Jessica Panek

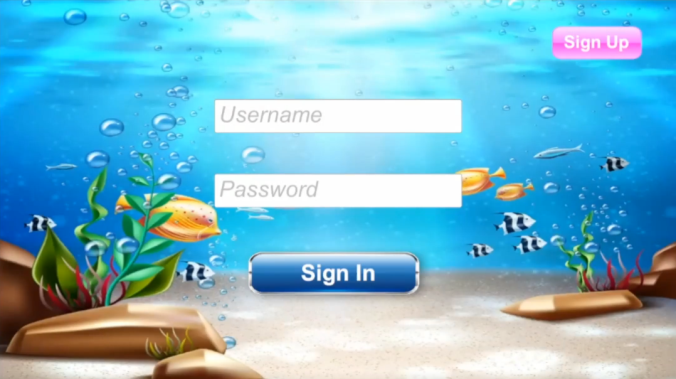
Project Documentation



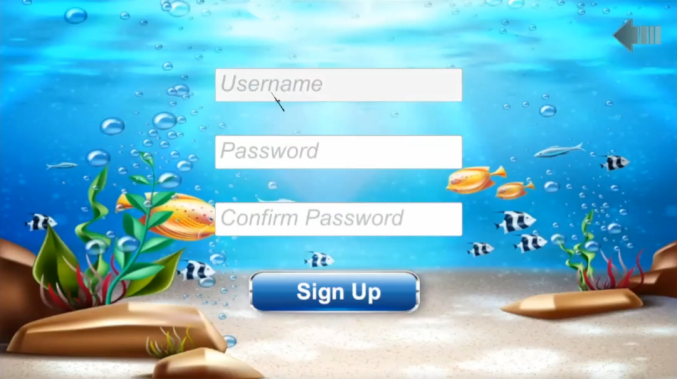
# **Executive Summary**

My project is a 3D fishing game. The game will take place at a lake where you can control a characte to go fishing. In the lake you can catch bluegill, bass, and Muskie. Each fish can then be sold for its valued cost giving the user coins in return. This game is targeted for all ages to relax and play. Due to the simplicity of the game level it's great for kids and unwinding for older ages.

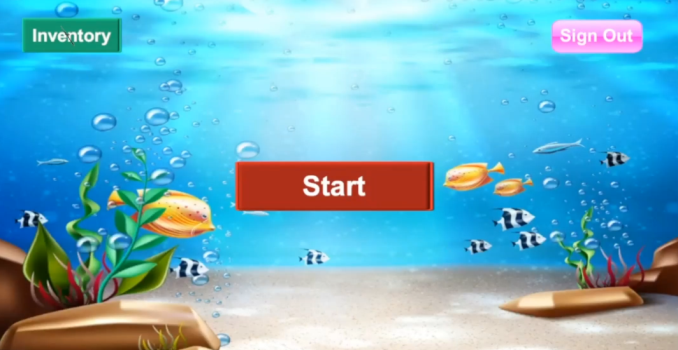
Scenario 1 | Log In

This is the login screen which will appear once the game starts. The user can enter their existing account information or click sign up og the top right to create an account. 

Scenario 2| Sign Up

On the sign up screen an additional field is shown for the user to enter their password twice for confirmation. Also this screen allows the user to go back to the login screen and submit their entries to create an account.

Scenario 3| Main Menu

After an account is logged in the display will show the main menu of the game. This screen allows the user to sign out, view their inventory, and then start the game. 

Scenario 4| Game Lake Environment 

Once the player starts the game they will be brought to the game environment where they can control the character and move around the lake.

Scenario 5| Start Fishing

When the player walks up to the lake the throw button will appear. Once the player clicks the throw they will begin fishing.  
  
  
  


Scenario 6| Waiting for Bite

After the throw button is clicked the player waits a short while until a fish bite.   
  
  


Scenario 7| Catching

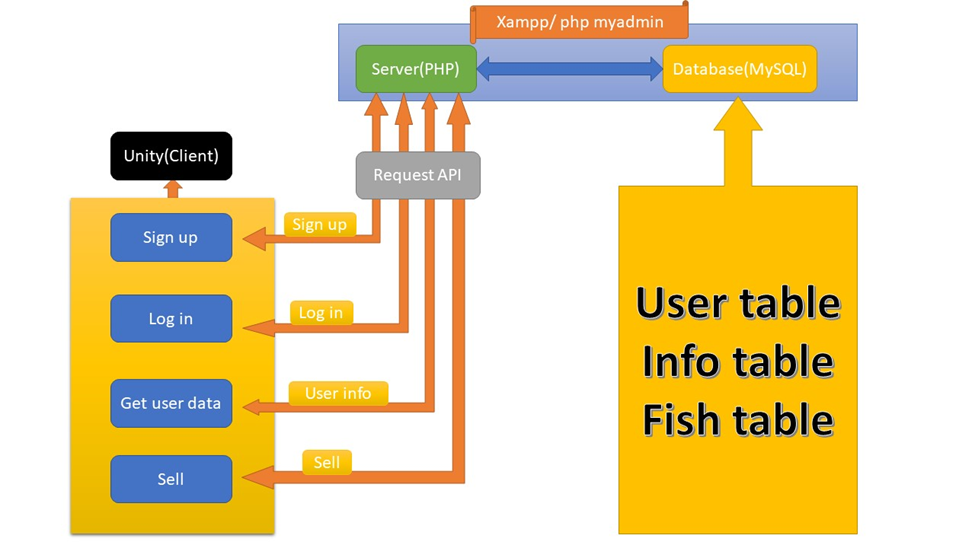
When a fish bite the bait then the fish ico will appear along with the reel button. The player must click the reel button to catch the fish.  
  
  
  
  


Scenario 8| Inventory

# After the user catches some fish they can view their inventory to see their totals. Also in their inventory they can sell the fish to increase their coin amount.

# System Architecture

Fishing game flowchart:



In our unity fishing game there are several functions such as sign up, log in , get user data, and sell. Which are connected into the MySQL database using their own request API using PHP. The PHP and MySQL user interface is Xampp which is an open source package that provides a graphical interface for SQL making it easier to maintain and adjust.

## Source Code Structure

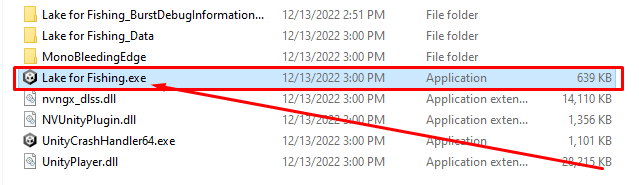
Directories:

|  |  |
| --- | --- |
| **Code Directory** | |
| **Directory** | **Usage** |
| Unity/assets/scripts/fishing engine.cs | This is a game control engine including some APIs such as read, sell and catch url. |
| Unity/assets/scripts/fishing UI.cs | This is the game user interface including several click actions. |
| Backend/Models/fish.php | This is the backend fish database model. |
| Backend/Models/Info.php | This is the backend info database model. |
| Backend/Models/User.php | This is the backend user database model. |
| Backend/Http/Controllers/Controller.php | This is the backend controller to authorize, dispatch, validate requests. |
| Backend/Http/Controllers/DataController.php | This is the controller to manage request data from Unity through API. |

# Executables

### Lake for Fishing (Lake for Fishing.exe)

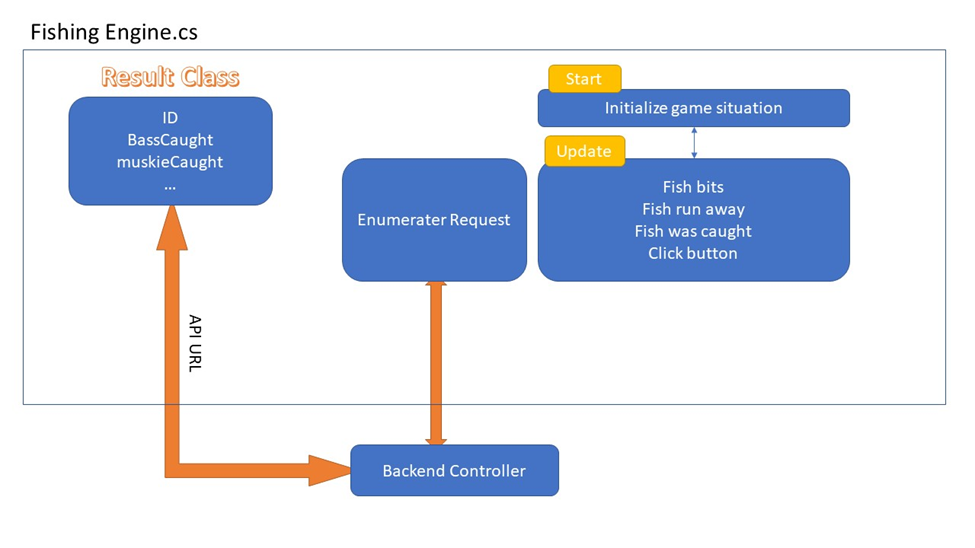
### The game is launched by double-clicking the.exe file.



# Code Architecture

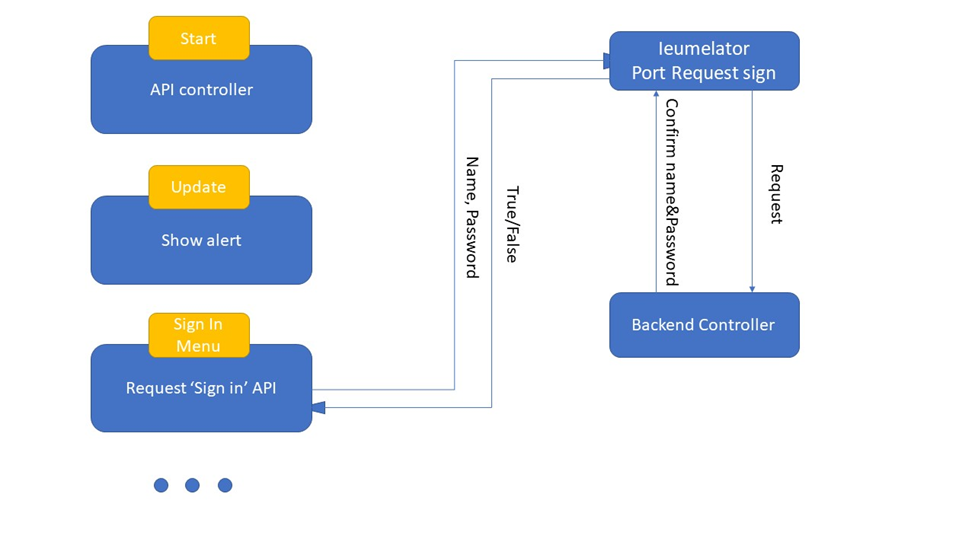
The Fishing game is run with through the Unity 3D engine and C# scripts. While the backend of the game is run with MySQL and Laravel.

Unity/assets/fishing engine.cs



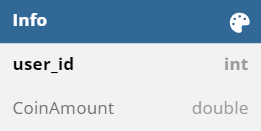
If the start() function is running then the initial game situation is selected by the player state such as first time ,holding  rob and fishing. In this time the player or rod face and camera are set as active. And then in the update(); the fish avatar or reelbutton state is changed according to the fish actions such as fish bits, fish run away and caught. Next, the button click functions are defined. Finally the api requests are called into the backend  database with the Ienumerater mode then the respond data is displayed in the text field.

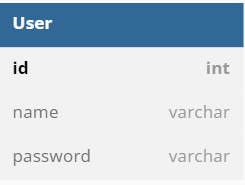
Unity/assets/fishing UI.cs



## The UI elements are defined and controlled by the player. This creates a user interface that creates a new member, and logs in. At first, the sign in menu, the name and password are delivered into the backend controller using the Port Request sign , the info is verified by the controller with the model. Then the return info is boolean and it goes into unity c# again to determine a result.

## Database or Data Store



There are 3 relation tables used in the database design: User, Info, Fish

User - This table includes the information of an user:

. id(primary key, auto increase)

. name(unique)

. Password

Fish - The table includes the info of fishes the user caught in seven fields:

. user\_id

. Basscaught

. MuskieCaught

. BlueGillCaught

. BassTotal

. MuskieTotal

. BlueGillTotal

Info - The table includes the coin amount of a user in two fields:

.user\_id

.CoinAmount: The amount of coin that the user owns now.

## Views, Stored Procedures and User Defined Functions

The Laravel framework was implemented for the backend. Laravel has MVC structure, which are Model, View and Controller. The main one is the Controller 2 controllers are used UserController and DataController:

UserController.

.signup(): This function is called when the user signs up the game. This registers a new user.

.login(): This function is called when the user logs in the game.

DataController

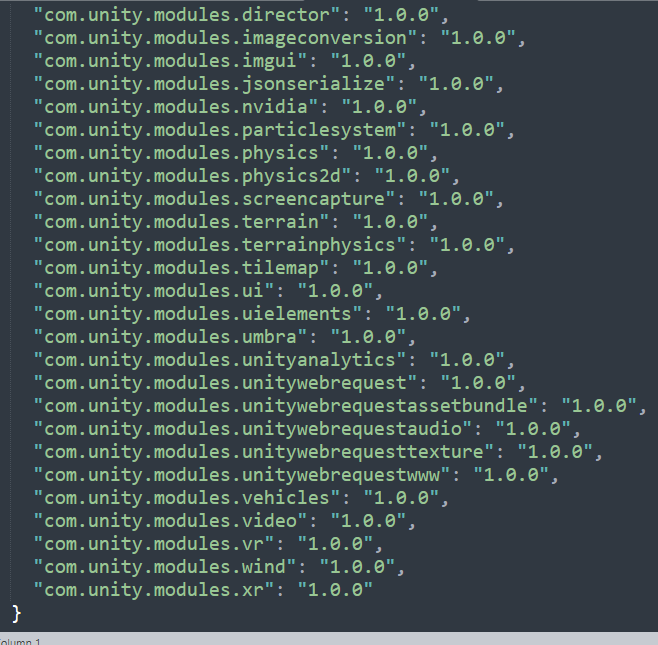
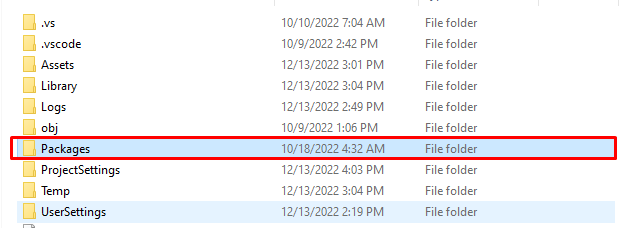
.getUserData(): This gets the coin amount and total counts of each fish of the user.

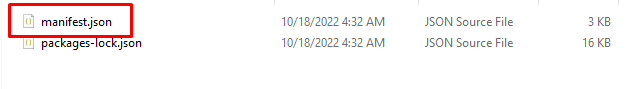
.sell(): This reduces the counts of each fish sold and increases the coin amount.

.catch(): This increases total counts of caught fishes.

External Files & Data

Unity packages used for this project:





Packages for Development: com.unity.ide.visualstudio, com.unity.ide.vscode

Packages for environment: com.unity.modules.particlesystem, com.unity.modules.terrain, com.unity.modules.terrainphysics, com.unity.modules.wind, com.unity.modules.terrain-tools

Packages for Third Person Controller: com.unity.modules.vr, com.unity.modules.xr

Extras: internal packages supported from Unity3D

Programming Language | Unity3D, C#, Laravel, MySQL

C# is used for the Unity Project and PHP(Laravel) for backend.

C# is used to write code for a fishing game engine.

PHP(Laravel) is used for Backend and integrate Unity into PHP via API.

As for a database, MySQL is used.

Project Classes

### Game Engine| FishingEngine.cs

This class includes all game logics.

Throw, reel, catch, move, API integrations etc.

### Game UI| FishingUI.cs

This class includes game UI interface.

Showing inventory, Sign Up / Sign In interface etc.

Database | BaseController.php

Creates a controller with the function of api calls

Database | DataController.php

Contains a controller to manage Data information to connect the unity frontend.

Database | UserController extends Controller

Contains a controller to manage User information to connect the unity ui.

Project Modules

Third person controller: Starter Assets - Third Person Character Controller

Used module for controlling the player.

Environment: NatureStarterKit2 - used module for setting environment like terrain, trees, grasses and so on.

Water: IgniteCoders - Used this module for placing water on the lake.

Program Start and End Flow













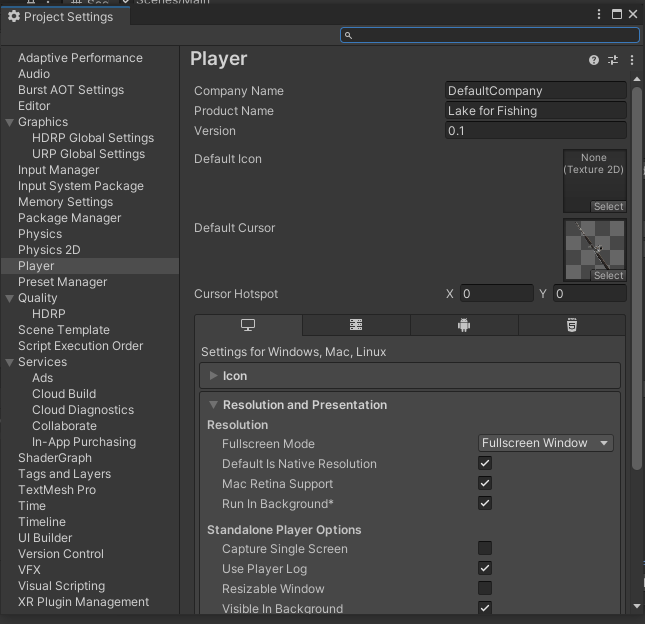


Summary

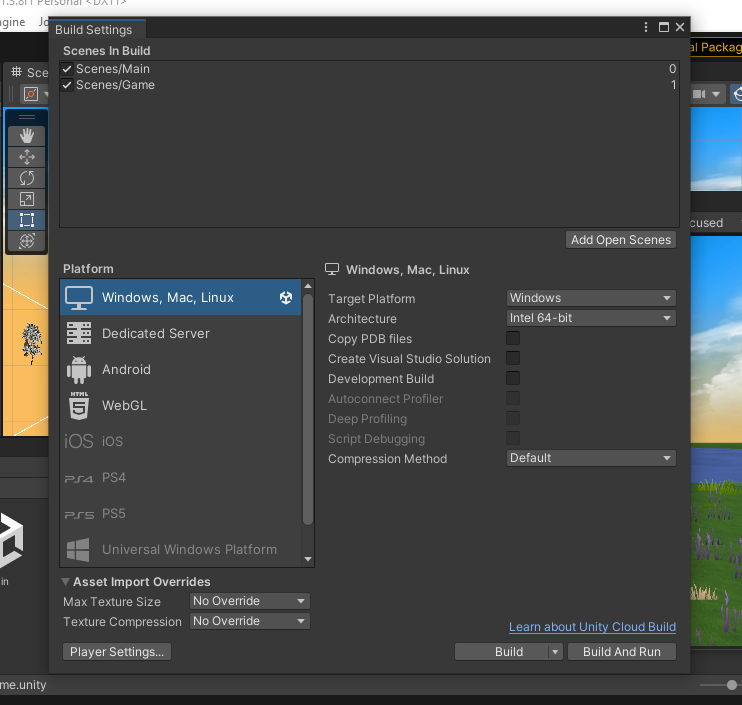
This project is a 3D fishing game using one of the most powerful game development engines Unity3D. This 3D game is in third person view with the character holding a fishing rod in a simple environment with a lake. The character should move with WASD, cast with ‘click’, and reel with ‘hold click’. When catching a fish, the fish might fall off the hook making the game more challenging. The game uses a local database MySQL to save the game information communicating with Unity game engine through Laravel Backend APIs. When players catch fishes, it will be saved to MySQL Database via PHP backend.

# BUILD AND RELEASE PROCESS

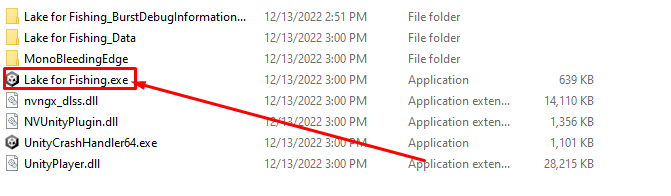
To change project settings:



The project can be further built with Desktop, Android, WebGL and more options.



# USER INSTALLATION INSTRUCTIONS)



Users can open the Lake for Fishing.exe file to play this game; then run PHP Backend (Laravel) for database connection.

# DEVELOPER SETUP INSTRUCTIONS

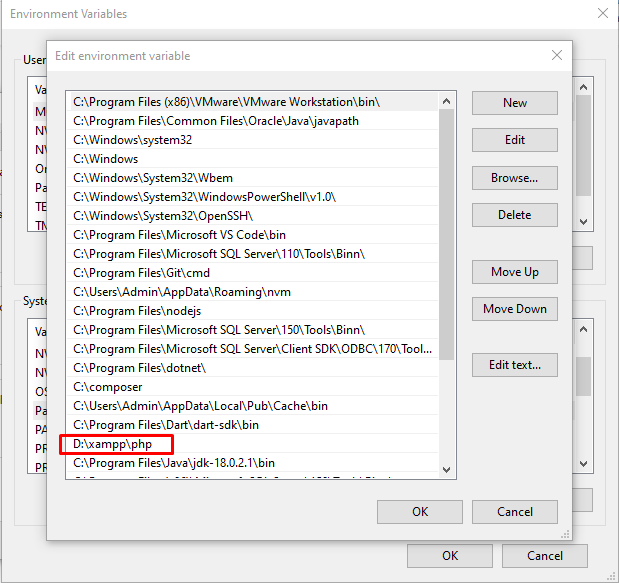
Development Environment: Unity 2021.3.8f1, Visual Studio, Xampp, Visual Studio Code

1. Developers need to install Unity. It is developed with Unity 2021.3.8f1.
2. In order to write and edit code, Visual Studio must be installed.
3. As for the backend, PHP(Laravel) is used in this project. Install Xampp and VS Code.

Backend Part.

1. Install Xampp.
2. Set the following folder’s path as the environment path.

Installation folder of xampp/php



1. Open the command prompt in the backend folder.
2. Then write the following command. “php artisan serve”