ANIME SCHOOL TEACHER

Final Year Project

Session 2018-2022

A project submitted in partial fulfillment of the degree of

BS in Computer Science

Department of Computer Science

Faculty of Computer Science & Information Technology

Superior University, Lahore

Spring 2022

	[✓] **D**evelopment [ ] **R**esearch [ ] **R&D**			
Area of specialization	GAME DEVELOPMENT			
FYP ID	FYP-BCSM-F21-034			
Project Group Members				
Sr.#	**Reg. #**	**Student Name**	**Email ID**	***Signature**
(i)	BCSM-F18-426	Abu Bakar	Bcsm-f18-426@superior.edu.pk	
(ii)	BCSM-F18-437	Rehman Ali	Bcsm-f18-437@superior.edu.pk	
(iii)	BCSM-F18-420	Ayaz Sarwar	Bcsm-f18-420@superior.edu.pk	

*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

Plagiarism Free Certificate

This is to certify that, I am Abu bakar S/D of Muhammad Saleem, group leader of FYP under registration no FYP-BCSM-F21-034 at Computer Science Department, The Superior University, Lahore. I declare that my FYP report is checked by my supervisor.

Date: Name of Group Leader: Abu bakar Signature: Abu bakar

Name of Supervisor: Shahzad Nemat Malik

Designation: Lecturer

Signature: _____

HoD: Dr. Irfan Ud din Signature: _____

Project Report

ANIME SCHOOL TEACHER

Change Record

Author(s)	Version	Date	Notes	Supervisor's Signature
	1.0		<Original Draft>	
			<Changes Based on Feedback from Supervisor>	
			<Changes Based on Feedback From Faculty>	
			<Added Project Plan>	
			<Changes Based on Feedback from Supervisor>	

APPROVAL

PROJECT SUPERVISOR

Comments: _____

_____

Name:_____

Date:_____ Signature:_____

PROJECT MANAGER

Comments: _____

_____

Date:_____ Signature:_____

HEAD OF THE DEPARTMENT

Comments: _____

_____

Date:_____ Signature:_____

Dedication

This project is whole-heartedly dedicated to our parents, and teachers who have been our source of inspiration and give us strength when we thought of giving up, who continually provide their moral, spiritual, emotional, and financial support. To our friends, mentor, and classmates who shared their words of advice and encouragement to finish this project.

Acknowledgements

We are three students of Computer Science Department in Superior University Lahore, preparing an FYP report on “ANIME SCHOOL TEACHER”. Primarily we would thank ALLAH for being able to complete this report with success. Then we would like to thank our Supervisor “Mr.Shahzad Nemat Malik” whose valuable have been the ones that helped us patch this report and make it full proof of success his suggestions and his instruction has served as the major contributor towards the completion of the report. Then we like to thank our friends who have helped us with their valuable suggestions and guidance has been helpful in various phases of the completion of the report.

Table of Contents

Chapter 1
Introduction

Chapter 1:
Introduction of the Game

Anime School Teacher Simulator is developed as a bachelor's thesis documentation for the department of computer science at Superior University Lahore. The aim of this documentation is to show the whole concept along with development processes, implementation design, and structure as well as its functionalities (functional and non-functional) and others. In this project, Unity3D Game Engine is used to develop this action based shooting game.
Nowadays digital games are solving queries where children cannot involve in physical activities because they do little to do so, space limitations, and dangers associated with these activities. These problems can be solved by providing them 3d games like Third person games,

1.1 Background:

Before starting anything we must know the background of that specific niche. If we found something has already been developed then we can start working on its next upgraded module. By knowing the background we save our lot of time by just producing existing results. In our project, it's a single-player strategy-based game where game play will be dependent on the logical thinking and planning. Players should have a sharp mindset to overcome other players' actions. Good technique usually requires playing and grasping the hand movements of players.

1.2 Motivation and Challenges:

The Biggest motivation behind choosing the game development niche was a keen interest to learn about new technologies. Now the world is going toward digital and virtual things so it was the biggest motivation part for me. The biggest motivation is for us to build an audience who is going to play our game and after reading their reviews to do improvements in our game is the biggest challenge for us. Addictive game play and attractive graphics are always a challenging part of game development.

1.2.1 Addictive Game play:

The long-term sustainability of the game is measured by its addictive game play. The game result must impress new joiners of the game because it will be helpful to increase the impression rate of the game. In this way, we players will be interested to come again and again.

1.2.2 Fascinating Graphics:

Most players' measure game quality by good graphics and it is quite true because the first thing that we notice in any game is its graphics. If graphics are ultra-level then definitely attracts more users. So we came to a conclusion and decided to use unity for our fps game project because of its good community support and it provides a lot of inbuilt plug-in so we don't need to work from scratch like unity has its own physics and animation components.

Good graphics play an important role in any project so unity provides a superb graphics system by doing little improvement so we can achieve good graphics in unity. The game contains graphics made in both Blender and adobe Photoshop.

1.2.3 Single player Functionality:

The biggest challenge is to introduce Single features in our game by using a photon cloud and its APIs. Also, it can be played offline in a single-player mode. This is the major feature of the game that seems to handle difficultly as none of them have the experience to make such a game.

Our plan is to create a lobby system for our single player game in which players can create new rooms and can join existing rooms to play a game.

1.3 Goals and Objectives:

Our goal is to design and develop an interactive single player game for the age group of 3 to 18. This game combines the elements of traditional games and elements of current successful games available in a market.

Our main goal is to make game graphics and game play so unique so if a user comes to our game, the player should play for maximum time. Our main menu screen will play with the mindset of players when they see the initial buttons screen so the UI of the button should be so unique so the user wants to click that and interest will be shown by the user to discover more areas of the game.

1.4 Solution and Overview:

At the start of the game, a name panel will appear where the user will insert his/her name which will be saved into the game database. The next screen will be the main menu where users see buttons of single player from where the user will go to the next screen. After the main menu, the next screen will be a selection menu where users can select their favorite weapons to use in the game. These weapons can be upgraded by spending digital currency like in-game dollars or coins.

When the user will select the single-player button the game modes menu will appear, there will be a level selection mode where the user will select the desired level, and as hits the play button the offline game will start. During game play whenever a player will kill any enemy, points will be given to the player. Similarly, after pressing the multiplayer button, users will be able to create a lobby and can join other lobbies to play with their friends or random players.

Chapter 2
Literature

Chapter 2:
Literature

2.1 Introduction of Literature:

Anime School Teacher Simulator games are getting famous because of ever-growing easy to do actions in Android. This is all due to the smart capabilities of the Smartphone. The physics behavior of bullets and weapons gives the player a real virtual experience. This virtual experience gives a feeling that we are using our favorite weapon and fighting with our enemies and that too on our mobile devices. 2015 and 2016 have been the year of virtual reality and games are enjoying the perks of it. The effects and graphics of the games today are way beyond our imagination. The features are amazing and keep updating according to the need of the game.

2.2 Technologies Overview:

Some of the best games developed in the past that inspire our project, we are elaborating their features and what makes them successful.

2.2.1 Anime School Teacher Simulator:

Anime High School Teacher Simulator to regulate your schoolroom by obtaining quiz take a look at. your anime faculty teacher and obtain excellent grades within the quiz take a look at that are conducted throughout the category at school life days. The virtual high school teacher anime girl games gets up early in the morning prepared for top school during this anime high school games. Multiple quizzes takes a look at what you have got to conduct from the anime faculty students and provides the mark per the paper try during this virtual high school teacher anime girl games.

The high school teacher reaches school at the time and obtains the category frequently. The teacher then goes to his employees' area and meets together with his colleagues in Anime School Teacher Simulator - Be the simplest student and obtain nice marks or grade A+. The virtual teacher starts the category attending of the varsity students during this virtual high

school teacher anime girl games. Get the daily quiz take a look at and learn plenty to reinforce your information and choose the proper answer during this Anime School Teacher Simulator. The anime faculty teacher began to do the introduction of the scholars within the educational institution teacher game. Afterward, the virtual teacher starts the teachings within the schoolroom. wonderful atmosphere journey alongside best faculty life days fun during this Anime school Teacher Simulator.

2.2.2 Anime School Teacher Simulation (Mobile) :

In this virtual anime high school teacher anime girl games, there are a lot of teacher simulator virtual activities to perform. Like some students are missing from the classroom and they are playing games in the high school ground instead of studying in virtual high school teacher anime games. So find them and give some punishment to counsel them being a responsible school teacher in High School girls simulator crazy teacher games. Now give lecture to the class students and solve their study-related problems in this anime school teacher simulator 2021. Now announce the surprise quiz for class students and make sure all the students participated. There are some students who are cheating on the quiz so find them and give punishment in this teaching life school game. So there is a lot of fun and joy of teacher role in this anime high school game.

2.3 Summary:

In this Anime School Scari Teacher Simulator game, there are plenty of teacher machine virtual activities to perform. Like some students are missing from the schoolroom and that they are enjoying games within the high school ground. Currently provide lectures to the category students and solve their study-related issues during this anime faculty scari teacher teacher simulator. Therefore there's plenty of fun and joy of scari teacher teacher role during this anime high school game.

Chapter 3 Requirement Analysis

Chapter 3:
Requirement Analysis

3.1 Introduction:

Requirements Analysis is the process of defining the expectations of the users for an application that is to be built or modified. It involves all the tasks that are conducted to identify the needs of different stakeholders. Therefore requirements analysis means to analyze, document, validate and manage software or system requirements.

Quality Function Deployment is a technique that translates the needs of the customer into technical requirements for software/games. It concentrates on maximizing customer satisfaction from the Software engineering process.

3.1.1 Normal Requirement:

Modules and Functions of the project usually discussed with the relevant people in the meeting, all these come under the Normal Requirement.

Normal requirements of our project are:-

1. System should be user-friendly.
2. Maintenance costs should be low as much as possible.
3. Devices should be available in the market on which project will run.
4. Project should be easy to start and understandable.
5. Professional environment should be provided for the project.
6. Expected completion target will be set by measuring coding and designing.

3.1.2 Expected Requirement:

These requirements are implicit to the system and may be so fundamental that the actor/gamer/ relevant people do not explicitly state them. Their absence will be a cause for dissatisfaction.

Expected Requirements are those which are not described while briefing but if we miss those it can be cause of dissatisfaction for them.

1. Cost should be minimal.

2. Quality should be better than previously developed games.

3. Game should run on low-end devices also with the same performance.

4. Lag should be minimum in game.

5. Ad position should be at the right place so it could not irritate the user.

3.1.3 Exciting Requirement:

Requirements that customer usually doesn't demand but if we add additionally it shows your professionalism:

1. Game cheat codes can be provided additionally for excitement.

2. Extra Rewards in game for users.

3. Players compete with each other.

4. Update of game with new features.

3.1.4 Software and Hardware Requirement:

Here are the software as well as a hardware requirement for this project to be work

- (a) Software
  - 3d Modeling Software's
  - Adobe Photoshop
  - Unity 3D (game engine)
- (b) Hardware
- Game will be playable on all android devices and tablets.
- It should not be dependent on the screen resolution of each device

3.3 Functional Requirements:

Functional requirements are those that we have to implement in our game project.

- Single Player.
- Target Android-based devices.
- 3D platform.
- Multiple levels.
- Action-based.
- 2d GUI and menu system.
- Testing for removing all the bugs in-game.
- Written in C sharp for unity.
- Photon cloud.
- Models.
- Upgrading and Purchasing game currency.
- Graphical ejects.
- Sounds and music.
- Time integration.

3.3.1 Single modes:

The game is based on single. Single-player is for offline and the multiplayer feature is added when the user wants to play online with others throughout the world.

3.3.2 Android-based:

The game is for android cell phones. We decided to make a game for android the reason that nowadays android phones are going to be very popular.

3.3.3 3D platform :

Our game is based on the 3d platform rather than 2d. But we implement buttons and game cover in 2d.

3.3.6 2d GUI and menu system :

The overall game will be in 3d but the GUI menu and button are in the 2D platform. 3d buttons are non-functional requirements.

3.3.7 Testing:

The game will be tested in the proper way until every bug is removed. Bugs in the game should be zero.

3.3.8 Written in C sharp:

The game is written in c-sharp code and is integrated into the unity 3d platform. Unity also supports other languages like boo and JavaScript but now these are depreciated from unity.

3.3.9 Photon cloud API:

We will use Photon cloud API's to integrate 3rd person features in our game. Initially, we decided to use Unity's own cloud but later we decided to use photon cloud to connect the server.

3.3.10 Models:

3d models are made in a blender. As blender is easily integrated into unity 3d game engine.

3.3.11 Textures:

The model's textures were satisfactory, although they were not always particularly uniform.

3.3.12 Variety of Level:

Different varieties of weapons will be offered into the selection menu where users can buy or upgrade different kinds of Stages to improve their gaming experience.

3.3.13 Visual Ejects:

Visual Effect enhances game play graphics and gives the overall game a fascinating experience through which users stick to the game for more time. Unity has a built-in component called a Particle system which is used to make visual effects,

3.3.14 Sounds Effects and Background Music :

Our plan is to use good background music because it suits the action type of game. Sound effects like bullet, walking, and button clicking effects are popular ones.

3.3 Non- Functional Requirements:

Non-functional requirements are those that are not necessary to be implemented in the project if time is found then they may be added to the project later.

Following are the non-functional requirements of our project.

- IOS build.
- Multi-level
- Creation of 3D models
- Using unity cloud
- Battle Royale System.
- In-app purchases.
- Google play services

Chapter 4
Project Design

4.1 Introduction

Adaptation of methodology in Game Development is necessary as it teaches us by which techniques our project should go next. By keeping this in mind we have to select the right process model for our project because success of the project will be dependent on the right process model. We have to see a few things before selecting any process model.

- Estimated Time and Resources for our current Project.
- Can we expect other requirements to be added later on.

4.1.1 Development Process:

The purpose of the development process is to ensure that our product will fulfill specific qualities such as product should be able to handle Robustness, usability and dependability etc. To have these qualities we have to choose the right development software model which provides proper strategies to handle and gain these qualities.

4.1.2 Software Process Model:

There are so many software process models which companies use to make the quality product. With these development life cycles, our process of designing, developing and testing any product goes smoothly at every stage.

4.1.3 Software Development Life Cycle :

Popular Game Studios go through SDLC for making game projects. By keeping in mind the steps of SDLC. we develop, maintain and alter qualities of specific projects. There are certain methods that are followed during the process of following it. Popular stages are:

- **Analysis of Requirement :**

In SDLC, the most vital phase is analysis of requirements. First of all, the requirements of the project are analyzed. The reason is to judge the economical, technical and risk factors. If we calculate these things at start then the success ratio of the project increases with good numbers.

- **Planning :**

Once we are done with the analysis of requirements then further we have to be part of the product and document explanation where we will mention all requirements and details about our project.

- **Modeling or Product Design:**

The Final design is being selected at this stage Now selected design will represent the complete architecture and flow of communication which we will be using later on in our project.

- **Implementation:**

Now it is the stage where our actual project will be implemented according to all proposed requirements from previous stages. The code guide lines will be followed by project developers. According to our project, a suitable programming language will be chosen for the project.

- **Product Testing:**

In SDLC, products are done in all stages but specifically at this stage testers usually identify the bugs and defects in the product and process it continuously until desired results do not come out.

- **Maintenance and Distribution:**

Once the standard and quality of product is according to desire, now the product distribution process continues under the company itself. Customer Value and feedback gathered by company teams which help them to improve the product in the next update.

4.1.4 Incremental Process Methodology:

In an incremental process, we can enhance the product qualities and function with time until all project is implemented. In this process, there are no specific requirements that we need to give at the start of the project, it can be modified with the implementation of the project. All modules are independent of each other, so new functions and modules can be added until we meet the desired requirement.

Importance of Incremental Model:

- When all the partial components are made then by connecting all these we get our final system.
- Priority based a lot in this model, top priority components are handled first then moves to others.

4.1.4 Current Project Model:

After studying the incremental process model, we decided to choose this model for our current Anime School Game. The main reason to choose this model was the size and duration of our project. In a short time, we wanted to add the maximum amount of features that we could add in our game.

After deciding our model, we divided our project into four phases. We assign the first phase for project planning, the second phase for game designing, third phase for game coding and the final phase for game testing.

- **First Phase (Game Planning) :**

Project success depends on the planning, if your planning is right and going in the right direction then there are more chances of your success. So our project plan was clear to give users a good experience and add a maximum amount of features in less time.

- **Second Phase (Designing of Game) :**

Design is the thing which users see first in your project, if your design is interactive, user friendly and has good graphics then there are maximum chances users will not get bored or leave your project early. We, as group members, discussed a lot of ideas but we were not able to decide which design should be picked. So after a lot of conversation, we got an unique idea: why not design our university map where players can move around and can see university buildings. Everyone liked this, and we decided to go with this.

- **Third Phase (Coding of Game) :**

Third Phase was dedicated to coding of game for which we used microsoft Visual Studio for coding c# langage. Unity Engine use c# language and we were quit comfortable with that because we had understanding of c# before.

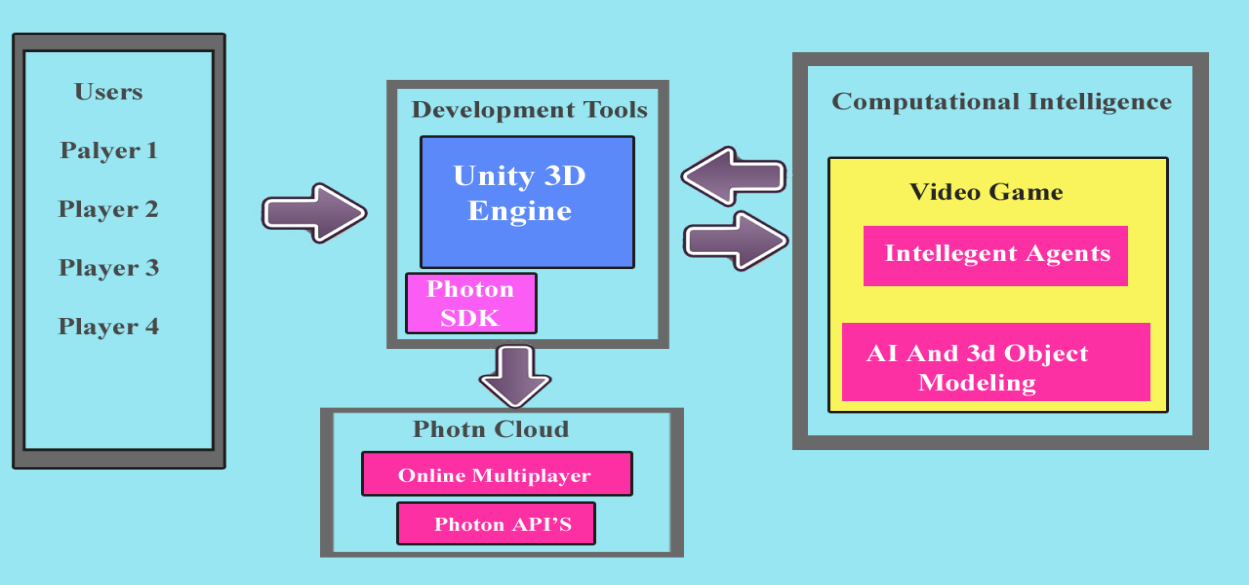
- **Fourth Phase (Testing of Game) :**

Testing was the vital part of our game project because later on we could face a lot of bug issues in our game. We tried to test and debug games many times so we can reduce the risk of bugs on mobile devices. We make a test scenario for testing our game which we have discussed in our next chapters.

4.2 Architectural Design:

Our FPS Shooting Game have four main conceptual frameworks that are demonstrated below:

- Computational Intelligence
- Development Tools
- Photon Cloud



4.2.1 Computational Intelligence:

Our player are using Artificial Intelligence for intelligent behavior.

Intelligent Agents:

Intelligent Agents are those players who act like intelligent people who can identify where our target is, when we should move, which path is short and when to shoot our opposite player. In our online mode, we don't need these agents because on every end there will be real players but in offline mode, we will have these agents on every level which can us if we go close to them or study them.

Artificial Intelligence:

AI is computer based calculation of tasks that human intelligence can perform at any specific point so we implemented some techniques in enemies where their behavior will be like some real human is playing but in reality it will be computer techniques.

whenever ray will hit any obstacle the enemy will start moving into the opposite direction.

4.2.2 Development Tools:

These are the following tools that we are using to build our Anime School Game:

- Unity 3D Game Engine
- Microsoft Visual Studio for C#
- Adobe Illustrator and Photoshop
- Blender for Modeling

4.2.3 Photon Cloud:

Unity 3D Photon Cloud API used to interact with other client side players. Currently this is a free version of photon cloud which allows 20 CCU means 20 users can play at same time . Our plan is to buy premium after uploading on Play store or App store.

4.3 Detailed Design:

As we were working on a multiplayer game, as we were proceeding new unique ideas came to our mind and we started implementing them side by side.

4.3.1 Initial Concept:

So our first idea was to create a single player game where users can play without the internet and by completing each level will get a reward with new guns that can be unlocked. It was quite simple and boring because now users want to play against each other so we moved toward our second concept.

4.3.2 Second Concept:

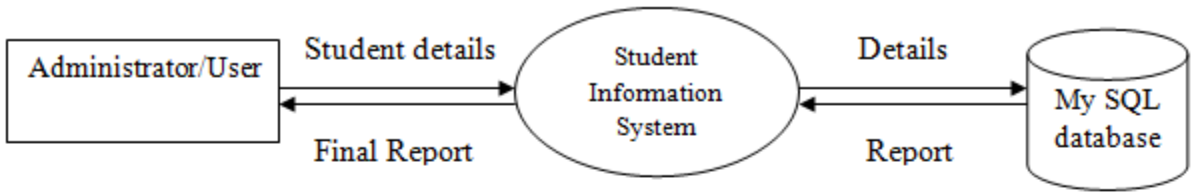
In the second concept, we thought of adding a multiplayer function because the internet is easily accessible nowadays so it can be a good option for our game where players can compete against each other and can play from any place around the world.

4.3.3 Final Concept:

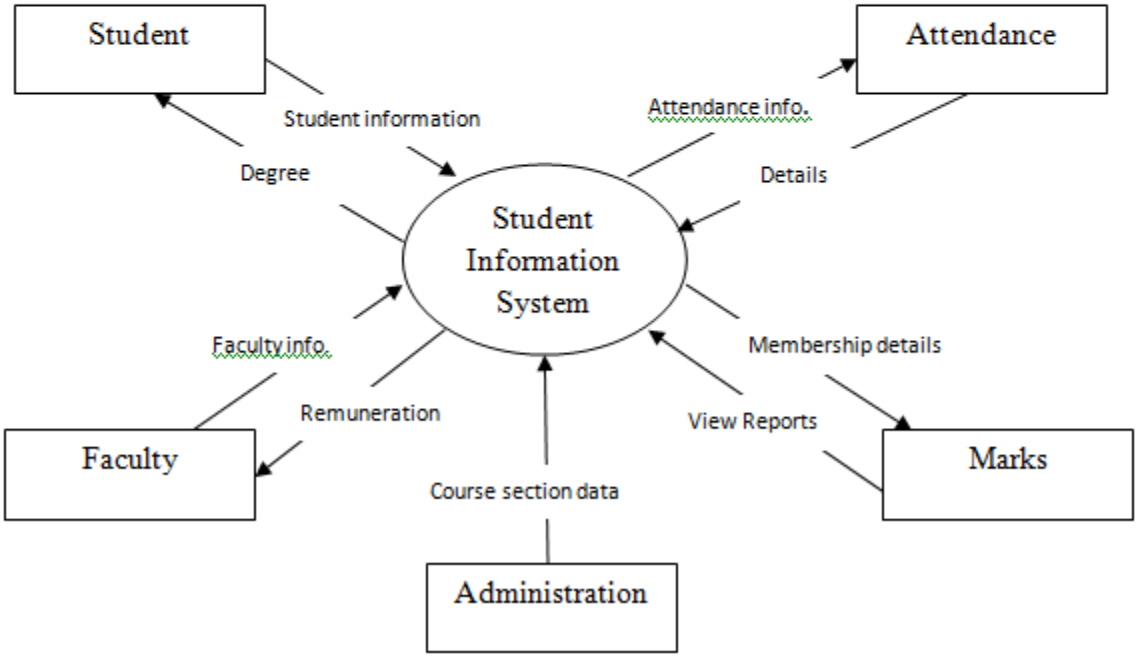
In our final concept we decided to implement both modes single player and multiplayer if the internet is available then users can play against each other or if there is no internet access still the game can be playable due to single player mode where users can complete exciting levels and earn reward.

4.4 Data Flow Diagrams:

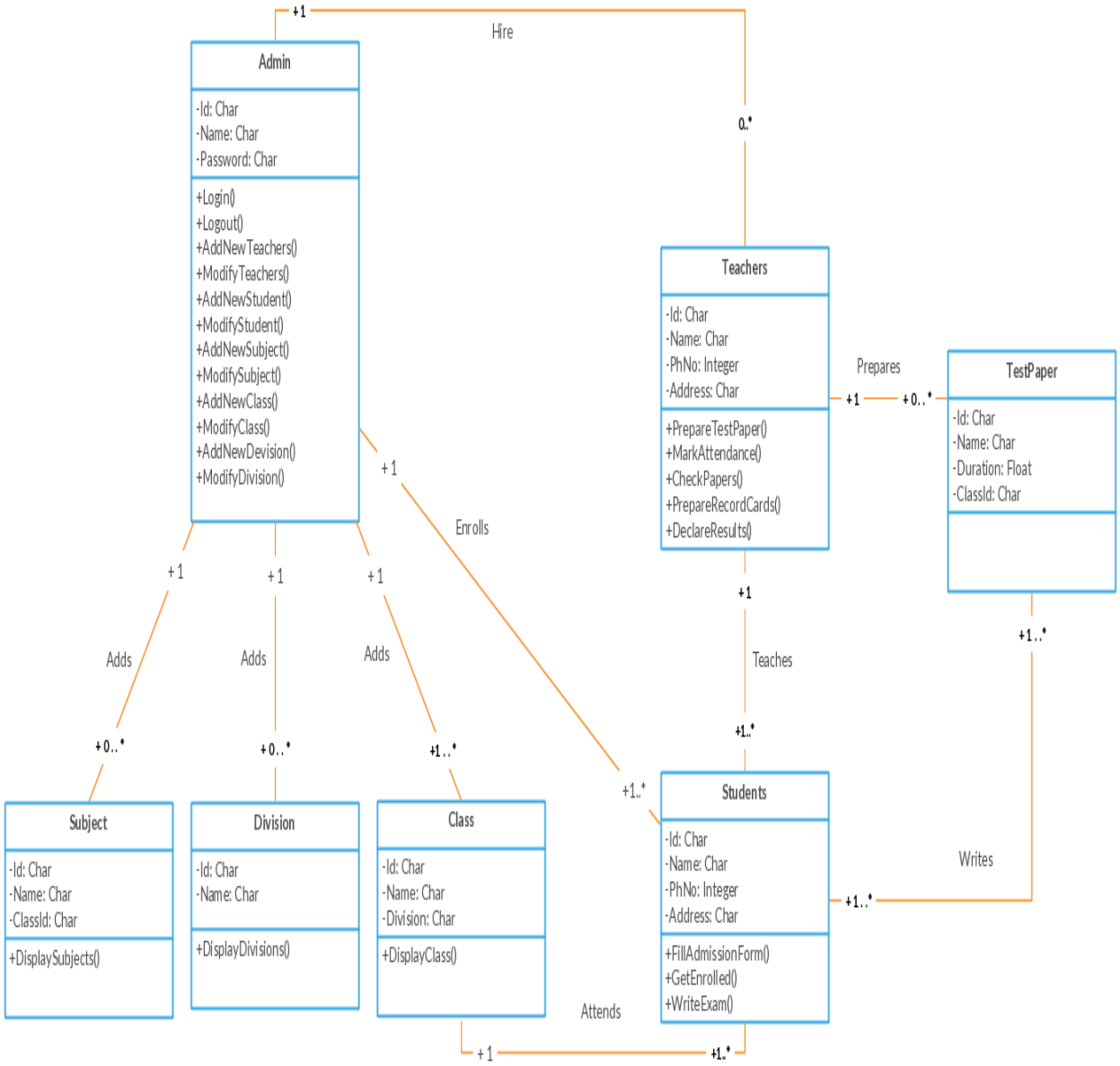
Level 0 DFD :



Level 1 DFD:

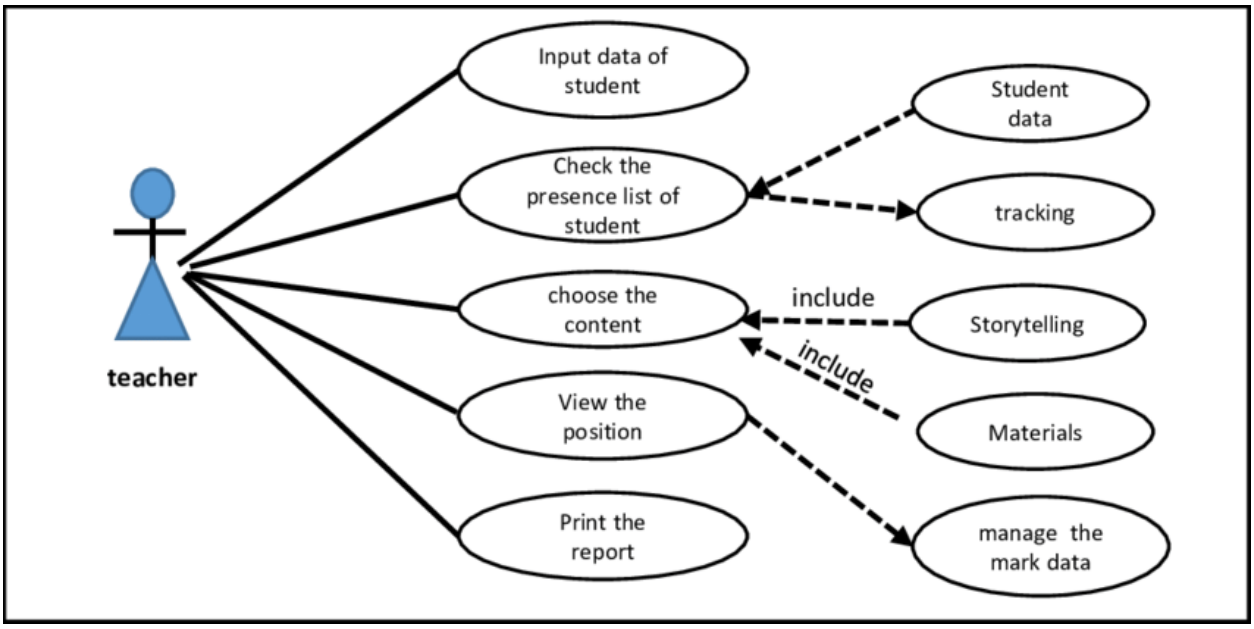


4.4..1 Class Diagrams:

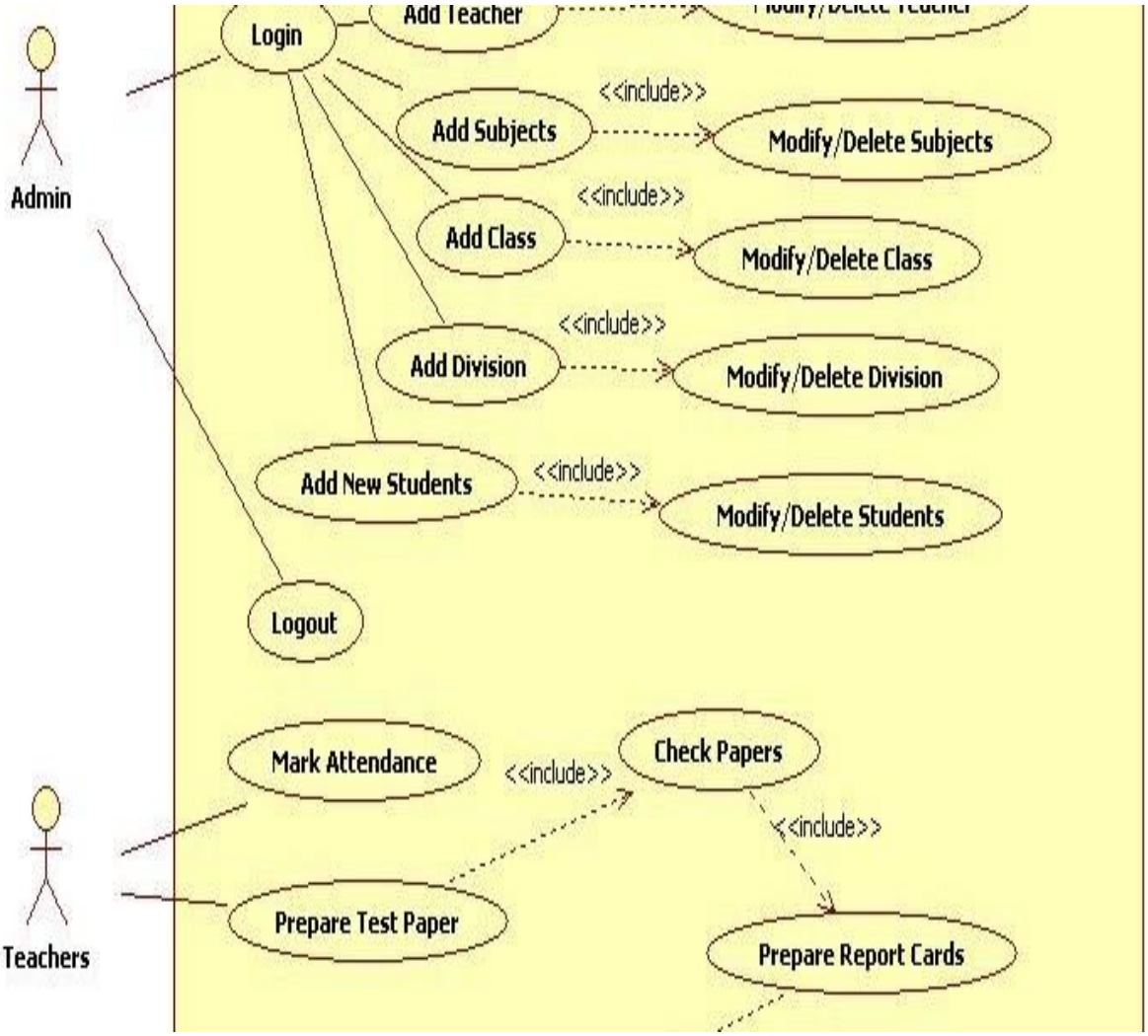


4.5 Use Cases Diagrams:

Use case diagram for teachers

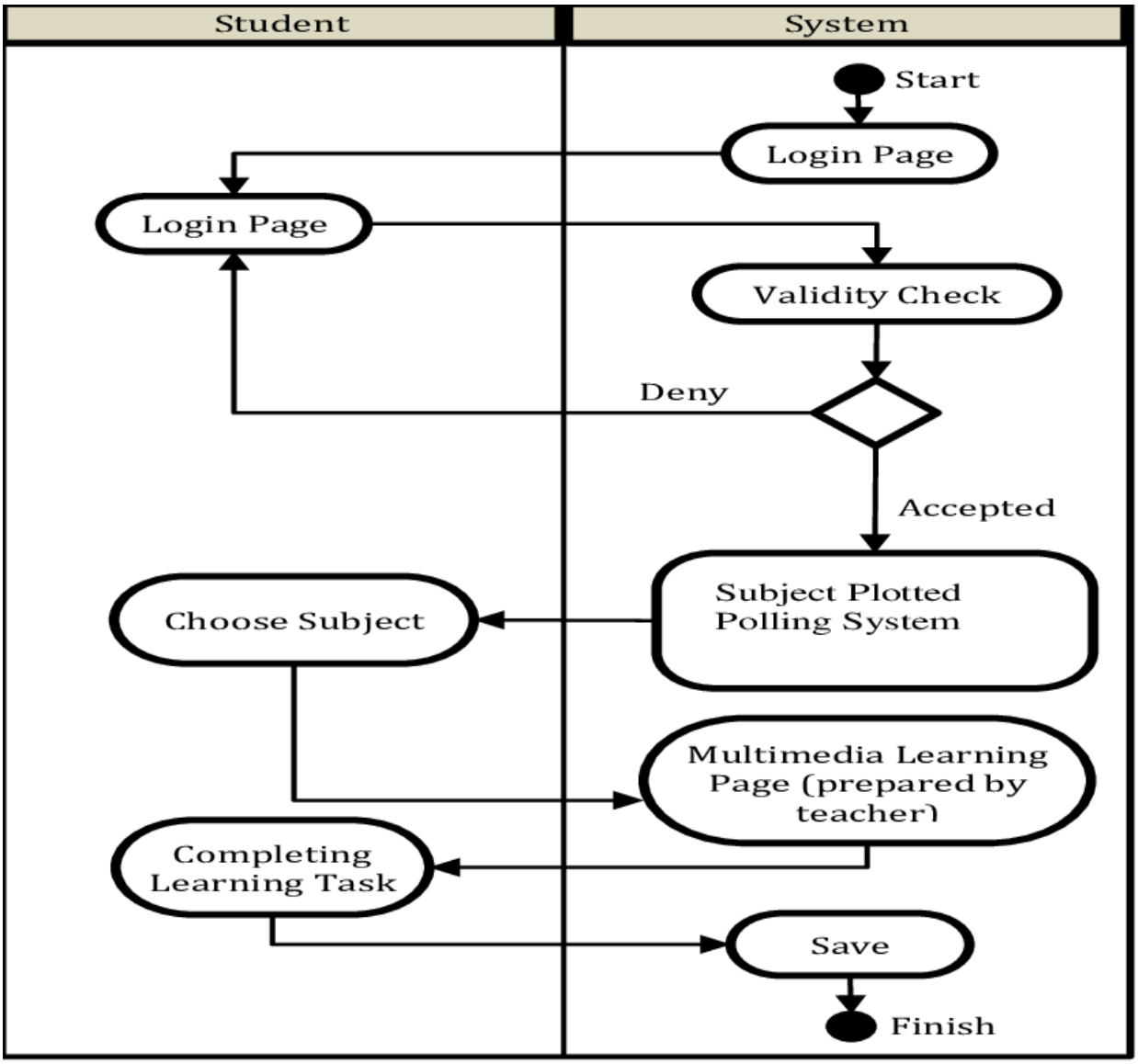


Use case for school system

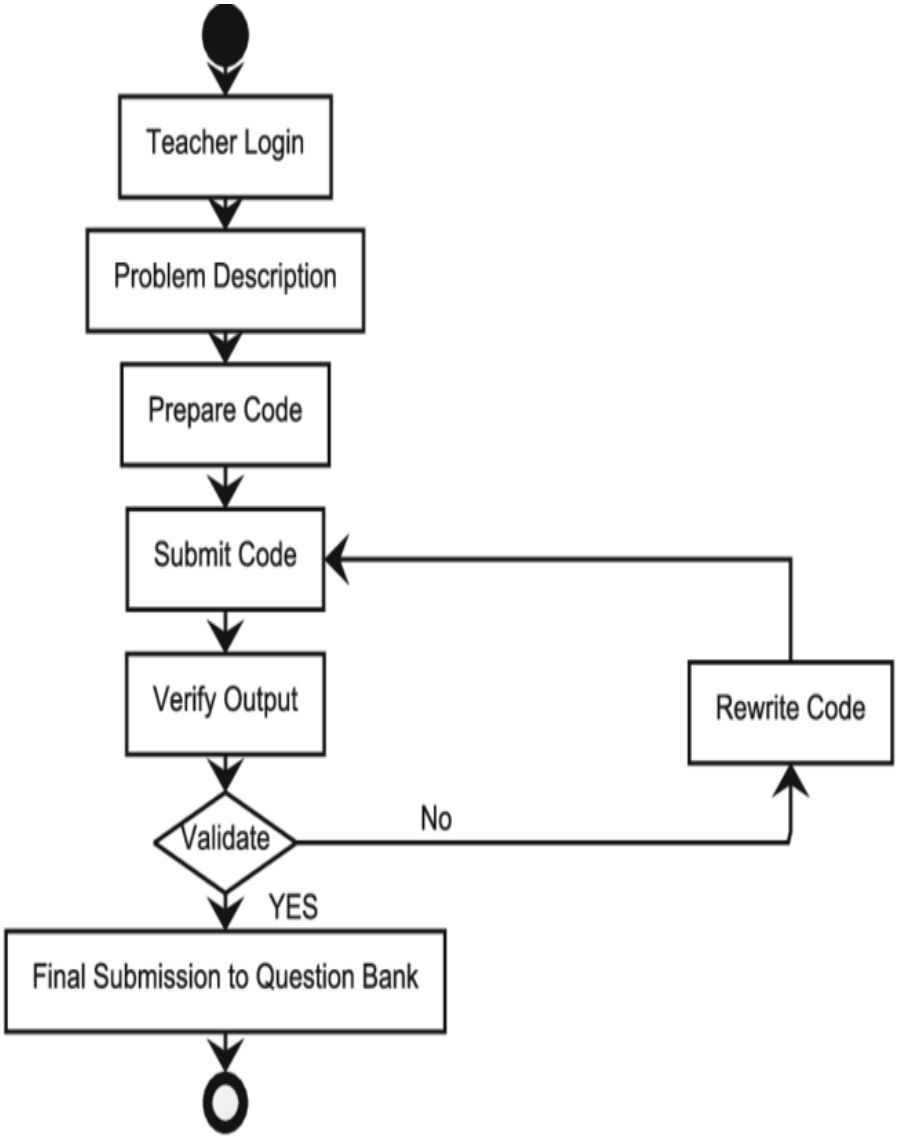


4.6 Activity Diagrams:

For student



For teacher


Teacher Login
Problem Description
Prepare Code
Submit Code
Verify Output
Rewrite Code
Validate
No
YES
Final Submission to Question Bank

4.7 System Requirements:

System must have following requirements:

- Android Minimum OS = 5.0
- At Least 1 GB Ram

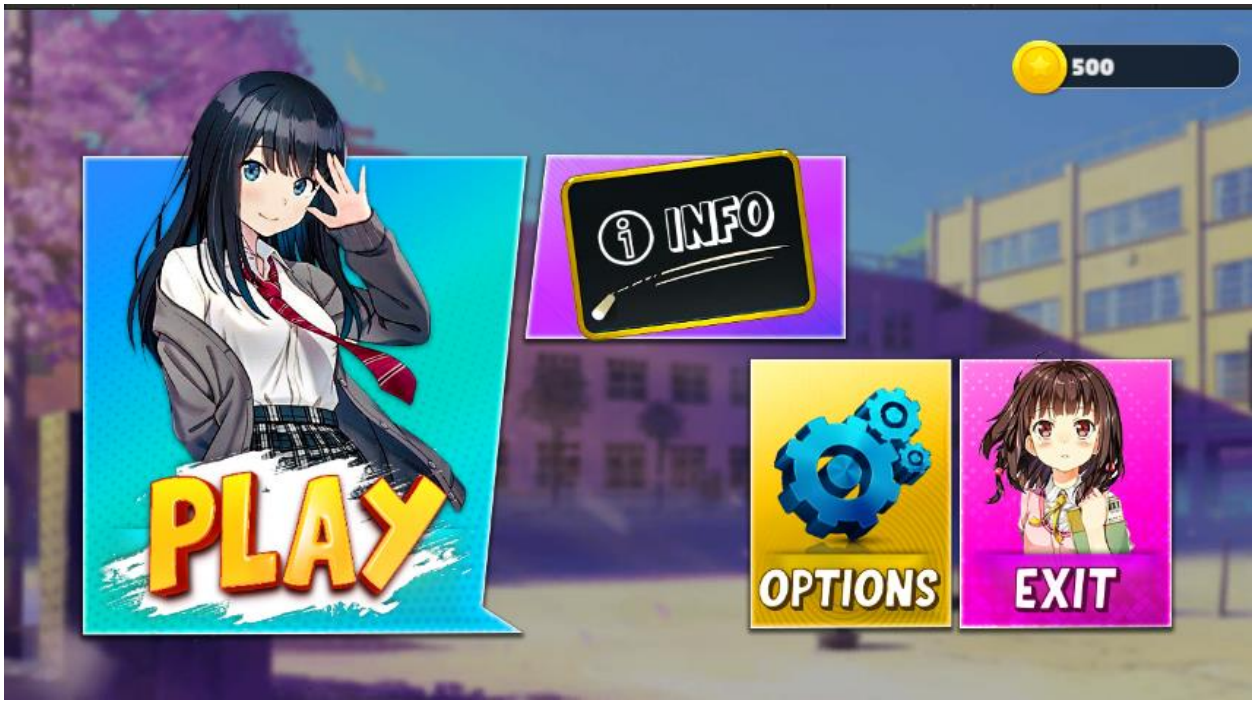
Chapter 5
Game Design and Implementation

5.1 Screen Shots and Working:

We have added screen shots of our Anime School Teacher Simulator Shooting Game in this section and explaining the working and functionality of the game.

5.1.1 Main Menu :

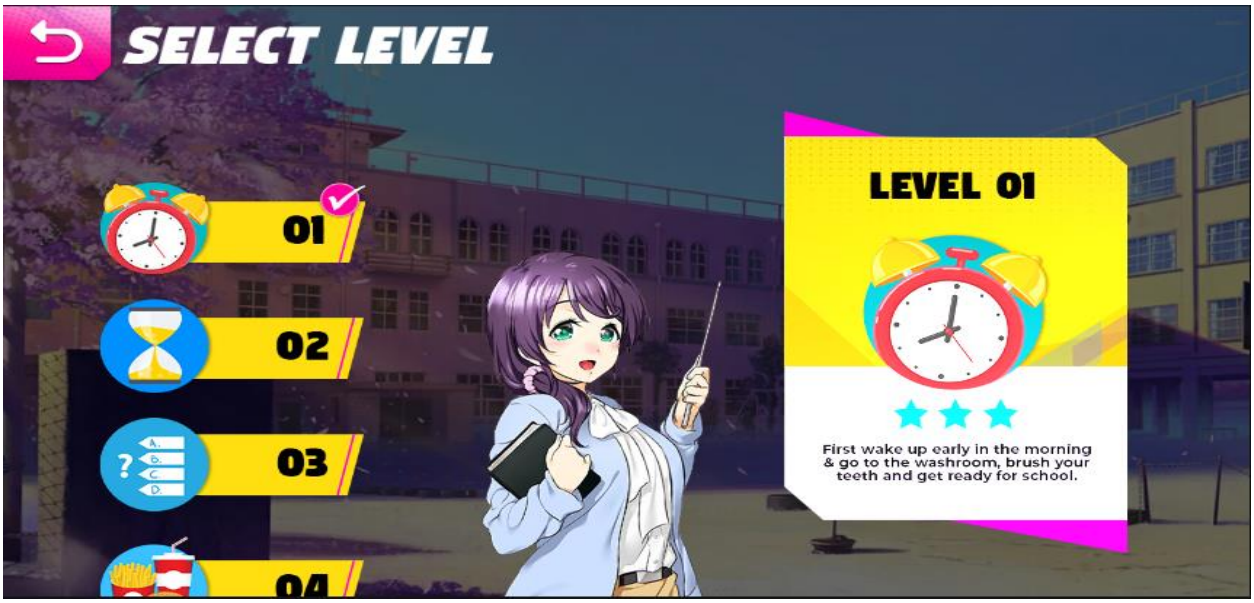
When the game will start, the first scene that will appear is the Main Menu. At the top of the scene, the game name will appear which is “Anime School Teacher Simulator Shooting”.



All these buttons are intractable and have next scene loading data behind. On pressing the Single Player button the gun selection menu will be open, the multiplayer button will take us to the Lobby and the Quit button is used to exit from the game.

5.1.4 Level Selection Menu :

On pressing the next button in the gun selection menu, a new level selection menu will be loaded. There are multiple levels all locked except the first level as you clear the first level. The next level will be unlocked which now you can access and the lock icon will be removed from this, similarly the process goes on.



5.1.5 Single Player Scene :

This is the single player gameplay scene where the player is holding a gun in his hand and can shoot enemies.

5.1.6 Player Controls GUI:

There is a joystick on the screen with which we can move our player.

JoyStick Up : Player will run in forward direction.

JoyStick Down : Player will run in backward direction.

JoyStick Left: Player will move in the left direction.

JoyStick Right: Player will move in the right direction.

There are other buttons on screen by pressing them, allowing the player to perform different actions.

Jump Action : Player will move upward as the jump button presses.

Camera Movement : On a touching screen, the camera will rotate around in 360 degrees.

5.1.7 Game Sounds:

There are multiple game sound used in this game list of these are mentioned below:

Background Music : At the load of every menu scene, attractive background music will be played which will please users to stick to the game for a long time.

Button Click Sound : At the press of any button, there is a specific click button that will be played each time, it will show our player is interacting with the button.

Level Complete Sound : Whenever the player will complete, then a cheerful sound will be played which will give the user a feeling that he has achieved something.

Level Failed Sound : If a player fails to complete any level then an interesting sound will be produced which will give the feeling to play again this level.

Player Waik Sound : Whenever the player will waik on ground, then the waking sound will play, that is the real world waik feel for game users.

Fire Sound : On pressing gun fire button, bullet fire sound will play.

Damage Sound : Whenever any bullet will hit our player, a hurt sound will play which will conscious our game user.

5.1.8 Enemy AI:

There are Single AI agents added in different levels of games that have capability to make decisions according to the situation. They use ray casting to identify what they are going to interact with. If on the front any collision comes, they can change their direction in the opposite direction. If there's a player near them, they will move toward the player and start shooting them. If a player goes out of the range they can start patrolling again between their waypoints.

These AI are capable of doing these:

- Patrolling
- Following
- Collision Detection

5.2 Important Scripts:

Here are the few important scripts from the game.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerCollision : MonoBehaviour
{
    GameObject interactedObject;

    private void OnTriggerEnter(Collider other)
    {
        if(other.gameObject.layer == LayerMask.NameToLayer("levelComplition"))
        {
            UIManager.instance.interactBtn.SetActive(true);
            interactedObject = other.gameObject;

        }
        else if(other.gameObject.layer==20)
        {
            if (GameManager.instance.LevelSelected == 3)
            {
                if (GameplayHandler.instance.companionFollowPoint.activeInHierarchy)
                {
                    GameplayHandler.instance.companionFollowPoint.SetActive(true);
                }
                else
                {
                    GameplayHandler.instance.companionFollowPoint.SetActive(true);
                }
            }

        }
    }
    Unity Message | 0 references
    private void OnTriggerExit(Collider other)
    {
        if(other.gameObject.layer == LayerMask.NameToLayer("levelComplition"))
        {
            UIManager.instance.interactBtn.SetActive(true);
            interactedObject = other.gameObject;

        }
        else if(other.gameObject.layer==20)
        {
            if (GameManager.instance.LevelSelected == 3)
            {
                if (GameplayHandler.instance.companionFollowPoint.activeInHierarchy)
                {
                    GameplayHandler.instance.companionFollowPoint.SetActive(true);
                }
                else
                {
                    GameplayHandler.instance.companionFollowPoint.SetActive(true);
                }
            }

        }
    }
    Unity Message | 0 references
    private void OnTriggerExit(Collider other)
    {
        UIManager.instance.interactBtn.SetActive(false);
    }

    0 references
    public void OnInteraction()
    {
        GameplayHandler.instance.CheckPointUpdate();
        interactedObject.GetComponent<Interactable>().onInteraction();
    }
}
```

5.2.1 Player Movement:

```
1 reference
public virtual void ControlKeepDirection()
{
    // update oldInput to compare with current Input if keepDirection is true
    if (!keepDirection)
    {
        oldInput = input;
    }
    else if ((input.magnitude < 0.01f || Vector3.Distance(oldInput, input) > 0.9f) && keepDirection)
    {
        keepDirection = false;
    }
}

/// <summary>
/// Determine the direction the player will face based on input and the referenceTransform
/// </summary>
/// <param name="referenceTransform"></param>
1 reference
public virtual void UpdateMoveDirection(Transform referenceTransform = null)
{
    if (isRolling && !rollControl || input.magnitude <= 0.01)
    {
        moveDirection = Vector3.Lerp(moveDirection, Vector3.zero, (isStrafing ? strafeSpeed.movementSmooth : freeSpeed.movementSmooth) * Time.de
        return;
    }

    if (referenceTransform && !rotateByWorld)
    {
        //get the right-facing direction of the referenceTransform
        var right = referenceTransform.right;
        right.y = 0;
        //get the forward direction relative to referenceTransform Right
        var forward = Quaternion.AngleAxis(-90, Vector3.up) * right;
        // determine the direction the player will face based on input and the referenceTransform's right and forward directions
                    OnFinishSprinting.Invoke();
                }
                else if (!isSprinting)
                {
                    OnStartSprinting.Invoke();
                    isSprinting = true;
                }
            }
            else if (!useContinuousSprint && isSprinting)
            {
                if (currentStamina <= 0)
                {
                    finishStaminaOnSprint = true;
                    OnFinishSprintingByStamina.Invoke();
                }
                isSprinting = false;
                OnFinishSprinting.Invoke();
            }
        }
        else if (isSprinting && (!useContinuousSprint || !sprintConditions))
        {
            if (currentStamina <= 0)
            {
                finishStaminaOnSprint = true;
                OnFinishSprintingByStamina.Invoke();
            }

            isSprinting = false;
            OnFinishSprinting.Invoke();
        }
    }

    /// <summary>
    /// Manage the isCrouching bool
    /// </summary>
```

```
            StopCharacterWithLerp();

            transform.position = animator.rootPosition;
            transform.rotation = animator.rootRotation;
        }

        else if (inputSmooth == Vector3.zero)
        {
            animator.ApplyBuiltinRootMotion();
            transform.position = animator.rootPosition;
            transform.rotation = animator.rootRotation;
        }

        if (useRootMotion)
            MoveCharacter(moveDirection);
    }

    /// <summary>
    /// Set the Controller movement speed (rigidbody, animator and root motion)
    /// </summary>
    1 reference
    public virtual void ControlLocomotionType()
    {
        if (lockAnimMovement || lockMovement || customAction || isRolling) return;

        if (!lockSetMoveSpeed)
        {
            if (locomotionType.Equals(LocomotionType.FreeWithStrafe) && !isStrafing || locomotionType.Equals(LocomotionType.OnlyFree))
            {
                SetControllerMoveSpeed(freeSpeed);
                SetAnimatorMoveSpeed(freeSpeed);
            }
            else if (locomotionType.Equals(LocomotionType.OnlyStrafe) || locomotionType.Equals(LocomotionType.FreeWithStrafe) && isStrafing)
            {
                isStrafing = true;
```

5.2.1 AI Movement:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.AI;

Unity Script (1 asset reference) | 1 reference
public class MarkerAI : MonoBehaviour
{
    public static MarkerAI instance;
    public Transform Destination;
    public GameObject MarkerStartPos;
    Transform StartPos;
    //public Transform target;
    public int CurrentDesNo;
    public NavMeshAgent Agent;
    float Dis;
    Unity Message | 0 references
    private void Awake()
    {
        instance = this;
    }
    Unity Message | 0 references
    private void Start()
    {
        StartPos = GameplayHandler.instance.Player.transform;
    }

    Unity Message | 0 references
    private void Update()
    {
        Dis = Vector3.Distance(transform.position, Destination.position);
        Agent.SetDestination(Destination.position);
        if (Dis <= 1)
        {
            //gameObject.SetActive(false);
        //  StartCoroutine(InActive());
```

```
    Agent.SetDestination(Destination.position);
    if (Dis <= 1)
    {
        //gameObject.SetActive(false);
      //  StartCoroutine(InActive());
       // Invoke("InActive", 3);
        gameObject.transform.position = StartPos.position;
        Invoke("Active", .8f);
    }

}
0 references
void Active()
{
    gameObject.SetActive(true);
    //Invoke("InActive",2);
}
0 references
public void OnStartPos()
{
    gameObject.SetActive(false);
    gameObject.transform.position = StartPos.position;
    Invoke("Active", .5f);
}
0 references
IEnumerator InActive()
{
    yield return  new WaitForSeconds(2f);
    gameObject.SetActive(false);
    yield return new WaitForSeconds(2f);
    gameObject.SetActive(true);


}
```

Chapter 6
Test Cases

6.1 Upgrade Menu:

Test Case ID	Upgrade Menu
Use Case Reference	2
Purpose	Upgrade Menu is opening properly
Environment	Unity 3D
Pre Requirement	Click on Single Player Button
Expected Result	After Clicking on Upgrade it should work
Actual Result	Upgrade Event work successfully

6.2 Gun Purchase and Selection:

Test Case ID	Gun Purchase and Selection
Use Case Reference	3
Purpose	Buy and Select
Environment	Unity 3D
Pre Requirement	Click on Single Player Button
Expected Result	Player will move by clicking button
Actual Result	Player is Selecting Successfully

6.3 Single Player:

Test Case ID	Single Player Menu
Use Case Reference	1.1
Purpose	Open Single Player Menu
Environment	Unity 3D
Pre Requirement	Click on Game Icon
Expected Result	After Clicking Single Player Button Selection Scene should Open
Actual Result	Scene is Opening Successfully

6.4 Multiplayer Player:

Test Case ID	MultiPlayer Lobby
Use Case Reference	4
Purpose	Multiplayer Lobby View
Environment	Unity 3D
Pre Requirement	Click on Multiplayer Button
Expected Result	After Clicking MultiPlayer Button multiplayer Lobby should be Open
Actual Result	Lobby is Opening Successfully

6.5 Play Game:

Test Case ID	Play Game
Use Case Reference	1
Purpose	Play Game View
Environment	Unity 3D
Pre Requirement	Click on Game Icon
Expected Result	After Clicking Game user should Play Game
Actual Result	User is capable of Playing Game

6.6 Exit Game:

Test Case ID	Exit Game and student Reward
Use Case Reference	5
Purpose	Getting Reward and skills
Environment	Unity 3D
Pre Requirement	Join Any Single Player Level
Expected Result	After Exiting Game user should Get Reward
Actual Result	User is getting Reward

9 Level Clear:

Test Case ID	Level Clear
Use Case Reference	6
Purpose	Clearing of Level
Environment	Unity 3D
Pre Requirement	Must have clear previous Level
Expected Result	After Killing all enemies level should be completed
Actual Result	Level Completion is working perfectly

6.10 Create Room:

Test Case ID	Create Room
Use Case Reference	7
Purpose	Creating of Room
Environment	Unity 3D
Pre Requirement	Must have Joined Lobby
Expected Result	Class Room Should Create with name
Actual Result	Class Room is creating successfully

6.11 Join Room:

Test Case ID	Join Room
Use Case Reference	8
Purpose	Joining of Class Room
Environment	Unity 3D
Pre Requirement	Multiplayer Menu
Expected Result	Existing Rooms Should be joined successfully
Actual Result	Existing Room is Joining successfully

6.12 Play Field:

Test Case ID	Play Field
Use Case Reference	9
Purpose	Play Field Working
Environment	Unity 3D
Pre Requirement	Level Selection or Room Joining
Expected Result	Play Field Should work properly
Actual Result	Play Field is working

Chapter 7

Conclusion and Future Work

7.1 Conclusion:

The End product of the game looks similar to what we had thought before starting the project. We played with our friends and family members. Almost everyone loved our product so it's kind of a little success for us too before releasing it into the market. We tried our best in making this game by using the latest technologies and after analyzing trending games in the market. Although nothing in this word can be perfect so with time we will try to improve it more according to the users demand.

7.2 Future Work:

There is a lot of work that can be done to improve the game which we could not do because of the time limit. New features like flying mode and new maps can be added to make it more interesting. After reading the users data, we will try to optimize it more such as on which devices the game is lagging and crashing. Game Retention is an important concept in the app and game industry so we will be looking into how we can increase retention rate. New Bugs that we would face, it will be in our priority list to optimize them so users can enjoy a smooth and reliable experience of the game.

7.2.1 Number of Users:

Currently we are using a free version of photon cloud in our game for multiplayer purposes in which 20ccu users can play at one time. As the number of users will increase, the number of limits would also increase.

References

References :

[1]. Csar Villacs, Walter Fuertes, Andrs Bustamante, Daniel Almachi, Carlos Procel, Susana Fuertes, and Theo_los Toulkeridis. Multi-player Educational Video Game over Cloud to Stimulate Logical Reasoning of Children. 2014 h
mmjIEEE/ACM 18th International Symposium.

[2] Anderson, Joshua. Innovative navigation artificial intelligence for motor racing games. 2016, University of Bedfordshire.

[3] Dayan, Shangguan, and Huang Xinyuan. Development of Unity3D Net work Functionality on Android. International Journal of Hybrid Informa tion Technology 9.2 (2016): 359-370.

[4] Dhuri, Shroneet, et al. Game Development for Android Device using Unity 3D

[5] Mike ,G (2012) Unity Game Development is 24 Hours. Sams; Third Edition

[6] Joe Hocking and Jesse ,S (2009) Unity In Action: Multiplatform. Manning Publications; Second Edition

[7] Matt Smith and Chico ,Q (2005) Unity 5.x Cookbook. Packt Publishing Limited; Third Edition

[8] Jerry Gibson ,B (2017) Introduction to Game Design, Prototyping, and Development. Addison Wesley; Third Edition

[9] Patrick ,F (2014) Unity From Zero to Proficiency(Foundations). Independently Published; First Edition

GAMEEE

ORIGINALITY REPORT

21%	20%	4%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	hellgeeks.com Internet Source	10%
2	Submitted to Higher Education Commission Pakistan Student Paper	7%
3	Submitted to University of West Florida Student Paper	2%
4	www.coursehero.com Internet Source	1%
5	Submitted to South Bank University Student Paper	<1%
6	Submitted to University of Nottingham Student Paper	<1%
7	Submitted to University of Western Sydney Student Paper	<1%
8	businessdocbox.com Internet Source	<1%
9	research-repository.griffith.edu.au Internet Source	<1%