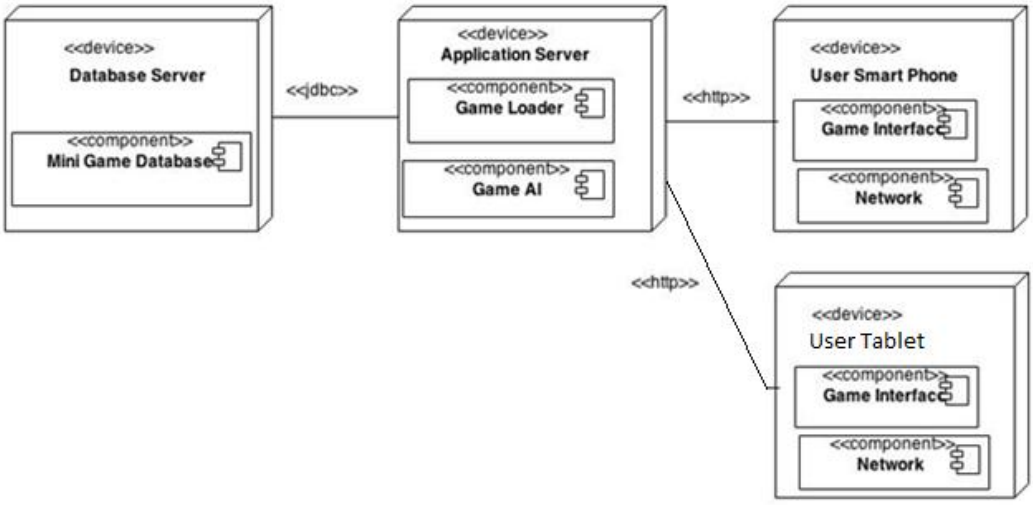

Figure 9 Hardware and Software Mapping

21f. Data Dictionary

Game Interface - It is responsible for loading user interfaces for the game. It provides menus for the game and retrieve the game screen details in order to set game video resolution

Game Loader - It is used to load the game AI, game settings, game controls and player settings. In addition to that game state is loaded to allow user to continue from last play

Game AI - This is game engine which will be played with player. Player score will be monitored and game level will be change accordingly to player game score.

Mini Game Database - Mini game database is used to load game state and game score of AI and player. In addition to that, game settings will stored and it will be maintained throughout the game.

Settings - It manages game settings, player settings , audio and video settings.

21g. Persistent Data Management

Mini Game Database - This game database is inbuilt with game product. It is used store player game score, current game levels and sate of the game when player left in last game. When user comes back to play the game, all the current state such as game level, game audio level, video resolution and game score will be retrieved from the database.

21h. Global software control

The control flow selected for this game product follows event-driven control flow paradigm. When player request some operations clicking a button, a event will become available and it will be dispatched to respective object based on information associated with the event. This includes clicking on "Play" button. Once user clicks on "Play" button, an event will be triggered and an appropriate object will be dispatched which will result in starts the game. The event is same for all features associated with this game product. Whenever player clicks on a button, appropriate event is triggered and respective screen will be displayed in player's device.

21i. Boundary conditions

The following is the boundary conditions of the system that specifies how the system is started and shut down and how it can be handled with major failures such as file corruption and software failures.

Start-up and Shutdown - Whenever user starts the game product a game instance is created starts run by restoring its last shutdown state from mini game database. The player can close the game application either by clicking "Exit" button option or terminate the session forcefully by closing the game application.

Exception Handling - This game product is updated periodically during major releases. Whenever there are updates available to game product, game product is updated when user connects his device to network. Due to network outages, the application software might not be updated completely, we need to handle these issues. In case of any network failure, the system should be able to restore its previous product version. There are exceptions handling mechanisms available in Java to handle unexpectedly closed sockets.

22. Subsystem services

- Game Loader : The game loader is responsible to load the game engine and game settings. Through game loader, player can able to access the game settings to control audio, video and game control keys.

- Game AI : The Game Artificial Intelligence is a game engine that plays with player. For each events triggered by player, Game AI provide responses by moving the game objects and storyboards as per user game key controls. Whenever user finishes a level the Game AI allows user to enter into next level with increased game difficulty. During each level, the player score is calculated and shown to player at the end of the game level.

- Mini Game Database : Mini Game Database provides memory to store the state of the player game and details such as player name and their game settings. When user close the game during middle of any level, current state of the game will be stored and it will be restored and accessed by player when he plays the game again.

- Game Interface : Game Interface provides features that allows user to interact with game product. The features include allows player to start the game, change the game settings, read game instructions, continue game from last play, exit the game and send feedback. The send feedback option allows player to give ratings and feedback about the game product.

23. User Interface

Below is the design for the user interface.

The Main Menu:

Figure 10 Main Menu

Play :

Below is the story what the whole game is based on

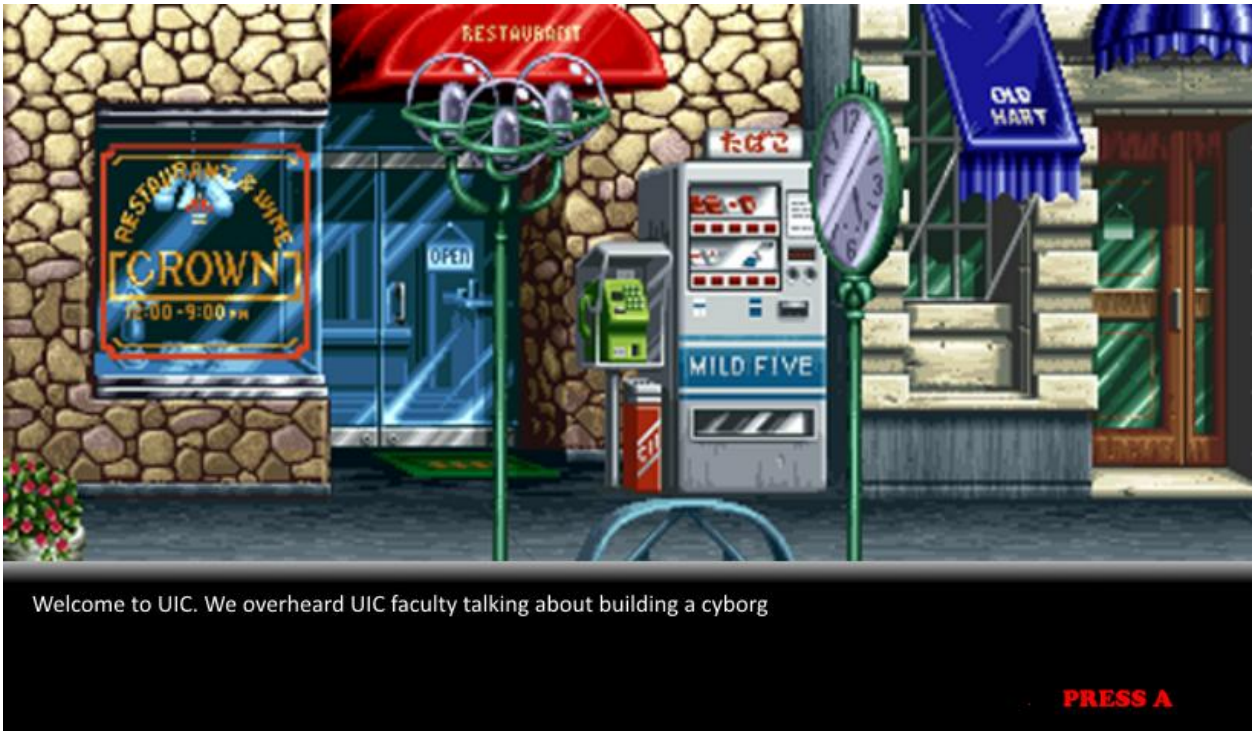

Figure 11 Game Story

Below is character, and enemy

Figure 12 Game Characters

Setting:

Figure 13 Settings screen

24. Object Design

24a. Object Design trade-offs

Efficiency vs Reusability: Efficiency is important, because it's ideal that the user has a seamless experience, not one hindered by a low framerate. A low framerate is a direct result of poor efficiency. Reusability on the other hand will not be a concern, since the codebase will not be used for any further projects. This is not the case because it is a school project, it would be different if it was a real world project.

Functionality vs Usability: As a rule of thumb, usability takes a precedence to functionality in a videogame. This is a recent trend for the industry, as the audience is much more diversified as of late. In the past, it was most important to ensure every feature connected to each other in a solid fashion, but with the event of infants and older people now playing games, it's more important that it's easy to understand how to play.

Space vs Speed: It depends on the platform, but in general speed will be much more of a concern than space. It comes down to the question "should you make the user wait longer during the download or longer during the gameplay?". The user will probably not remember the former once they're playing the game, but it would be very easy to get frustrated if you were, for example, forced to wait for two minutes in between levels.

24b. Interface Documentation guidelines

Examples	Rules for Naming	Identifier Type
Package UIPackage;	The package name should be named with unique prefix that describe about its underlying functionality	Packages
Class Level; Class Enemy_Character;	Class names should be nouns with the first character uppercase. With mixed cases, the internal words will be uppercase with an underscore separating them.	Classes
Interface Settings;	Interface names should be uppercase for the first letter.	Interfaces
Insert(); Destroyed_character();	All methods are verbs starting with an uppercase first word. All internal words from there will lowercase with an underscore separating them.	Methods
Int i; Char c; Int number_of_lives;	Variable names should be short and meaningful. They should be lowercase with an underscore separating internal words. One-character variable names should be avoided except for temporary variables.	Variables
static final int NUM_ENEMIES = 5;	Constants names should short and meaning as well as ALL uppercase letter with an underscore separating the internal words.	Constants

Table 8 Interface Documentation Guidelines

24c. Packages

The game is divided into three packages:

- UIPackage - This will include subsystems like main menu, settings, feedback and highscore screens that are implemented with GameInterfaces

- DBPackage - It holds mini game database to store the user game to resume from last play, cusomized game settings and user profiles.

- GEPackage - Game Engine Package holds Game artificial intelligence and game loader that handles game control flow depends on player key control action.

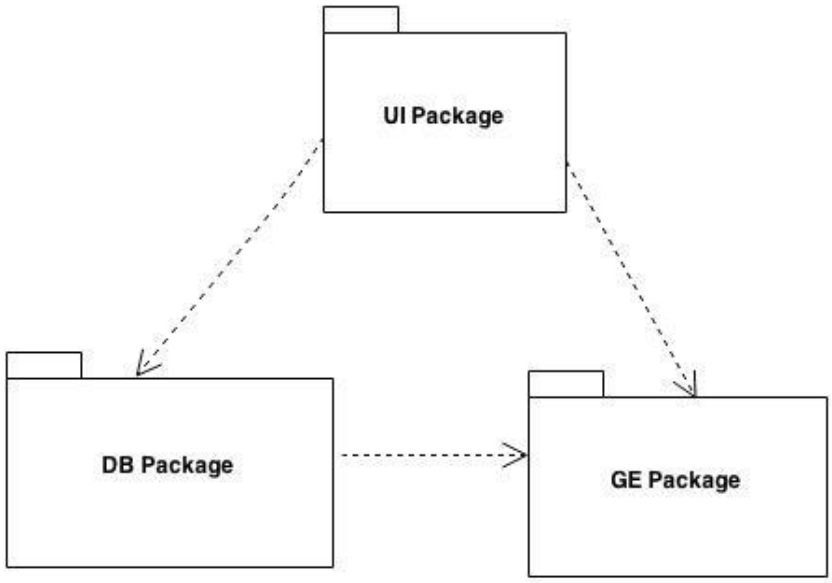

Figure 14 Package Diagram

24d. Class Interfaces

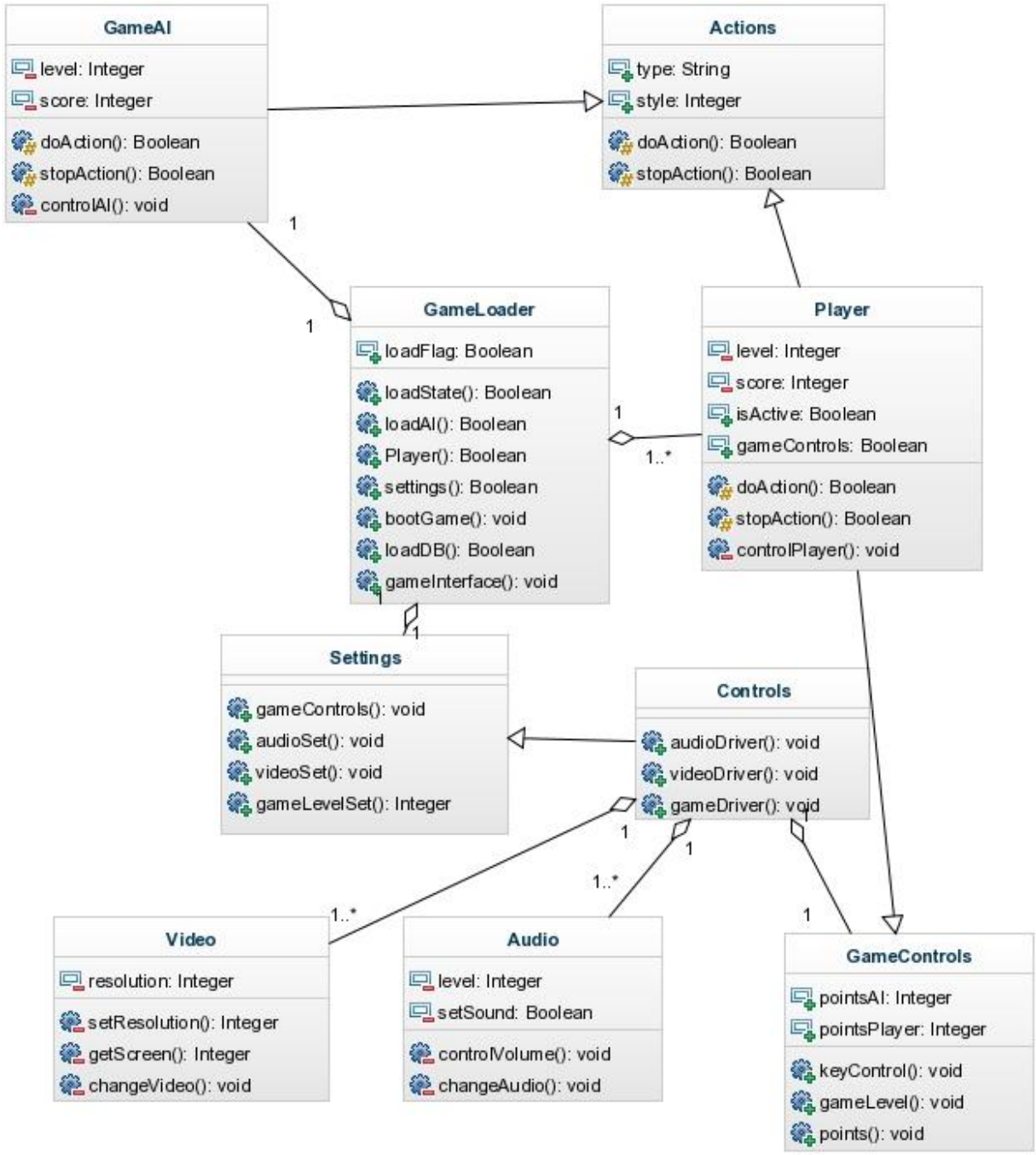

Figure 15 Class Interfaces

GameAI - This is the game engine which will be played with player. In addition to that it will monitor player hits and decide the game levels

Actions - All user control actions should be defined here. User controls will be right hit, left hit , left kick, right kick, jump, down, run and fast walk.

GameLoader - It is responsible to load the game AI and user's last state when it was left from game database

Settings - Settings have drivers function that is inherited into Audio, video and Controls class

Player - Player has functions such as game actions, player details and attributes such as player name, game score and last game state details are used to store in game mini database

Controls - This is the parent class for Audio, Video and GameControls that contains audio driver function, video driver function and player key driver function.

Audio - Volume control keys are defined in this class by implementing methods from Controls class

Video - This class has functions that allows user to change screen resolution by implementing functions from Controls class

GameControls - User key controls are defined in this class for each of player actions by inheriting the class Actions

IV. Test Plans

25. Features to be tested / Features not to be tested

Beat The Profs features will be tested to insure that this application is executing at a high standard. The following is a list of general features that will be fully tests before each release.

- New Game
- Load Game
- Save Game
- Exit Game
- Change Settings
- User's Controls
- Feedback
- Help

Features not to be tested would be features that we can not control. Storing user's information is required for our application for previous saved states and we can not test the accuracy of this information contained in the Beat The Profs Database or the reliability of the servers storing this information.

26. Pass/Fail Criteria

A test will be considered a pass if and only if the actual results of a certain test is the expected results specified in the test cases. Any deviation from the expected results will be considered a failure for that specific test case. Feature(s) / implementation(s) will be considered to have pass after all associated test cases have been conducted and be considered for code approval if and only if that test cases has pass over 95% of the time. If test cases is below this threshold, the developer will have to call for debugging for the associated code.

27. Approach

The testing approach for this application will have to take place throughout the process of development in parallel. Each new implementation of a feature will be tested as if it's the final features for the application.

All developers are expected to perform predetermine white-box unit testing as well as generate reports for each set of test cases before any code is submitted to the project.

One developer for the project will have the task of either accepting / rejecting each developer's code by first inspecting that developer's code and determining whether the test case reports are acceptable or need further review.

28. Suspension and Resumption

Suspension of the application resulting in failed test cases according to the pass criteria will be brought back to the developers for debugging and modification for a patch. Developers will be given reports on the failed test cases so they can reproduce the fail test case scenario. The updated code will follow the previous steps for submission until it passes.

29. Testing Materials (Hardware / Software Requirements)

As this is a web-based game simulator, it must be compatible with the most popular web browsers, which would include Chrome, Internet Explorer, Firefox, Safari and Opera. The application must be tested on all listed web browsers, which are freely available.

Also, the application must be compatible with all operating systems like Windows, OS X, and Linux. The application should be tested on a machine running each of these operating systems.

30. Test Cases

Test Name	Entry Condition	Flow of Events	Exit Condition
Application - Load Application	User is in the home screen in the application	User has options from here User will decide how to proceed to A) New Game B) Load Game C) Help Page D) Settings Page E) Feedback F) Exit Application	User will be on desire page after choosing
New Game	User will start a new game	User starts a new game Proceeds until User has finish playing	User will either save current game or exit current game which either way be directed to the application home page
Load Game	User will load a previous saved game	User loads game User will proceed to interact with the previous saved game	User will either save current game or exit current game which either way be directed to the application home page
Exit Application	User is currently in the home screen in the application	User will proceed to home screen User will select Exit	User exits the application and the application is closed

		button to exit the application	
Settings - Controls	User will enter the Settings page	User can change the controls for the player's avatar	User will proceed to the Settings page
Settings - Visual	User will enter the Settings page	1. User can change it's avatar with some different options like color, type, size, …	User will proceed to the Settings page
Settings - Audio	User will enter the Settings page	1. User can change the audio in the application like the theme song, sound effects,…	User will proceed to the Settings page
Feedback	User will enter the Feedback page	1. User will be able to enter text to any feedbacks whether good or bad, bugs,…	User will proceed to the Home page
Help	User will enter the Help page	1. User will be inform will the information needed to interact with the application.	User will proceed to the Home page
Avatar Controls - Right	The user will be in interact mode with it's avatar	1. When the user's right button is active, the avatar will proceed to the right	User will proceed to the same screen
Avatar Controls - Left	The user will be in interact mode with it's avatar	1. When the user's left button is active, the avatar will proceed to the right	User will proceed to the same screen
Avatar Controls - Jump	The user will be in interact mode with it's avatar	1. When the user's jump button is active, the avatar will proceed to the right	User will proceed to the same screen
Avatar Controls - Punch	The user will be in interact mode with it's avatar	1. When the user's punch button is active, the avatar will proceed to the right	User will proceed to the same screen
Avatar Controls - Kick	The user will be in interact mode with it's avatar	1. When the user's kick button is active, the avatar will proceed to the right	User will proceed to the same screen
Avatar Controls	The user will be in	1. When the user's block	User will proceed to the same

- Block	interact mode with it's avatar	button is active, the avatar will proceed to the right	screen
Collision Detection	The user will be in interact mode with it's avatar	1. The avatar must detect collision when appropriate during the	User will proceed to the same screen
A.I. Movements	The user will be in interact mode with it's avatar	1. A.I. movements for enemy avatar must accept and follow all the rules given	User will proceed to the same screen

31. Testing Schedule

Inspection	Throughout Development
Unit Testing	Duration: 1 week
	Begins: 2 weeks before each release
Integration Testing I	Duration: 1 week
	Begins: 2 weeks before each release
Usability Testing	Duration: 1 week
	Begins: 2 weeks before each release
Integration Testing II	Duration: 1 week
	Begins: 2 weeks before each release

V. Project Issues

32. Open Issues

Some issues we might face are using teacher's name and characteristic of the teacher for the game. We want to use the teacher's name and characteristic to get the best feeling overall for the game, almost as the user experiences first hand by seeing how they are, and how each teacher has different difficulties, and its special abilities. Having a user experience where something they can relate to in a game will be a new, rewarding, and enjoyable experience.

33. Off-the-Shelf Solutions

33a. Ready-Made Products

Ready made products will have lower cost, less work, less problems, but most of the work will be done. By having most of the work done, it will be harder to change the game to feel, and appearance to what we want. The user experience is what is important to us, this will make the game enjoyable, relatable and something a user won't forget. Also most of the readymade products are old school game when they were created in 1995, so it might be harder to port the solution to our new platform. It might be best to create it from scratch, and not use the read made solution.

33b. Reusable Components

We will most likely use or buy the fighting box/hit box as it will be the hardest to create in a game like this, and we want it to be accurate so the user is not frustrated if he hits the enemy, and doesn't do damage.

33c. Products That Can Be Copied

There are many games that are similar to Beat the Profs, such as Double dragon, Battle Toad, and Little Fighter to name a few. All of these games are old school games meant for 16-bit game console, so it would be hard to find a new latest version of it now. Each of this game has its own objective, character, and story. The fighting style is very similar thou, it is beat up em, as many hordes of enemies come, and at the end there is a boss fight. We can take ideas from them, but the different platform, and old hardware means we will still need to make the game from scratch. We can still copy the logic, and ideas of the game. We will spend less time on logic, and ideas because of this.

34. New Problems

34a. Effects on the Current Environment

Beat the Profs will not cause harm to current implantation environment. It won't harm the people in any way. The player won't be affected in any way playing the game. The means the high score will be consistent, and secure. The user will be warned of updates an hour, and will be securely and forcefully prompted to exit the game so system can update.

34b. Effects on the Installed Systems

There is no pre-existing system, and this will be the first, so there will no effect on the system.

34c. Potential User Problems

The game can be very addicting, and very violent. It is advice to stretch, and take breaks every hour between game play. User's finger might get blisters, and user may experience lack of sleep if prone to playing for long hours. The user should be taking care of his own body at his own time. The game only is for entertainment and not taken any other way.

34d. Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

The game doesn't have any limitation with any new technology as it will be using the same infrastructure.

34e. Follow-Up Problems

We want at least 60% of the user to be able to get the final Android boss level, if they cannot get there, we will need to decrease the game difficulty so that more people can get to that level, and are less frustrated. Also make sure the game doesn't hog all the system resource, and graphic usage.

35. Tasks

35a. Project Planning

The development of Beat the Profs will be planned in the following way: Sprints (or phases) will have set deadlines and will require the team to section off and complete their own tasks within the sprint. It can be look at as moving from Design to Implementation to Deployment, but the sprints are not necessarily split up in that way. Each sprint should take about three weeks to complete, and then there will be a testing phase after the product is finished.

35b. Planning of the Development Phases

Phase 1 includes most of the design and implementing the core engine including the player movement and basic main menu.

Phase 2 will see most other features added like attacking, enemy players, and the bosses.

Phase 3 is the time to clean up all of the features (mostly the main menu) and implement the art.

Phase 4 is time for polishing up everything to make sure the game flows and the player is able to finish the game.

Phases can be synonymous with sprints.

36. Migration to the New Product

There is no current product, so we will not be migrating from existing any products to the new system. We will be starting the new product from scratch

37. Risks

All projects involve risk—namely, the risk that something will go wrong. Risk is not necessarily a bad thing, as no progress is made without taking some risk. However, there is a difference between mismanaged risk—say, shooting dice at a craps table—and managed risk, where the probabilities are well understood and contingency plans are made. 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. The following risk being the most serious:

• Inaccurate cost estimating - We know that the estimate cost of product will be, but if we miss a deadline, or a feature takes longer it will cost more.

• Creeping user requirements - having specific requirements of UIC's teacher, background and grads.

• Low quality - game could a be a bad graphics with sloppy frames. This will make the game unplayable, and not experience it fully.

38. Costs

The Android Play Store requires a one-time payment of $25.

The iOS App Store requires a yearly recurring fee of $99.

The development environment will most likely be free.

Assuming a team of four programmers, and an average wage of $30/hr and an estimated development time of 3 months, a possible total salary would be $57,600.

An artist will also be required for the pixel art and assuming an average wage of $25/hr will cost about $12,000.

Total fees will be around $69,700 and another $100 for every year you wish the app to remain on the iOS app store.

39. Waiting Room

There are many extra requirements that we would like to be implemented in future versions. These include:

- Special abilities for the player. Ex. Summoning quad-copters to attack your enemies.
- An alternative main character that is female, to appeal more to a broader audience.
- An extra boss after the final boss to surprise the player and provide extra

challenge.
- An online leader board that allows you and your friends to compare scores.

40. Ideas for Solutions

RoboVM may be worth looking into for developing an iOS app with Java, as you can't use it natively. Only Objective C, Swift and C++ is allowed for that.

If there is a lag when it comes to saving the game and there isn't enough time to develop a "wait" while saving, then execute the saving method on an asynchronous thread.

Collision detection can be very resource-intensive if hard-coded by hand, so maybe try to find an already implemented solution.

41. Project Retrospective

A high priority was put on learning and utilizing IceScrum. Creating tasks and assigning them proved easy using this tool, though it took a bit of time to learn the tool. Ideally, we would have used a tool like Jira which has more features than IceScrum and is a bit less outdated. The Email and instant message communication at some times was not enough due to the need to constantly check it, but that was perhaps at fault of the group. A higher priority should have been placed on keeping up with other member's progress on the project.

Glossary

Environment: Any picture that depicts real world. In each level, different side scrolling pictures are used to give users a real world experience.

ESRB - Entertainment Software Rating Board

GAMEAI : Game Artificial Intelligence

GAMEUI : Game User Interface

IDE: Integrated Development Environment, is a set software tools that helps to develop the game **w**ith computer programing languages. Example, Mono IDE and Visual Studio

Unity plugin: Unity plugin is a freely available software to run the game in any personal computers.

References / Bibliography

- http://www.cs.uic.edu/~i440/
- http://www.icescrum.org/en
- http://unity3d.com/pages/2d-power
- http://monodevelop.com/

Index