Group 13

Project Name: Snake Eating

Group Members:

1. 吴东阳 **118010324**
2. 刘俊达 **118010184**
3. 吴仲越 **118010338**
4. 赵昌浩 **118010435**

A Classic Web Game: Snake Eating

1. Introduction

1. 1 Project Overview

Our project aims to provide a new version of snake game for the players of this famous game. We also provide a record system for the game, where players can see their game record and the top100 players of the game among all the players. Besides, we also make the API available for other games, thus we can append more games in our system.

1.2 Objective

Our game comes from the famous game Snake. Snake is a video game genre where the player maneuvers a growing line that becomes a primary obstacle to itself. The concept originated in the 1976 two-player arcade game Blockade from Gremlin Industries, from there, hundreds of versions of the game has been provided in different platform. Our team plan to provided a new version of the Snake game for the players.

1.3 Highlights

In Snake we can adjust the difficulty (so there is a different initial speed) and the speed of snake (but not below the initial speed). On the other hand, we designed different skins for the snake, which can also be changed before the game starts, so players can play with different skins. For the UI interface, we designed a list of top 10 players with scores, and every player can query this list, their scores will be updated in the database in real time, and the list will be updated if their scores make it to the list.

1.4 Systems Features

Our system is mainly composed of 3 parts, including registration and login, game playing and a ranking-recording system, whereas players of our game could see their record and the top 100 scores among all the players in their home page and. More details of these features will be included in later sections.

2. System Architecture design by DFD

2.1 System Architecture

We will describe the system architecture of our system in this part. Architecture diagram of the whole system and UMLs of the server end will be presented, together with the text explanation of our design. We would also briefly discuss what platform and language we are planning to use during the development of our system.

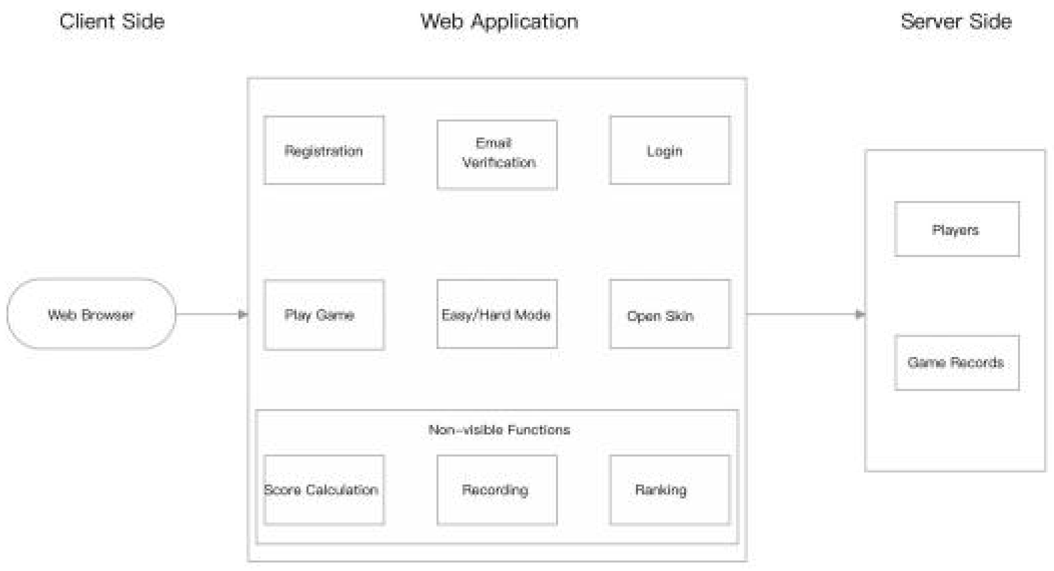


Figure 1 Components Graph

The above figure describes the overall architecture design of our system. Users can get access to our system through a web application. With the user interface, players will be able to register an account before starting the game, we will also do the email verification to promote safety. Logging in with his/her own account, the user can check his/her historical game records, including the time and scores. Players can also see a ranking list of the top 100 players on the home page. Functions designed to facilitate some features of our system will be hidden from users. They basically do calculations and logic controls of data. As for the server side, there will be tables in the database storing the information of players and game records.

2.2 DFDs

2.2.1 User Login and Registration

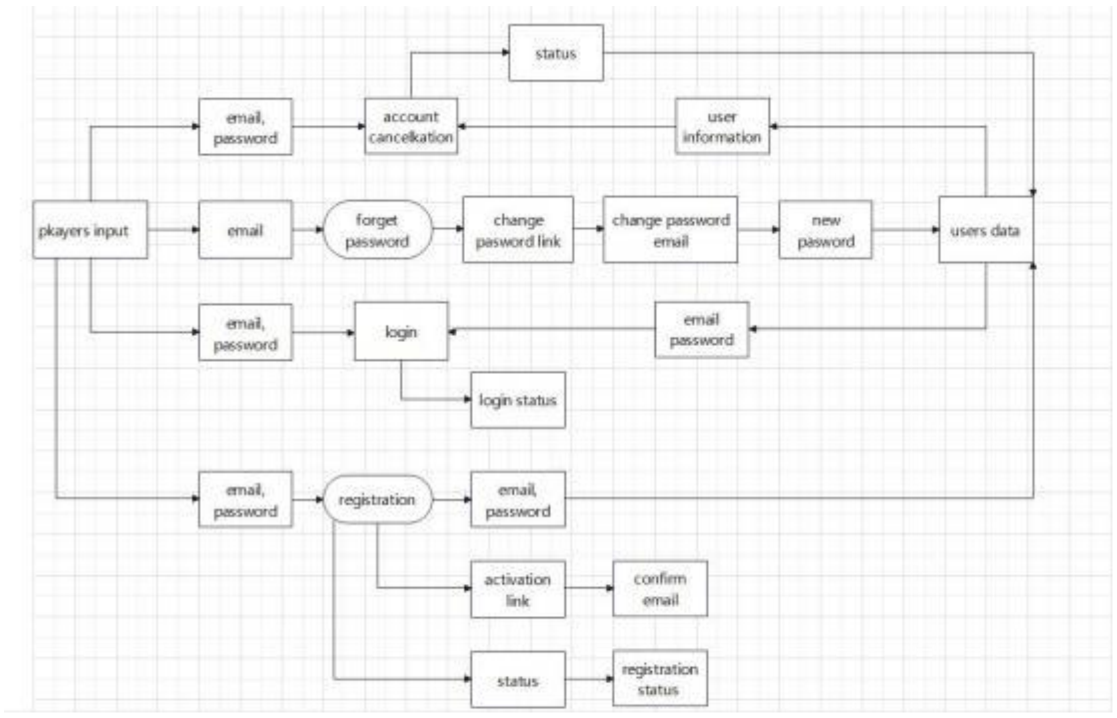


Figure 2 Data Flow Diagram: User Login and Registration System

Users must go through the user registration process before using our system. The user registration function is mainly for customers to use each function, each player must have an account. To create an account, we need users to enter their email and create their own password. Format validation checks will be applied to ensure this. The email format is similar to xxx@xxx.xxx. There is also a data validation process that asks the user to confirm the password by reentering the desired password in another field. After that, an email is sent to the user's email account to activate their account by clicking on the link provided in the confirmation email.

Users want to play the game, they will have to log in and enter an email as a username and their own password. If the user has any login problems, our system provides password change function. These users will be asked to enter their email to receive the email, and in the email they can click into a link to change their password. By submitting the new password in the link, our database updates the data in the user's password field. Users can then log in and play.

2.2.2 playing system

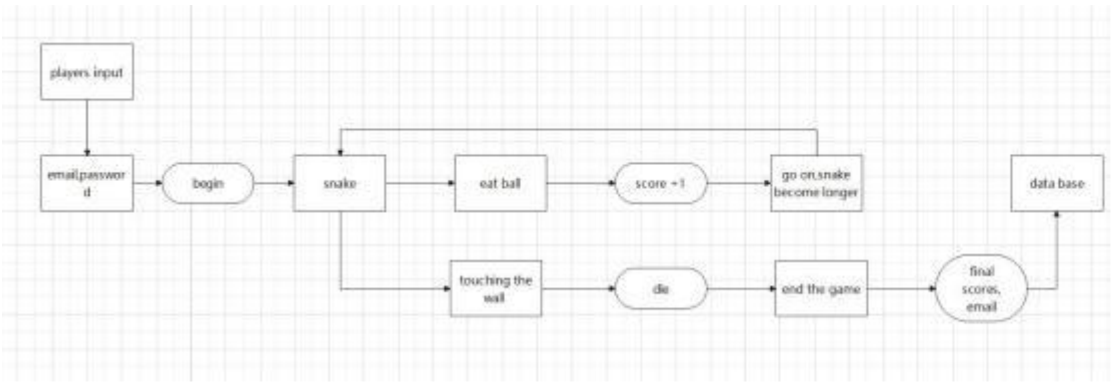


Figure 3 Data Flow Diagram: Playing system

This is a snake game, every time you eat a ball, the snake gets longer, and the game ends when you hit the wall with scores and player data uploaded to the database

2.2.3 ranking system

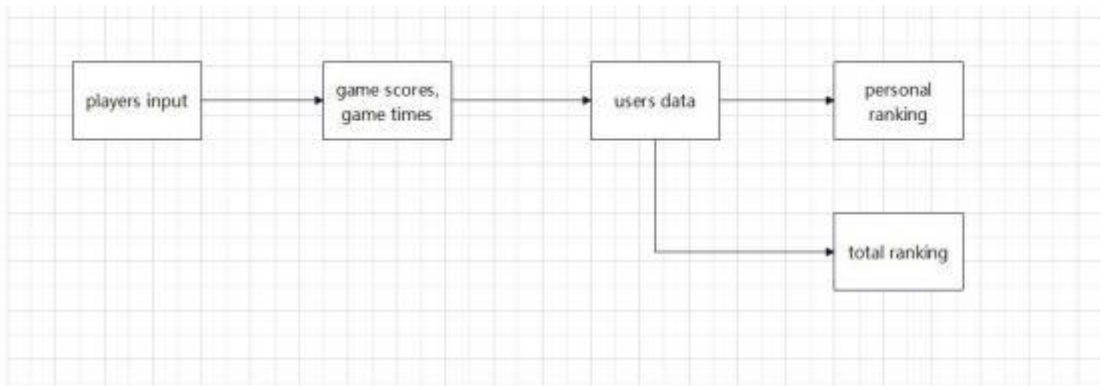


Figure 4 Data Flow Diagram: Ranking System

For the ranking system, there are two ways to rank players. One is the ranking of individual play scores.   This record is only available to players themselves, ranked by game score and playing time. The higher the score, the higher the ranking, and the shorter the time with the same score, the higher the ranking. For individuals, only the top ten scores are recorded.   The other ranking is the overall ranking, using the same rules as above, but based on the records of all players in the database.

3 DETAILED DESCRIPTION OF COMPONENTS by UML

3.1 System Components

3.1.1Fr ont-end Development

Our expected product will be a web game, composed of a html, CSS and Javascript files. We also plan to use the Bootstrap framework to assist us with the development of our website. Players can use their keyboards to play the game, we also provide many advanced options to make it more interesting, including speedup option, skins, easy/hard mode. We expect the user interface to be user friendly.

3.1.2Functions

As stated in part 3 of this design document, there are three major functions:

User Login and Registration

This function requires a database to store the input information from the users. Customers account can be freely registered using any popular email account, including QQ, 163, Outlook, Google and so on. Email will be sent automatically to new customers in order to do confirmation of the registration process.

Game Playing

This function is the main part of our system. The player will act like a snake, which aims to eat as much food as possible. Every time the snake managed to get a food, its length will increase by one and a new food will occur at a random position on the screen. By pressing the speedup key, the snake will move much faster. By pressing the hard mode key, the directions control will be inverted. By pressing the skin mode key, the path that the snake has passed will be colored. Player can stop the game at any time he wants and continue to play freely, with all things reserved and no need to restart a new game. If the snake eats its own body, the game is over and the score player gets will be recorded.

Ranking and Recording

This part does the ranking and recording part of our game. Every time a player finishes a game, the time and score he got will be recorded in the backup database, in order to let players check at any time. We also have a ranking list on the home page of our website, showing the top100 players all over the world and their scores.

If the player got a high score, his name may be shown on it. We update the ranking list every day, by searching the backup database.

3.1.3 Middleware and Communications

As mentioned before, it is obvious that all the functions are highly related to the usage of database. In order to show the data appropriately and efficiently, we plan to adapt a way for the front-end to communicate with the back-end. We plan to use the GET and POST methods of Ajax to do the request and respond procedure, so that we can pass data from the user interface to the databases and vice versa.

3.1.4 Backend Development

The backend part mainly focuses on the management of the database. In our initial design, we are planning to have around two tables, namely players and game records. As for the database we are now considering using one of the following two:

Redis

It offers a framework for building in memory application that is at the same time versatile and scalable. By using RAM as the memory, Redis speeds up the query of data. Moreover, Redis is a multi-utility tool and can be used in use cases such as caching, messaging-queues, any short-lived data in your application (e.g. web application sessions, web page hit counts).

MySQL

It is a secure and reliable database management system. MySQL has high performance designed to meet the demanding applications while ensuring optimum speed, full-text indexes and unique memory cache.

3.2 Description of Major System Components by UML

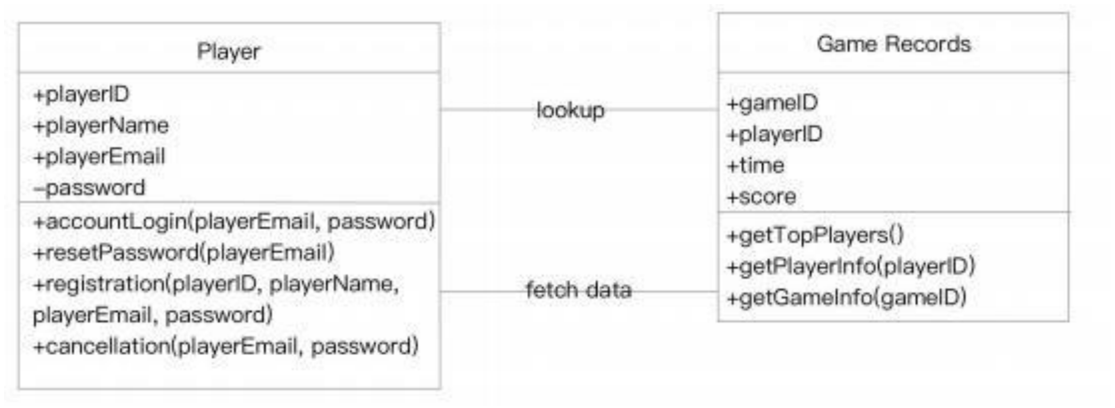


Figure 5 UML Graph of Database

The above class diagram shows the relations between the classes in our software development design process. It is also basically all the classes we plan to have in the database. The player table stores all the basic information of players. When a player registers for an account for the first time, he will be assigned a unique player ID. Also, he needs to specify a name, password and provide his email address. If he forgets his password, he can reset the password by sending an email using the email address provided when registering the account.

As for the game records table, each time a player finished a game, our system will automatically record it by storing the player’s ID, the time and the score. One record will also be assigned a unique game ID, which is used for later analysis. By calling the getTopPlayer() function, we will get the top n players with the highest scores. If a player wants to see his historical data, he can look up to the game records table. Also, game records table can fetch some data form player table, like the player’s basic information.

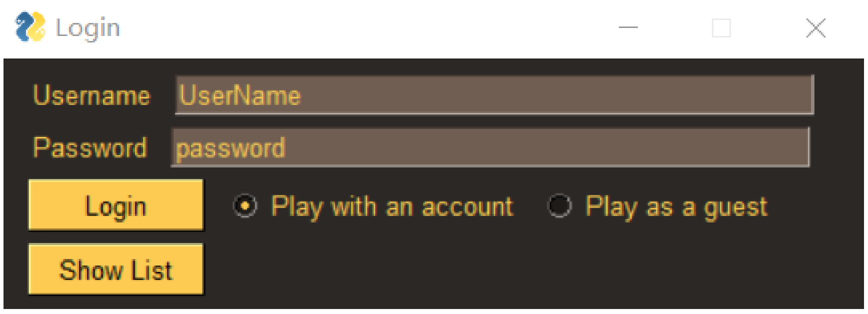
4 USER INTERFACE DESIGN

4.1Description of the User Interface

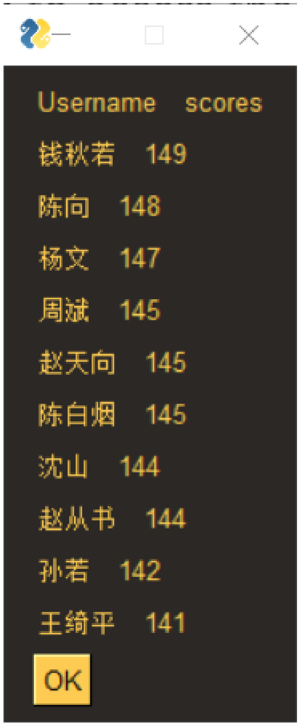
There is a GUI for users. The user interface is the login interface of the snake game. In this interface, the user can use their accounts to access the game, or directly access as a guest if they do not have an account. The data of the accounts are stored in a database, and will be transferred to the program for verification. Additionally, the database also stored the highscores of users, and in the UI, the user can access the topscore list, which shows ten of the highest data.

4.2Screen Images

4.2.1Login Screen

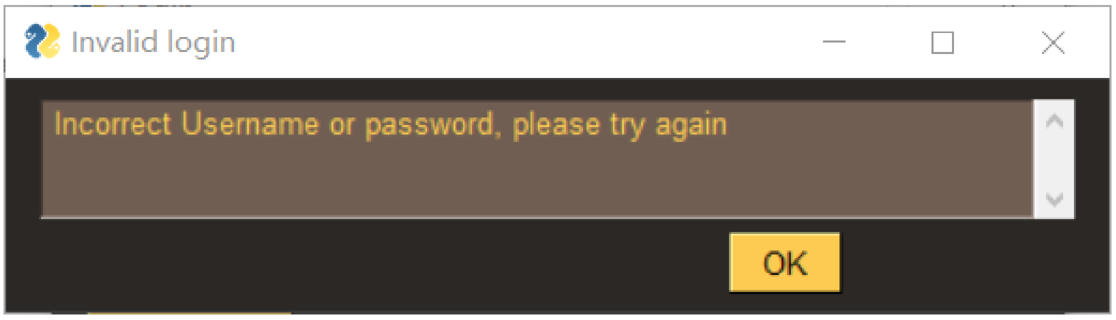


4.2.2Highscore Screen



Username	scores
钱秋若	149
陈向	148
杨文	147
周斌	145
赵天向	145
陈白烟	145
沈山	144
赵从书	144
孙若	142
王绮平	141

4.2.3Screen of incorrect password



Incorrect Username or password, please try again

4.3 Objects and Actions

In the program, the GUI will be created with login screen. On logging in with account or requesting highscore list, the program will connect to the database. After using SQL to retrieve data, for highscore list, the data will be directly shown, just as the screen image. For password verification, if the password is right, the snake game will be triggered. For logging in as guest, the database will not be accessed and the game will be directly triggered.

5.Test

For the test part of our program, we have two test procedure. One is a black test, and another is a white test, in which we verify the program by checking the data flow path of a testing variable "test_V"

5.1 Black Box testing

In the black box testing, we used a simple input-output test to validate the running result of the test program. The test of the project is divided into two parts, which is the test of the login system and the test of the webpage game.

For the test of the login system, we used 4 different kinds of input to verify the system. The inputs are list below:

1) Right Username + Right Password with User login
2) Right Username+ Wrong Password with User login
3) Wrong Username + Right Password with User login
4) Wrong Username + Wrong Password with User login
5) Right Username + Right Password with Guest login
6) Right Username + Wrong Password with Guest login
7) Wrong Username + Right Password with Guest login
8) Wrong Username + Wrong Password with Guest login

5.2 White Box tesging

In the white box testing, we defined a testing variable “test_V”, which is used to check the login process of the program. At first, the variable is initialized as “None”, and in the process of login, it will become ‘G’(represents Guest) or ‘U’(represents User) based on its login mode. After the login, if show_list function is used, the program will append an ‘s’ at the end of the variable. Then the program will detect the starting and ending of the webpage game, and adding ‘B’ and ‘E’ at the end of the variable. For example, if the final output of the “test_V” is GSBE, means the process of “login with guest and show list, then play the games and close the game” is correctly conducted. Therefore, we just need to check if “GSBE”, “GBE”, “USBE” and “UBE” can be conducted correctly to validate the program.

6.Lessons Learned

6.1 Liu Junda 118010184

In this project, I have learned many things that I never learned before. In the past three years, my interest was on data science and machine learning. I participated in some really exciting research in our university and worked in a company as an intern to do machine learning. However, I knew very little about the so-called ‘front end’. In this project, I learned a lot of things in the area. We construct a web-page using html and css. We build a web game using javascript and we use tools like ajax to connect the ‘front-end’ and ‘back-end’. I was always curious about how a web-page can be built and how those facinating functions can be implemented. This time, this question is finally answered. Actually I think I can apply my knowledge in machine leaning to our project, but I had not time because I was busy doing my intern and preparing for my graduate study. Anyway, this is a great job to be done in the future!

6.2 Zhao Changhao 118010435

In this project, I learned how to write small games arranged on the web page. In the previous years, I mainly focused on low-level code and machine learning, so I did not learn this knowledge. In this project, I learned how to write JavaScript. Although I encountered many difficulties in the process, my team members also helped me a lot. We completed the writing of Snake game together. I have a certain foundation in UI, so I am responsible for connecting mysql in the later UI design. In this project, I learned the importance of teamwork and how to cooperate more effectively in future work

6.3 Wu Zhongyue 118010338

Learned how to code using JavaScript, test program using different methodology. How to build a program and hold a repository on Github. Building a project using different program language and assemble them in a single program is quite hard. But we learned how to use different module to assemble them. Also, I learned how to code systematically and how to code in a team. Besides, to test a program, using different method is quite important to double check the data flow and the correctness of the function and variables.

6.4 Wu dongyang 118010324

In this project, I have learned to design programs in JavaScript, and to collaborate on a holistic software project. Meanwhile, I have learned to remotely hold a repository on Github using Git. Assembling different programming languages (SQL, python, and HTML&JS) could be difficult, while we managed to merge them into a whole project. Meanwhile, compromising to others' needs is also very important.

7.Conclusion

In this project, we completed the design of a user interface and game ontology. The Snake game has the ability to change the skin, adjust the difficulty, and adjust the speed of the snake in the game. However, the speed should not be lower than the minimum speed set by difficulty. In order to improve the flow of the game, we increased the frequency of screen refreshes and decisions, which also reduced the occurrence of bugs. As a game it is fairly complete and bug-free, and it has all the features of Snake. On the UI interface, we designed a user database to record the game data and information of registered users. He doesn't need an account to play. He can also play through non-login channels, but his game data won't be recorded. Players can also query the scoreboard on the login screen to see which players have the highest score and where they rank. In this project, all our team members learned how to fix the bugs in the game efficiently and reasonably and cooperated effectively. It's a division of labor but everybody has some sense of where the other people are going. We think we have completed a successful project.