GROUP
2I

A DAY IN BILKENT
FINAL REPORT

CS319 OBJECT-ORIENTED SOFTWARE ENGINEERING

ASAF KAĞAN BEZGİN	21402006
ALPER KILIÇASLAN	21502103
IMRAN HAJIYEV	21503442
MUHAMMAD ASKARI IQBAL	21504228
ENES VAROL	21604086

SECTION 02 | SUPERVISOR: UĞUR DOĞRUSÖZ | COMPUTER ENGINEERING

1. INTRODUCTION

After delivering the first iteration of final report, implementation process take place. Programming of the game carried on with GitHub which is a commonly use repository. IntelliJ which is an advanced java programming environment used as an IDE because of the integration of it with GitHub. IntelliJ has a feature which enables the user to push the necessary changes to GitHub repository making the life of the coders easier. GitHub repository is created by Muhammad Askari Iqbal. Social media software such as Skype, Google Hangouts, etc... is not used because each group member strictly followed the meeting schedule.

Certain tasks are divided among the group members. Muhammad Askari Iqbal worked on game engine. Imran Hajiyev worked on user interface, Enes Varol worked on Game-Object, Asaf Kağan Bezgin and Alper Kılıçaslan worked on sub-classes. When the second iteration finished, most of the design of the game and game logic were already done.

Design & Analysis were done detailed and accurate. Therefore, instead of changing or fixing the design, implementation take place with the existing one after second iteration. A reliable design let group members focus on the implementation of promised features. However, after first iteration, design is fixed according to MVC architecture, Singleton design pattern and Façade design pattern.

2. DESIGN CHANGES

After the first iteration, design is changed according to MVC(Model-View-Controller) design architecture, Singleton design pattern and Façade design pattern. After applying MVC to the game, GameObject and GameManager represent model, UserInterface Subsystem represent View and Listeners represent controller. Singleton design pattern is used between GameManager and FileManager. Façade design pattern is used between GameObject and GameManager classes. Latest versions of some diagrams indicating high-level and low-level design is shown below.

Use Case of Multiplayer

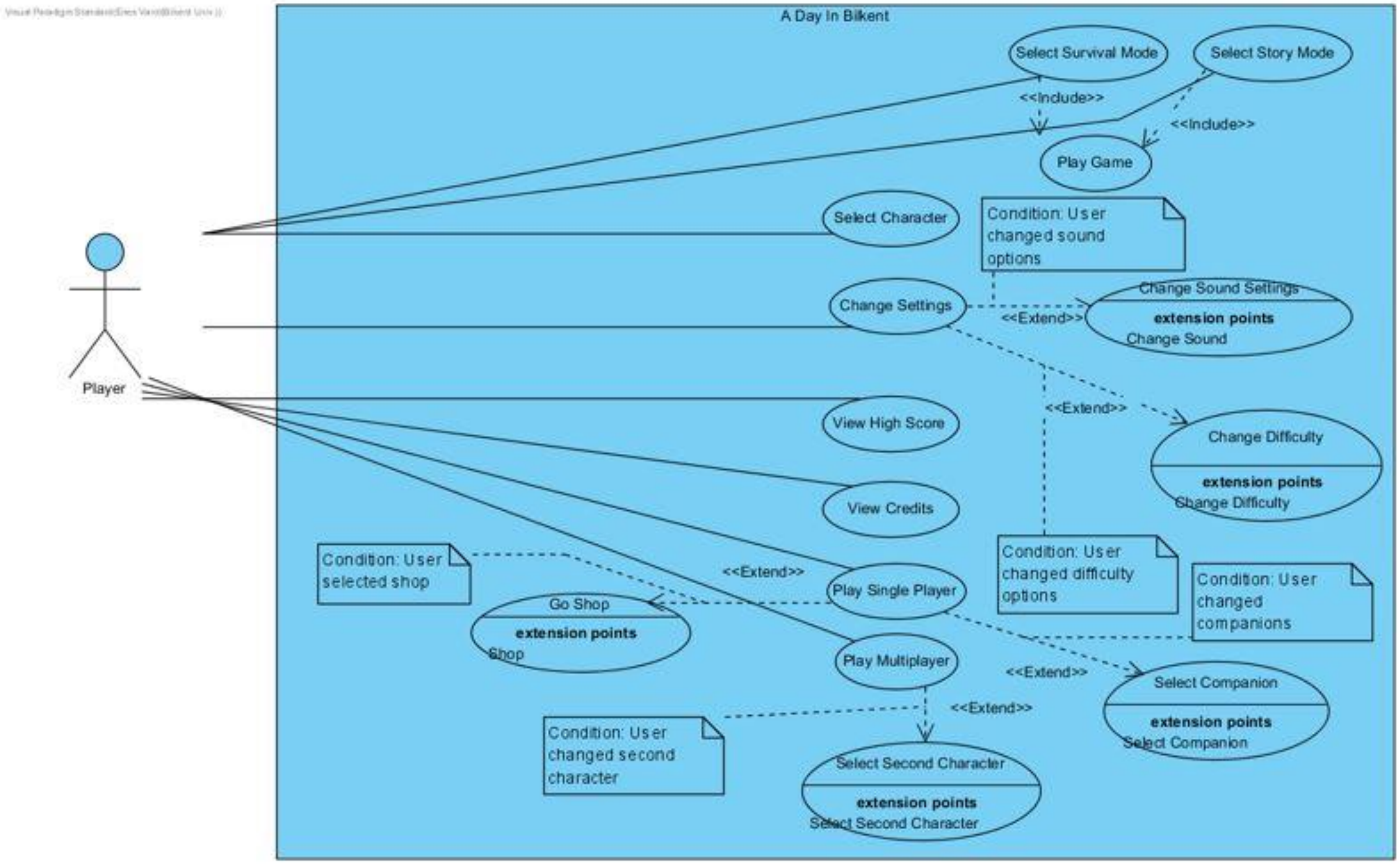


Figure 1

After first iteration, use case of the multiplayer became to *Figure-1*.

Activity Diagram of Multiplayer

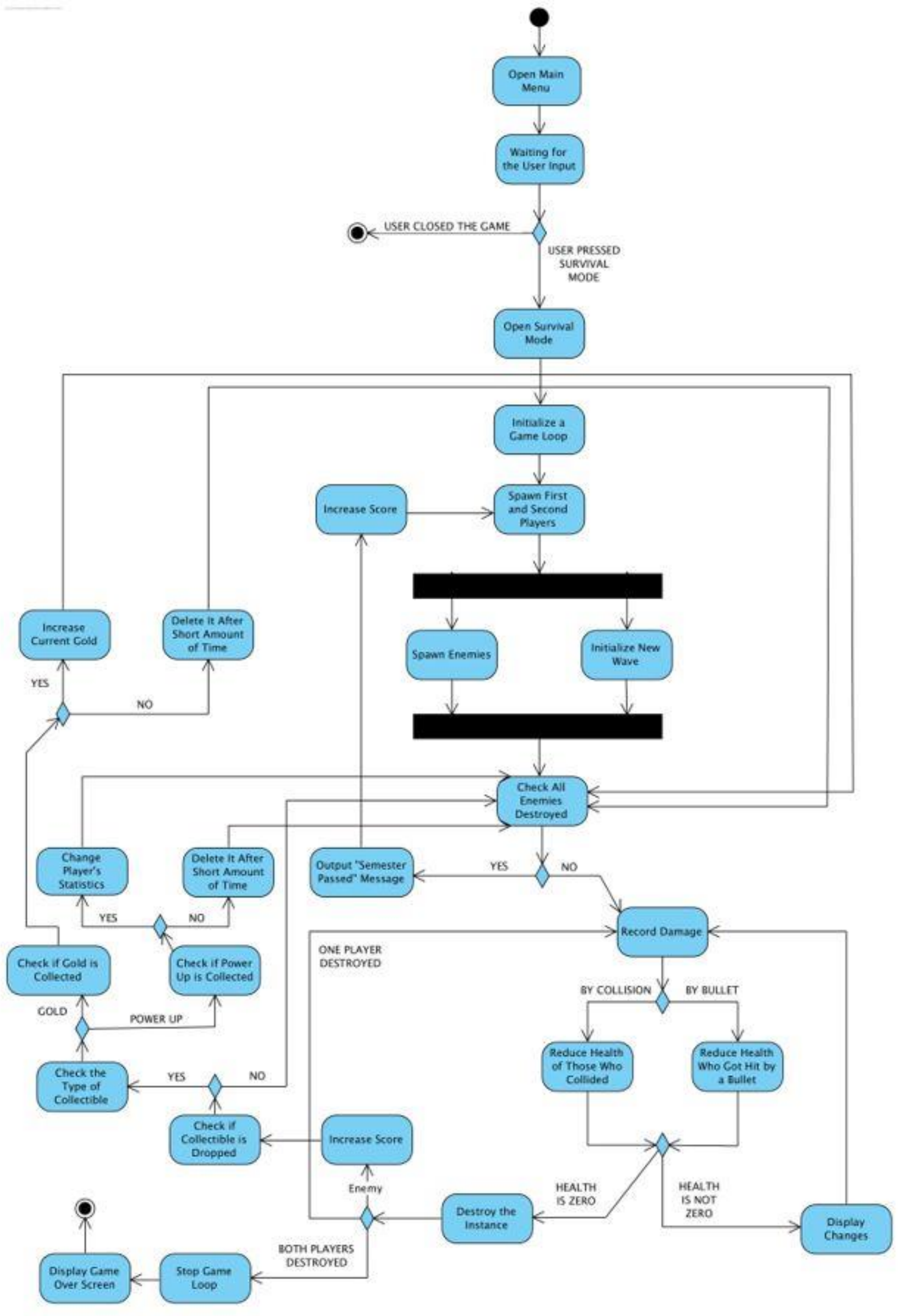

Open Main Menu
Waiting for the User Input
USER CLOSED THE GAME
USER PRESSED SURVIVAL MODE
Open Survival Mode
Initialize a Game Loop
Increase Score
Spawn First and Second Players
Spawn Enemies
Initialize New Wave
Increase Current Gold
Delete It After Short Amount of Time
YES
NO
Check All Enemies Destroyed
Change Player's Statistics
Delete It After Short Amount of Time
Output "Semester Passed" Message
YES
NO
YES
NO
Record Damage
ONE PLAYER DESTROYED
Check if Gold is Collected
Check if Power Up is Collected
GOLD
POWER UP
BY COLLISION
BY BULLET
Check the Type of Collectible
YES
NO
Reduce Health of Those Who Collided
Reduce Health Who Got Hit by a Bullet
Check if Collectible is Dropped
Increase Score
Enemy
HEALTH IS ZERO
HEALTH IS NOT ZERO
Destroy the Instance
Display Changes
BOTH PLAYERS DESTROYED
Display Game Over Screen
Stop Game Loop

Figure 2

After first iteration, activity diagram of the multiplayer changed to *Figure-2*.

Subsystem Decomposition

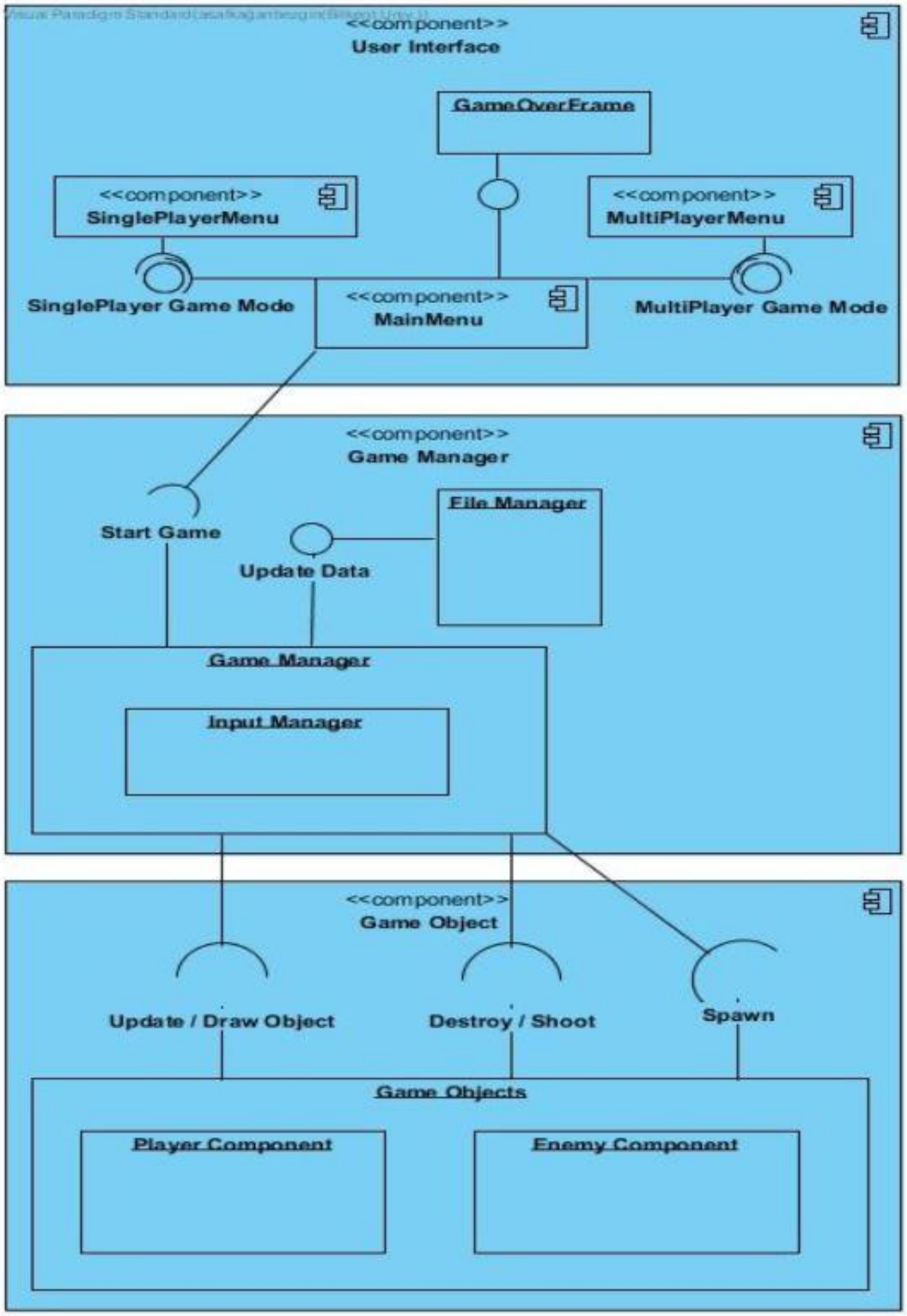


Figure 3

After first iteration, subsystem decomposition changed to *Figure-3*.

Class Diagram

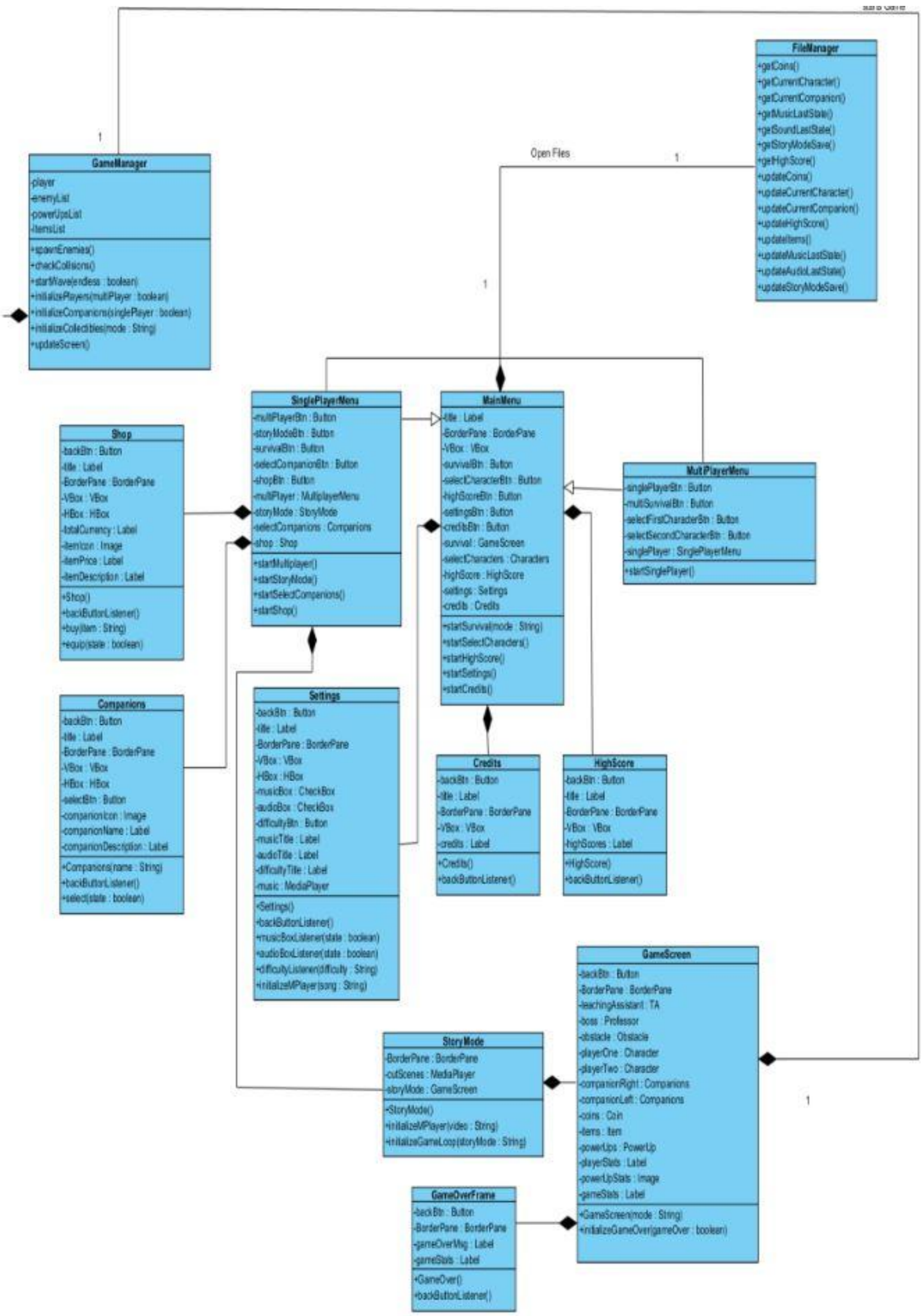


Figure 4

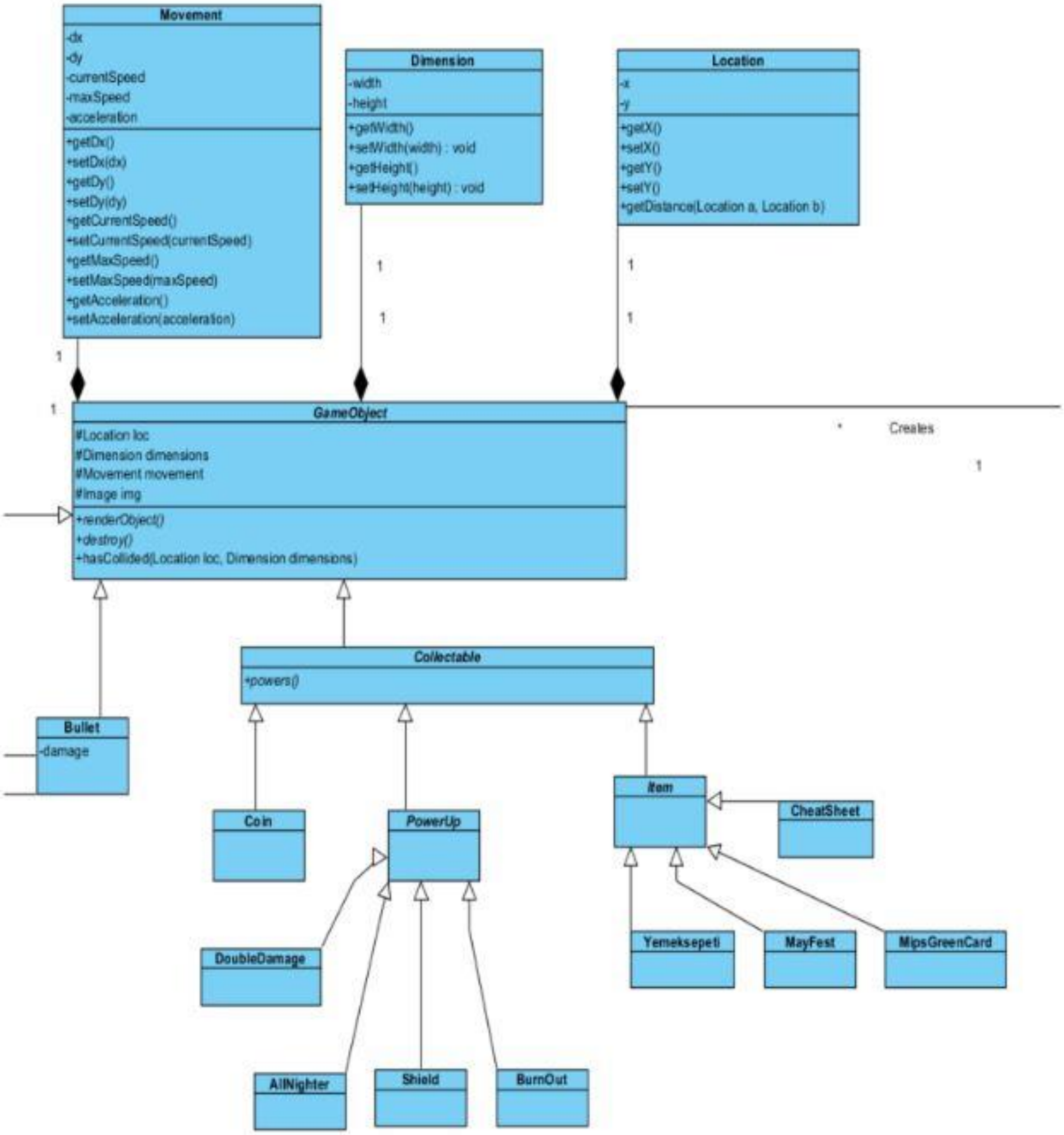

Movement
-dx
-dy
-currentSpeed
-maxSpeed
-acceleration
+getDx()
+setDx(dx)
+getDy()
+setDy(dy)
+getCurrentSpeed()
+setCurrentSpeed(currentSpeed)
+getMaxSpeed()
+setMaxSpeed(maxSpeed)
+getAcceleration()
+setAcceleration(acceleration)
Dimension
-width
-height
+getWidth()
+setWidth(width) : void
+getHeight()
+setHeight(height) : void
Location
-x
-y
+getX()
+setX()
+getY()
+setY()
+getDistance(Location a, Location b)
1
1
1
1
1
1
GameObject
#Location loc
#Dimension dimensions
#Movement movement
#Image img
+renderObject()
+destroy()
+hasCollided(Location loc, Dimension dimensions)
*
Creates
1
Collectable
+powers()
Bullet
-damage
Coin
PowerUp
Item
CheatSheet
DoubleDamage
Yemeksepeti
MayFest
MipsGreenCard
AllNighter
Shield
BurnOut

Figure 5

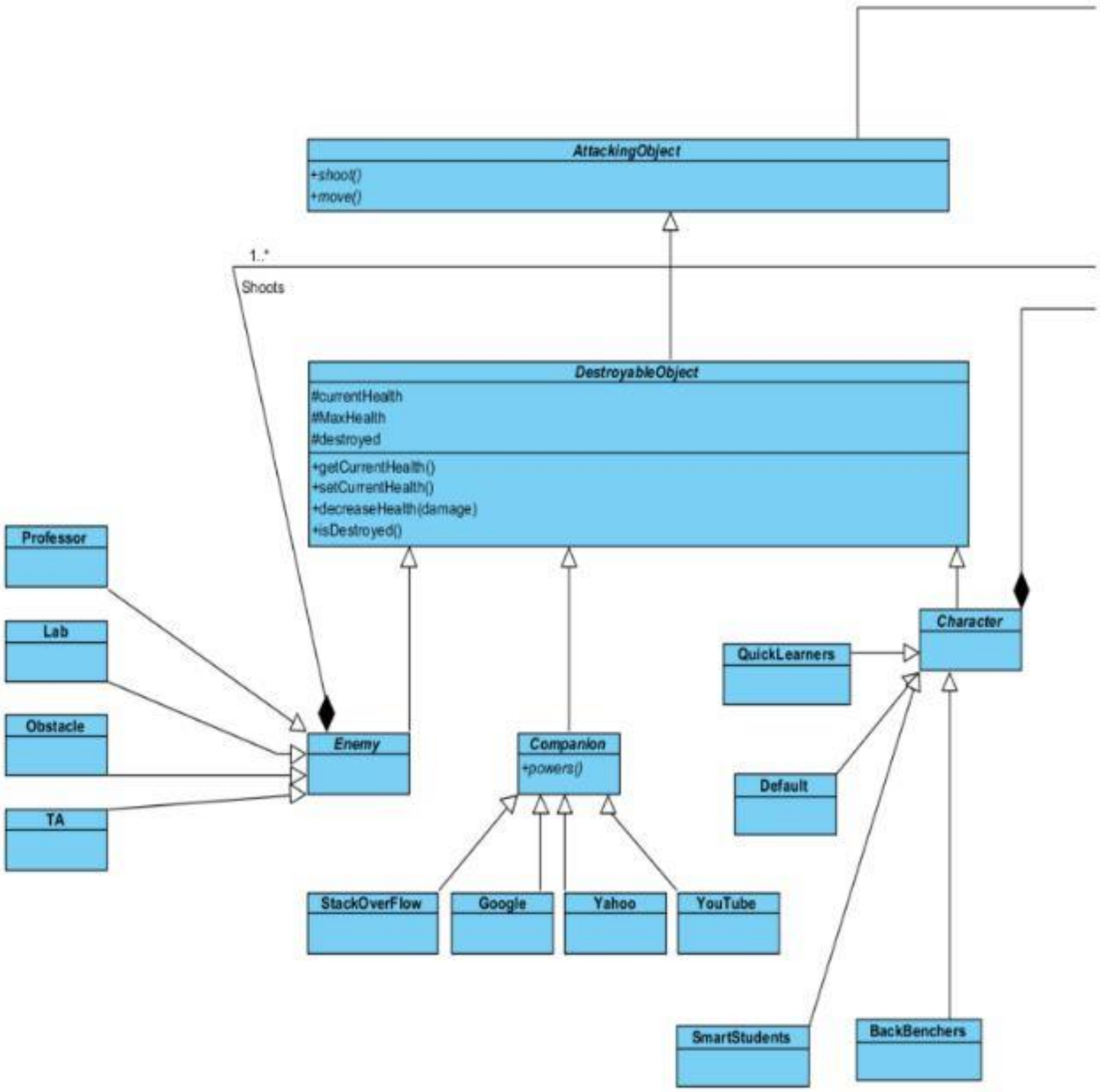


Figure 6

Figure-4, *Figure-5* & *Figure-6* shows latest version of the class diagram which is changed after first iteration.

Both diagrams illustrate the changes made during the implementation process when they are compared to older versions. Diagrams are shown going from high-level design to low-level design. Low level diagrams are re-arranged according to MVC design architecture. The other reason of changing diagrams is implementing the promised game features. When both situations combined, the need of new design and new diagrams occurred. Detailed versions are included in the last versions of the Analysis & Design reports.

3. IMPROVEMENTS

During the implementation process some extra features are added to the game as an improvement to the graphics.

3.1. CUTSCENES

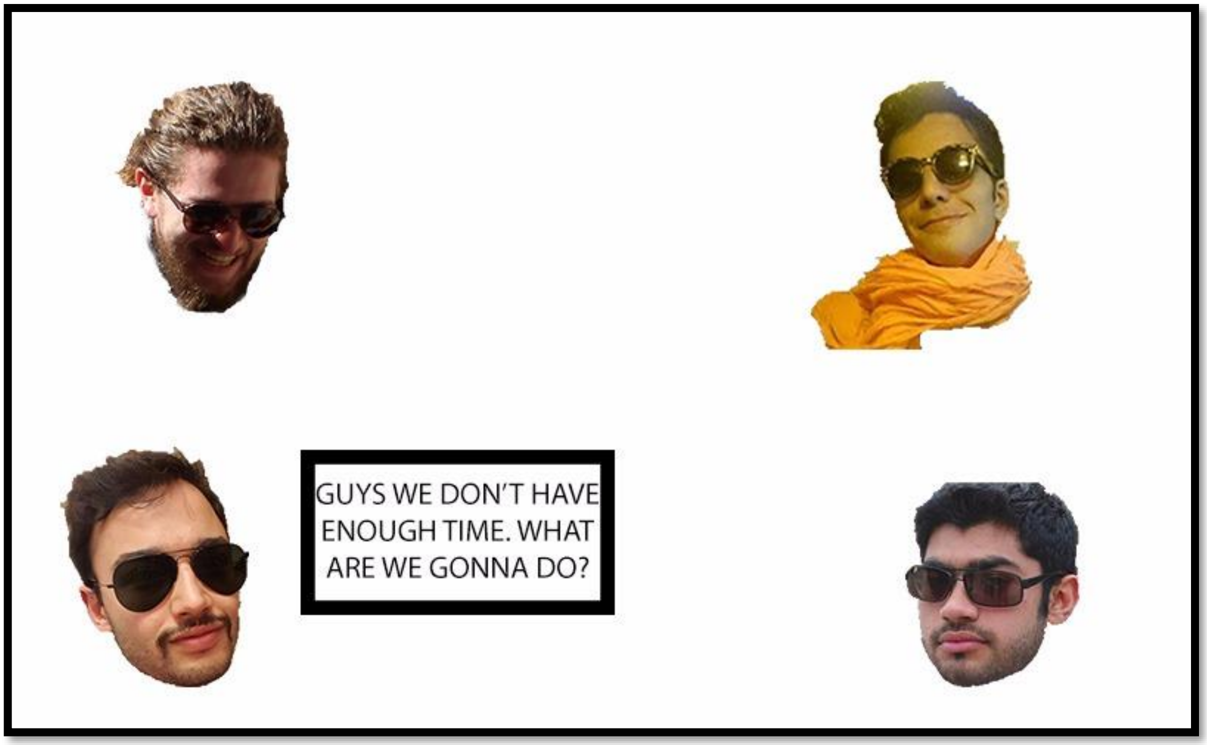


Figure 7

Figure 8

When story mode is selected, the user saws multiple cutscenes in order to make the gameplay a story-based experience. *Figure-7* & *Figure-8* are screenshots from the game. Pictures are cropped with photo shop from original photos, frames and transitions are prepared with Sony Vegas. Cutscenes are prepared by group members. No outside collaboration.

3.2. ANIMATIONS

Animations are included as an extra feature in order to let the user have a better gaming experience.

4. LESSONS LEARNT

Since the project is finished, A-Day-In-Bilkent group take lessons from this experience. First of all, communication is essential for the success especially in this kind of projects. Our group is the most crowded one when compared to the other groups this semester and being on the same page constantly is very important. Therefore, communication plays a key role. The communication among the group members take place first on WhatsApp group chat. Since WhatsApp is the most commonly used communication platform in the world we prefer to use it. Also, because it is very handy and comfortable, it made our days easier and definitely recommended in this kind of projects. However, every single group member of A-Day-In-Bilkent believe the importance of face to face conversations. Therefore, every single group member strictly followed the meeting schedule. Because we met regularly with discipline, we easily handle the communication aspect and it was our strongest side. Whenever we lost the communication, we made mistakes. Therefore, communication is the mot important aspect in a team work.

Being disciplined is not only about being on time but it also about being aware of responsibilities and fulfilling them. We as a group, divided the project in to parts and distributed it among the group members in every step of the project. Since everyone fulfill their own responsibilities, we finished the project on time and add almost every feature promised in the beginning. Distributing the parts according to the group member's skills is also very important. Instead of assigning works which are not the strong side of a member to that member is not efficient since he or she needs to learn it before doing it. Therefore, everyone should be assigned with the parts that they feel they are capable of doing it.

Time management is the last concern in a team work. Managing the time by implementing certain parts at certain time periods is very important. Otherwise, trying to handle everything all at once might cause problems which we face during the project rarely.

5. USER'S GUIDE

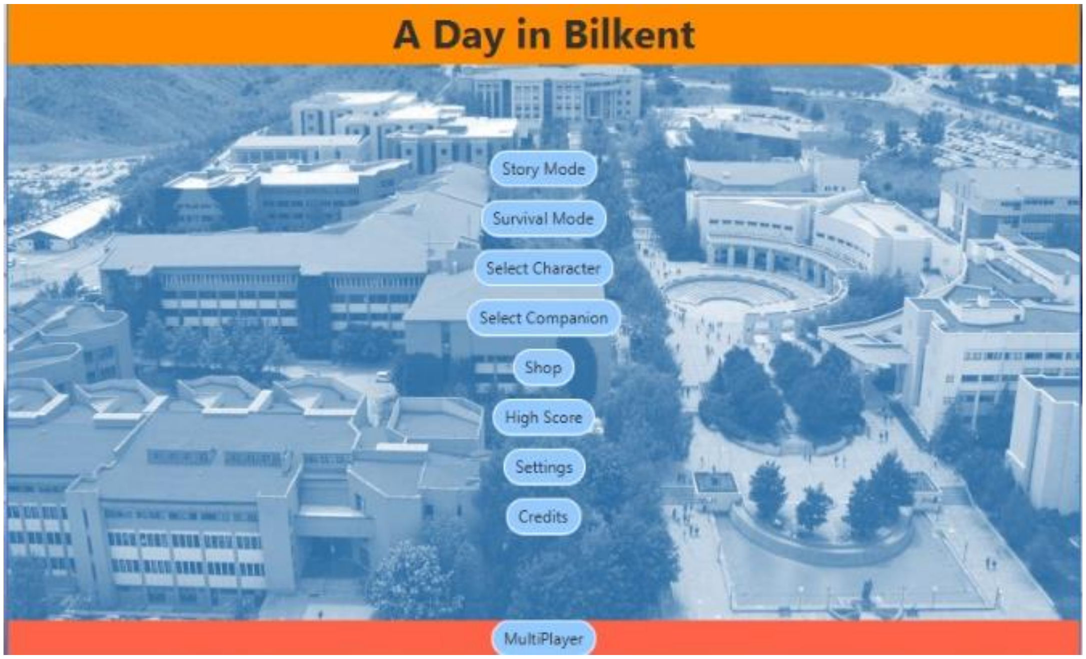
5.1. SYSTEM REQUIREMENTS & INSTALLATION

A-Day-In-Bilkent require JVM (Java Virtual Machine) and Java SDK. More detailed:

- Java 8
- Intel Graphics
- 4 GB Ram
- 500 MB free disk space

For the installation, go to https://github.com/askari12/2I.A-day-in-BIlkent and download the source folder. To run the game, a Java compiler is needed (IntelliJ, Eclipse, etc.). Running the code downloaded from GitHub source folder in a Java compiler will run the game.

5.2. HOW TO USE



This is the main screen of the game. Story-Mode is a story-based experience which is finite.

Survival mode is a wave-based game mode.

Select Character is for selecting in game character

Select Companion is for selecting companions for certain game modes.

Shop is for buying items which are useable in certain game modes.

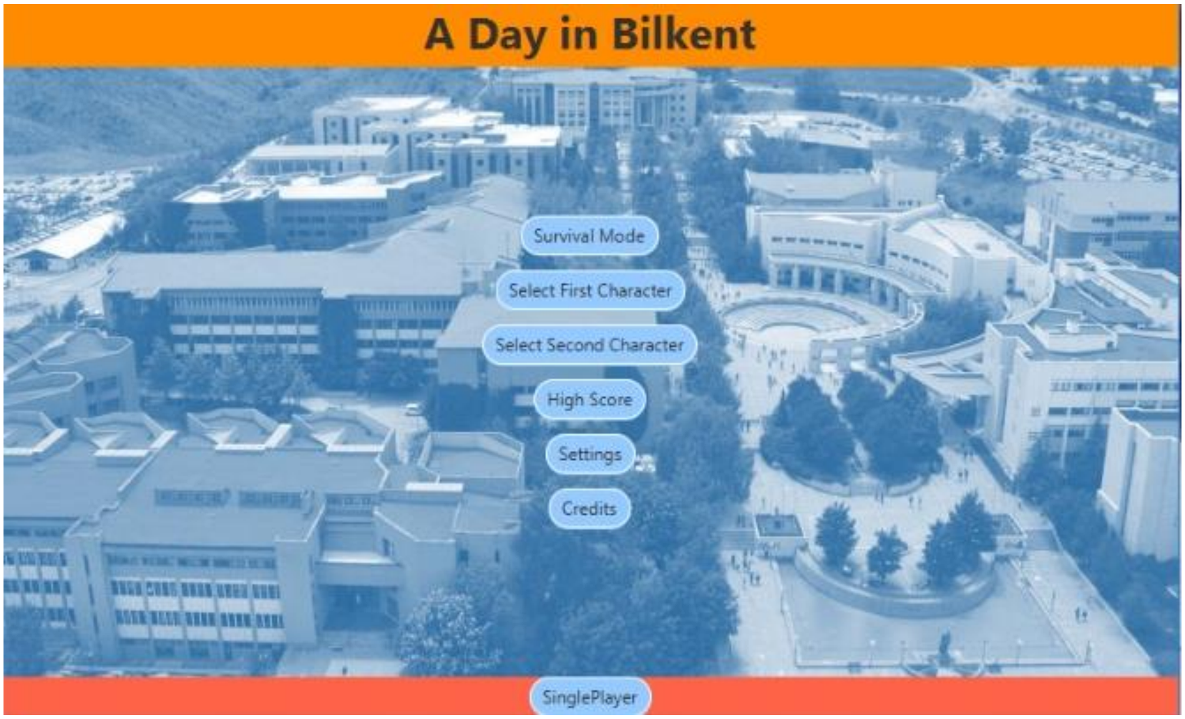
High score displays the high score.

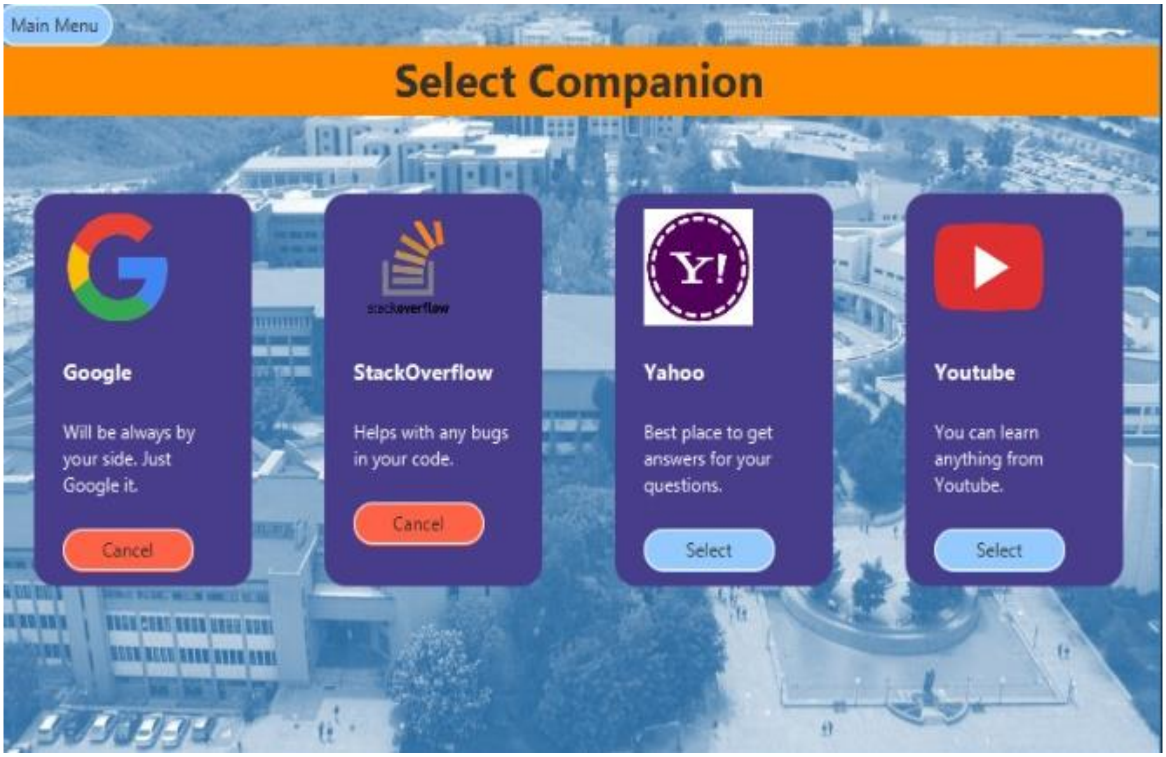
Setting is for adjusting the sounds.

Credits display the writers.

Multiplayer is a local multiplayer which allows two players to play in the same game at the same time on the same computer.

Some screenshots from the game:





Select Companion

Google

Will be always by your side. Just Google it.

StackOverflow

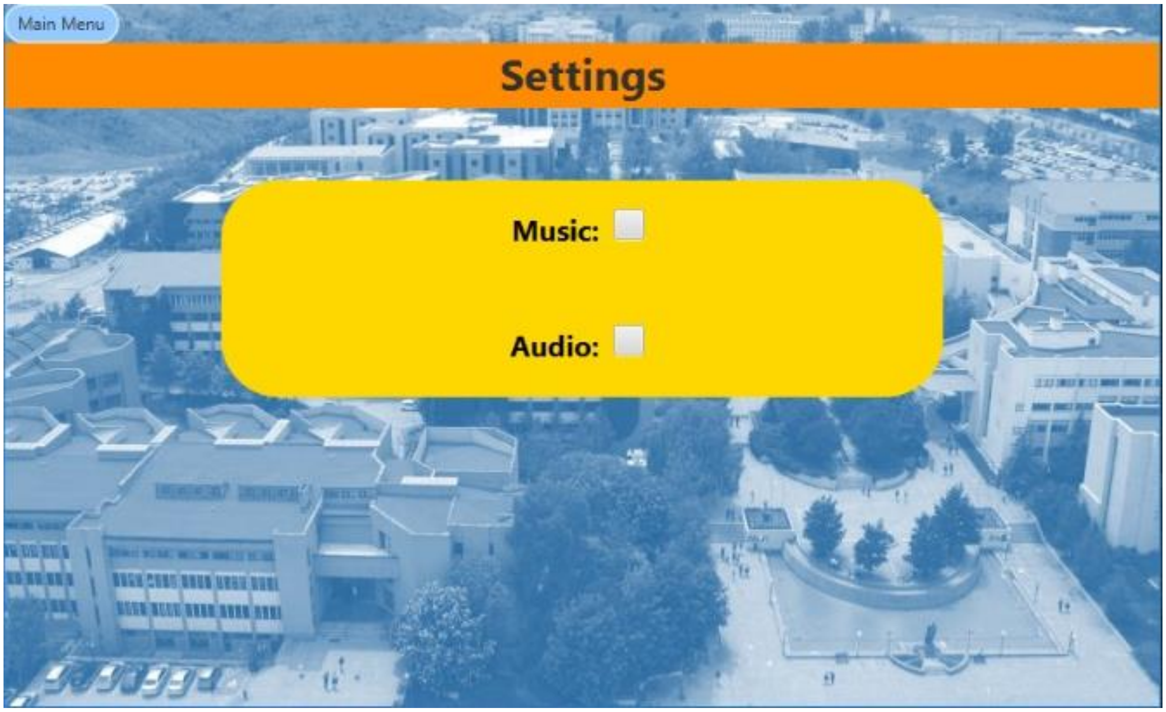
Helps with any bugs in your code.

Yahoo

Best place to get answers for your questions.

Youtube

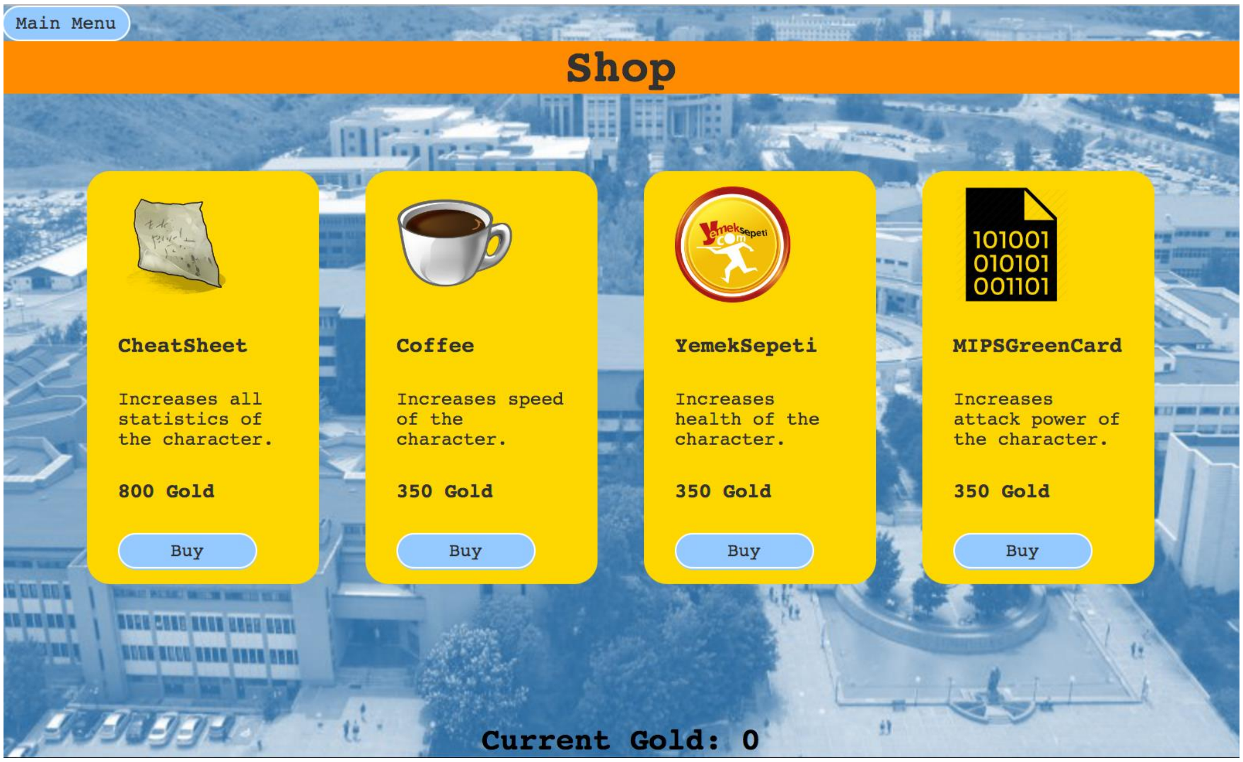
You can learn anything from Youtube.



Settings

Music: □

Audio: □



Shop

CheatSheet

Increases all statistics of the character.

800 Gold

Coffee

Increases speed of the character.

350 Gold

YemekSepeti

Increases health of the character.

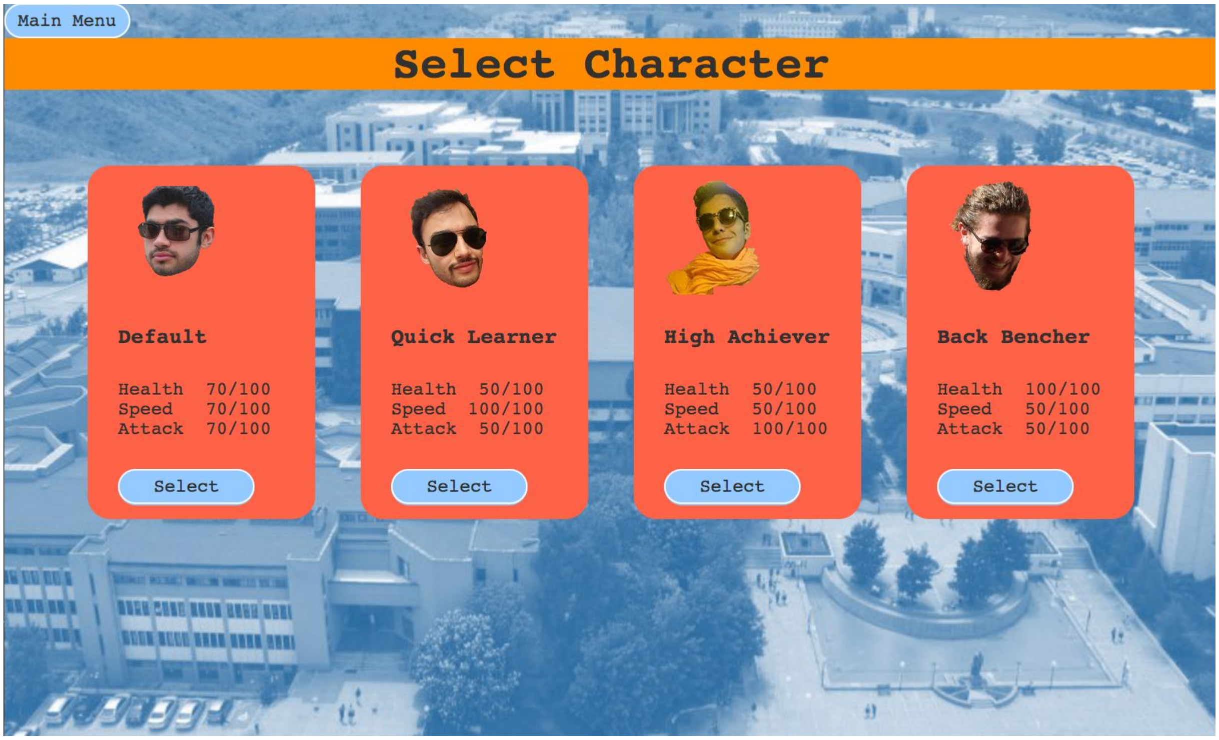
350 Gold

MIPSGreenCard

Increases attack power of the character.

350 Gold

Current Gold: 0



Select Character

Default

Health 70/100
Speed 70/100
Attack 70/100

Quick Learner

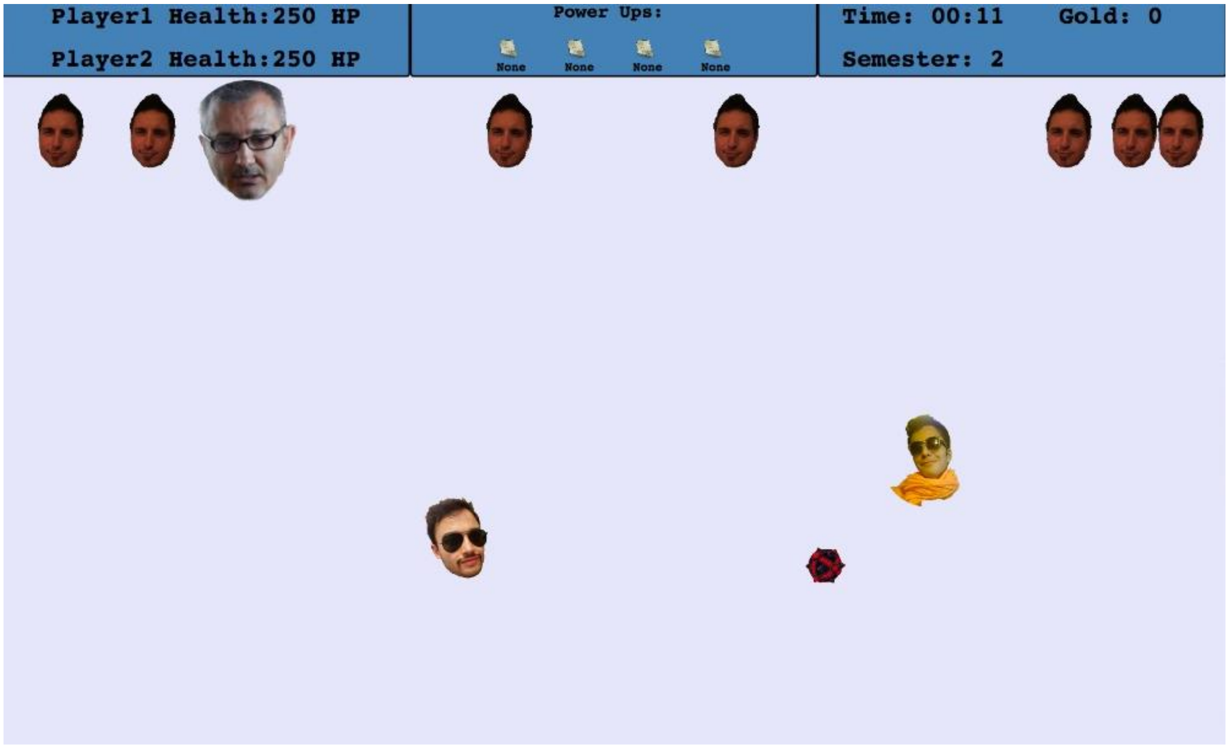
Health 50/100
Speed 100/100
Attack 50/100

High Achiever

Health 50/100
Speed 50/100
Attack 100/100

Back Bencher

Health 100/100
Speed 50/100
Attack 50/100



Player1 Health:250 HP

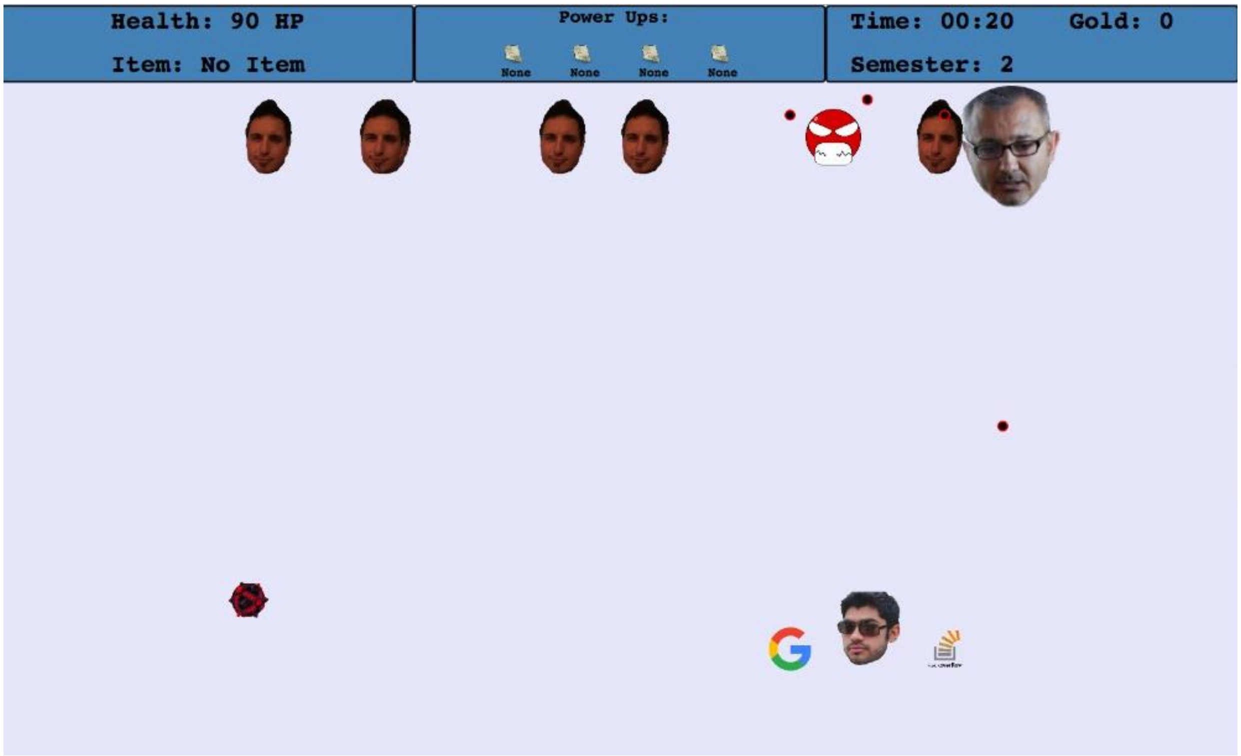
Player2 Health:250 HP

Power Ups:

None None None None

Time: 00:11 Gold: 0

Semester: 2



Health: 90 HP

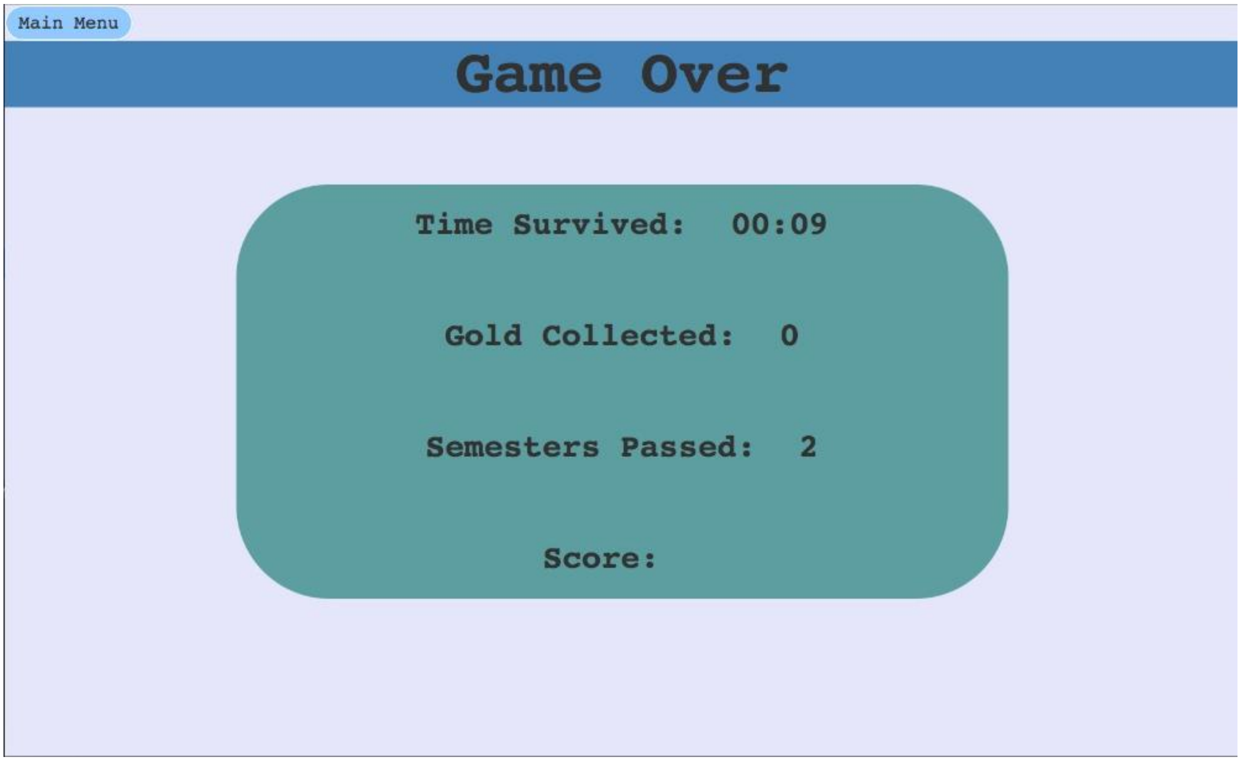
Item: No Item

Power Ups:

None None None None

Time: 00:20 Gold: 0

Semester: 2



Game Over

Time Survived: 00:09

Gold Collected: 0

Semesters Passed: 2

Score:

6. WHAT IS MISSING

- No special effects for companions.
- Errors occurred during the merge process of the code.