**HONG KONG COMMUNITY COLLEGE**

**THE HONG KONG POLYTECHNIC UNIVERSITY**

**CC3206 Programming Project**

Final Report Specification

Lecturer Name: Dr. Simon Wong

Class: 203

|  |  |
| --- | --- |
| **Student Name** | **Student Number** |
| Chan You Zhi Eugene | 11036677A |
| Fong Chi Fai | 11058147A |
| So Chun Kit | 11048455A |
| So Tik Hang | 11030753A |
| Wong Ka Wai | 11038591A |
| Yeung Chi Shing | 11062622A |

Topic: Web-based “Zero Pollution” Project

Submission Date: 16-5-2013

Table of Content

1. Executive Summary 4

2. Introduction 5

3. System Analysis 6-25

3.1 Current Situation 6-7

3.2 Functional Requirements 8-9

3.3 Use Cases 10-17

3.4 Process Model 18-25

4. System Design 26-73

4.1 Revised Physical Architecture 26-27

4.2 Functionality Screen Design 28-45

4.3 System Structure 46-57

4.4 Database Design 58-66

4.5 Interfaces Samples 67-73

5. Software packages and tools used 74

6. Installation Guideline 75-82

7. Samples of source code 83-94

8. Conclusion 95

9. Appendices 96-111

9.1 Game Rule 96-99

9.2 References 100-105

9.3 Assumptions 106

9.4 Test Plan 107-111

**1. Executive Summary**

With the development of information age, web-based applications become more popular so that more developers are most likely preferred to develop web-based applications instead of traditional applications. Using web-based application will be a geat trend on software developing in the future. Thus, a web-based board game is needed to develop for promoting environmental protection.

Zero Pollution, an innovative board game developed by the students of Hong Kong Community College, mainly focus on propagating a message about awareness of protecting the environment to Hong Kong residents. Through enjoying this board game, players can learn different classification of waste. In order to have a further promotion on green sense, this project, e-Zero Pollution, is to build a web-based application for this board game. This web-based version will not only benefit to both players and publishers, but also protect the environment due to the reduction of paper and plate in the game itself.

Moreover, there are some changes on playing rules so that this board game can be more challenging. Players can find more excitement when they are playing this board game. Although there are some changes on the rules, the purpose of this game will be kept unchanged, it can still educate players to pay more attention on the environmental protection and know more about the ways of protecting our living city.

Apart from changing some playing rules, the new web-based mobile application of Zero Pollution will increase interaction between players and creativity. For example, a number of animations will be added into the application. The whole board game will be transform from 2D into 3D so that more people can be attracted to play this educational game.

**2. Introduction**

We are an Information Technology team, called Zero Group, from Hong Kong Community College which is our project sponsor. We will provide a variety of application development services during the process of building your application, including as-is system analyzing, system programming, interface designing and system maintaining.

We find that “Zero Pollution” is not promoted widely in the existing board game market because Hong Kong people do not have enough environmental awareness. Besides, the copyright of this game is reserved by Hong Kong Community College and it is designed for educating people especially teenagers in Hong Kong but not focuses on gaining profit. Since the physical version of this game is indeed well-designed, our development team can mainly focus on the system programming in design phrase in order to build a comprehensive web-based application for this game.

In this project report, our project team has provided some documents which include system analysis, system design, installation guideline, and software list. It contains thorough descriptions of different modules that were to comprise the new system. We have also made a number of test cases so as to investigate the problems of existing version and provide a number of ways to develop and ameliorate a better web-based version of this game. In the later section of report, you can review the part of system design. A number of screen captures will be inserted for reference. Different suggestion about improvement on web-based version can be given to our project team in order to make it be more attractive.

It is hoped that our report can let your project team have an all-round and deeper understanding on the forthcoming new game. Also, we hope that our suggestion will be beneficial to the existing version of this game and the message of environmental awareness will be widely spread to the public.

**3. System Analysis**

**3.1 Current Situation**

**3.1.1 Difficulty of promoting game**

Existing version of board game generates a variety of elements which limit the distribution coverage and lacks community group because it is just print out in paper form, low cost method. A popular product usually needs high promoting cost and high distribution on the Internet. Moreover, promotion through the Internet will attract more potential players and even attract them to play the game for a long time. They can also build up a community through chatting in chat room which will be designed in web-based version. More topics about Zero Pollution can be discussed in the public. Therefore, it is difficult to make the game become a well-known board game if it continues to use a low cost promoting method.

**3.1.2 Mobility**

Most people rarely prefer to bring a number of entertainments during outdooring activities. Nowadays, most people are willing to bring a mobile smartphone with them instead of bringing a box of card game. It is because many entertainment applications can be found in their smartphones on the web. Thus, developing a web-based version for this game might bring convenience for the players. Players can access and enjoy the game anytime at anyplace over the Internet.

**3.1.3 Extendibility**

A sustainable development is a crucial process to maintain its lift cycle. For the exciting version board games, updating the current game elements obtains a higher cost. Processes such as producing a new physical board game, redistribution of the board game and even re-promotion of the new version will make an extremely high cost for the company, and thus it will break sustainable development processes for the game. However, after the development of web-based version, it is easy to update game elements like modification of the design, new rules, others game experience, and you named it. As a result, a cheaper and quicker updating process can be executed in sustainable development of the game.

**3.1.4 Easy to lose physical cards**

In fact, children aged above 6 are one of our target players. In normal experience, they often lose their toys’ components due to their careless mistakes. There is a similar case for other exciting board games. Players might lose a few components easily due to their careless mistakes. This generates a great inconvenience for players and our organization as we may have a possible chance to produce a set of separated components to them, or perhaps, for a worsen case, they may need to purchase a new board game which is not environmental friendly and run in the opposite direction of the original objective of the game.

**3.1.5 Lack of “green message”**

The main purpose of the game is to educate the player to increase the awareness of protecting environment. However, the existing version of board game provides lack of environmental protection knowledge and it is difficult to update and modify the knowledge of protecting environment. After the development of web-based version, it can add the knowledge of environmental protection and update the knowledge frequently to ensure the knowledge which is close to the life.

3

**3.2 Functional Requirements**

**3.2.1 Maintaining the rules of the original board game Zero Pollution**

The rules of the original board game should be followed in our Web-based version. Although, we have decided to make a few changes in the Web-based version in order to make it more playable on digital devices, the whole gaming system is still based on the original design. All different types of cards from the origin design should be maintained.

**3.2.2 Maintaining the graphic design of the original board game Zero Pollution**

The Web-based version of Zero Pollution should use the same graphic design from the board version, so that the players won’t confused while playing between different versions and be more familiar on them.

**3.2.3 Providing Artificial Intelligence**

Artificial Intelligence should be designed in order to let user play with computer players. The AI should be smart enough to let the user feel difficulties.

**3.2.4 Allowing online multi-player mode**

The game should allow player to play online with other players, so that user can play with their friends or player around the world via Internet. Online server may be built to satisfy this requirement.

**3.2.5 Providing player account system**

Player account system should be provided to store user profile and information, for example their playing history and user name. Users should login their own account to play online.

**3.2.6 Accessibility**

The game should be a web-based program which is accessible by both desktops and mobile devices with Internet browsers.

**3.2.7 Extensibility**

The game should be well-programed and extendable to update version, or even supporting other platforms in the future.

**3.2.8 More playable than the original board game**

The game should have more elements to make it itself more exciting. However the Web version game rule should be based on rules of the original board game, and should be suitable and available for players above the age of 6.

**3.2.9 Privacy**

The personal information we collected from the account system should be well protected to preventing data leaking and privacy problem.

**3.2.10 Compatibility**

The game should be playable by any computer with 3 major browsers: Microsoft Internet Explorer, Mozilla FireFox and Google Chrome, including those with older hardware. Ideally, it should also be accessed from mobile devices with the iOS, Android or Windows Phone that is recently released. This means that it should not consume too much system resources.

**3.2.11 Performance / Response time**

The program performance should be acceptable by most of the user, which means that the response time of the game for every function generally fast, let say within 5 seconds.

**3.2.12 Error Control / Robustness**

The game server should be accessible by the system admin at any time, so that it can be recovered immediately after server error occurs.

**3.3 Use Cases**

**How to view Use Cases diagram?**

|  |  |
| --- | --- |
| **Element of use cases** | **Description** |
| Use case name | The name of the use case name |
| ID | The number of the use case |
| Importance level | The relative significance of the use case |
| Primary Actor | The external user, organization, external IS system, or device that trigger an event to which the system responds |
| Trigger | The event that causes the use case to begin |
| Description | Briefly explanation of the use case |
| Type | External : from outside the system  Temporal :time-based occurrences |
| Major Input Description | Each major input described with their sources. |
| Major Output Description | Eachmajor output described with their destinations. |
| Major Steps | |  | | --- | | The major steps needed to process the inputs and produce the outputs. | |
| Information of the steps | Identify its triggers and its inputs and outputs |
|  | Data flow (Source to Destination) |
|  | Data flow (Destination to Source) |

**Table 3.3 – Instruction to view Use Cases diagram**

**3.3.1 Login to the game**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name:** Login to the game | | | **ID Number:** 1 |
| **Primary Actor:** Player | | | |
| **Short Description:** This use case describe the process of a players login to the system | | | |
| **Trigger:** Player opens the game | | | |
| **Type:** External / ~~Temporal~~ | | | |
| **Major Inputs:** | | **Major Outputs:** | |
| **Description** | **Source** | **Description** | **Source** |
| “Open game” button | Player | Register page | Player |
| Account information | Database | “Main Menu” Page | Player |
|  |  | Account information | Database |
| **Major Steps Performed** | | **Information for Steps** | |
| 1. Open the game  If not register before  2. Click "Register" button  If Register already  3. Type the user name and password and click "Login" button  4 Locate the player information in database  5. Import player information into the game | | “Open game” button  “Register” page  “Main menu” page  Account information  Account information | |

**Table 3.3.1 - Login to the game**

**3.3.2 Start a new game**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name:** Start a new game | | | **ID Number:** 2 |
| **Primary Actor:** Player | | | |
| **Short Description:** This use case describes the process of players starting a new game | | | |
| **Trigger:** Player enter to main menu | | | |
| **Type:** External / ~~Temporal~~ | | | |
| **Major Inputs:** | | **Major Outputs:** | |
| **Description** | **Source** | **Description** | **Source** |
| “Login” button | Player | Single Player | Player |
|  |  | Multi Player | Player |
|  |  | Current game information | System |
|  |  | Match Information | System |
| **Major Steps Performed** | | **Information for Steps** | |
| 1. Enter to main menu  If start a single player game  2. Click "Single Player" button  If start a multiplayer game  3. Click "Multi Player" button  4 Record all player's step and action  5. Update match information in database | | “Login” button  “Single Player” page  “Multi Player” page  Current game information  Match Information | |

**Table 3.3.2 - Start a new game**

**3.3.3 View and edit game setting**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name:** View and edit game setting | | | **ID Number:** 3 |
| **Primary Actor:** Player | | | |
| **Short Description:** This use case describe the process of how players view or edit game setting | | | |
| **Trigger:** Player clicks “Setting” button | | | |
| **Type:** External / ~~Temporal~~ | | | |
| **Major Inputs:** | | **Major Outputs:** | |
| **Description** | **Source** | **Description** | **Source** |
| "Setting” button | Player | Pervious page | Player |
|  |  | “Game main menu” page | Player |
|  |  | “Main menu” page | Player |
|  |  | “Profile” page | Player |
|  |  | “Chat Room” page | Player |
|  |  | “Game Rules” page | Player |
|  |  | “About” page | Player |
|  |  | “Credit” page | Player |
|  |  | Music on/off | Player |
|  |  | “Confirm Logout” window | Player |
| **Major Steps Performed** | | **Information for Steps** | |
| 1. Enter to Setting menu  If want to start a New game  2. Click "New game" button  If go back to main menu  3. Click “Home" button  If want to open "Profile" page  4. Click “ Profile" button  If want to open "Chat room "page  5. Click "Chat Room" button  If want to open "Game Rule" page  6. Click "Game Rules" button  If want to open "About" page  7. Click "About" button  If want to view Credit  8. Click “ Credit” button    If want to turn music ON/OFF  9. Click “Music ON/OFF” button  If want to logout  10. Click “Logout” button | | "Setting “button  "Game main menu" page  “Main menu” page  "Profile" page  "Chat room “page  "Game Rules" pages  “About” pages  “Credit” page  Music ON/OFF  “Confirm Logout“ window | |

**Table 3.3.3 - View and edit game setting**

**3.3.4 Add other to friend list**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name:** Register a new account | | | **ID Number:** 4 |
| **Primary Actor:** Player | | | |
| **Short Description:** This use case describes the process of new players register their account | | | |
| **Trigger:** Player press “Register” button | | | |
| **Type:** External / ~~Temporal~~ | | | |
| **Major Inputs:** | | **Major Outputs:** | |
| **Description** | **Source** | **Description** | **Source** |
| “Login” button | Player | Personal information | System |
| Checking | System |  |  |
| Error Massage | System |  |  |
| **Major Steps Performed** | | **Information for Steps** | |
| 1. Enter to “Register page”  If it is not valid    2. Enter the player name  If it is valid  3. Enter the password  If it is valid  4. Re-enter the password  If it is valid  5. Enter the password  If it is valid  6. Enter the email address  If it is valid, provide enough information  7. Click “ Send”    8. Error massage displayed and need to re-type  9. If press “clear”, all data are cleared | | "Register “button  System Checking  System Checking  System Checking    System Checking  System Checking  Personal information  Error Massage | |

**Table 3.3.4 - Add other to friend list**

**Table 3.3.4**

**3.3.5 Play Single-Player Game**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case Name:** Play Single-Player Game | | | **ID Number:** 5 | |
| **Primary Actor:** Player | | | | |
| **Short Description:** This use case describe the process of a players play single-player game | | | | |
| **Trigger:** Player press “Single- player” | | | | |
| **Type:** External / ~~Temporal~~ | | | | |
| **Major Inputs:** | | **Major Outputs:** | | |
| **Description** | **Source** | **Description** | | **Source** |
| “Single player” button | Player | “Single player menu” page | | Player |
| Game information | Database |  | |  |
| Current Game information | Database |  | |  |
| **Major Steps Performed** | | **Information for Steps** | | |
| 1. Open the ” Single player menu” page  2. Press “New Game”  3. Game start  4. When players playing the game  5. Game end. | | "Single-player “button    Game information  Current game information  Quite the Use cases | | |

**Table 3.3.5 - Play Single-Player Game**

**Table 3.3.4**

**3.3.6 Play Multi-Player Game**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case Name:** Play Multi-Player Game | | | **ID Number:** 6 | |
| **Primary Actor:** Player | | | | |
| **Short Description:** This use case describe the process of a players play multi-player game | | | | |
| **Trigger:** Player press “Multi- player” | | | | |
| **Type:** External / ~~Temporal~~ | | | | |
| **Major Inputs:** | | **Major Outputs:** | | |
| **Description** | **Source** | **Description** | | **Source** |
| “Multi player” button | Player | Match information | | Database |
| Game information | Database | Main page | | Player |
| Current Game information | Database |  | |  |
| **Major Steps Performed** | | **Information for Steps** | | |
| 1. Open the ” Select table” page  If choose the table that have no players  2. Become host of that table  If choose the table that have no host  3. Become participant of that table  4. Host press” Start the game”  5 When players playing the game  6. Game end. Player quite the game | | "Multi-player “button    Game information  Current game information  Match information  Main page | | |

**Table 3.3.6 - Play Multi-Player Game**

**Table 3.3.4**

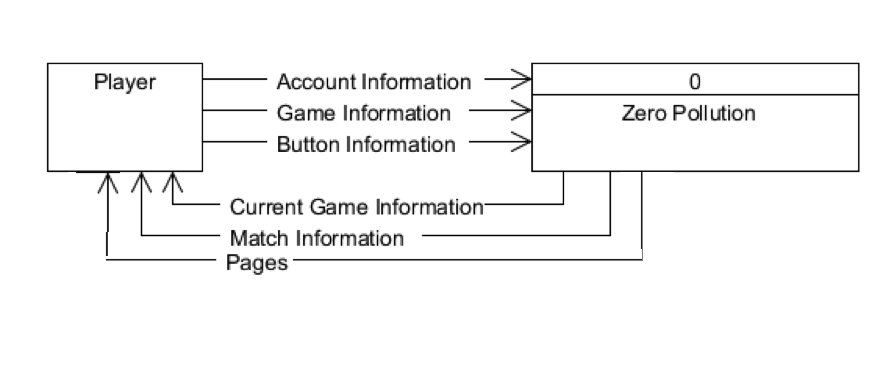
**3.4 Process Model**

**How to view data flow diagram?**

|  |  |  |
| --- | --- | --- |
| **Elements** | **Elements Name** | **Description** |
|  | Process | An activity or function performed for a specific business reason. Including the number of the process and the name of the process |
|  | Data flow | A single piece of data or a logical collection of data. Always starts or ends at a process. |
|  | Data store | A collection of data that is stored in some way. Data flowing out retrieved from the data store. Data flowing in updates or is added to the data store. |
|  | External entity | A person, organization, or system that is external to the system but interacts with it. Persons who execute a process are part of the process |

**Table 3.4 – Instruction to view data flow diagram**

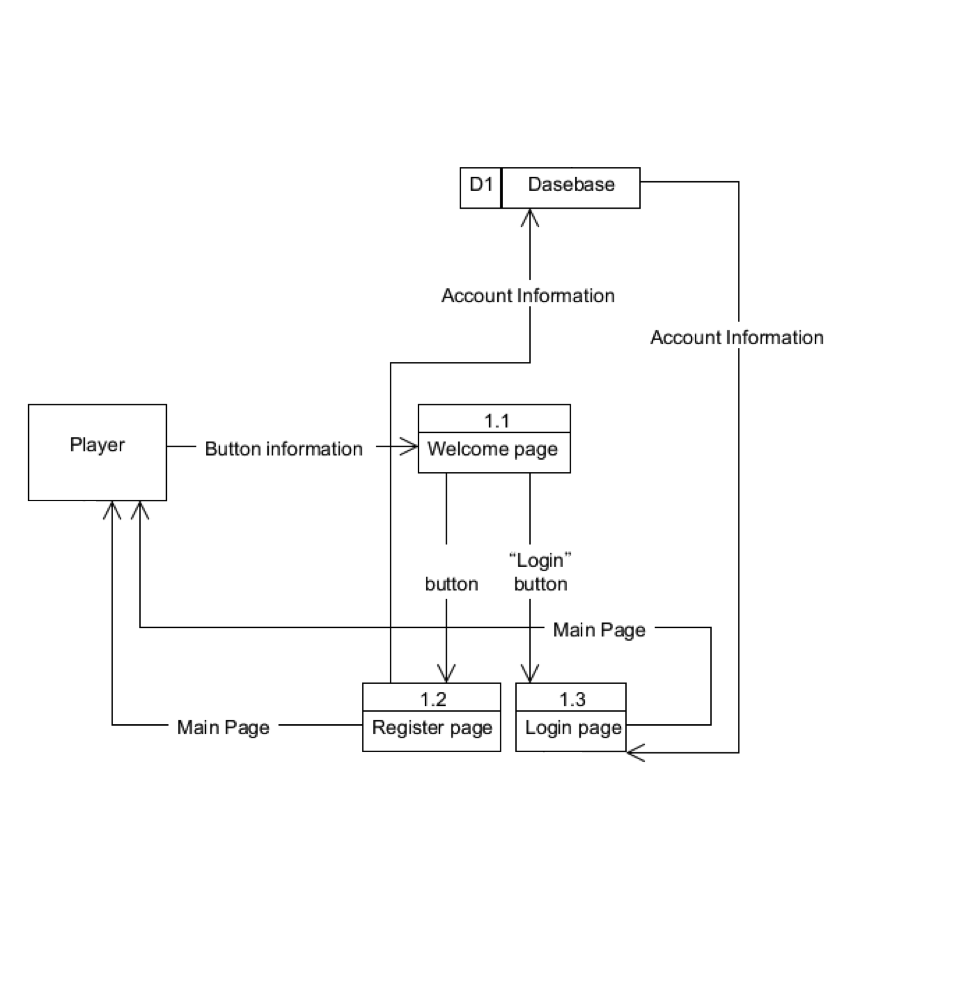
**3.4.1 Context Diagram**



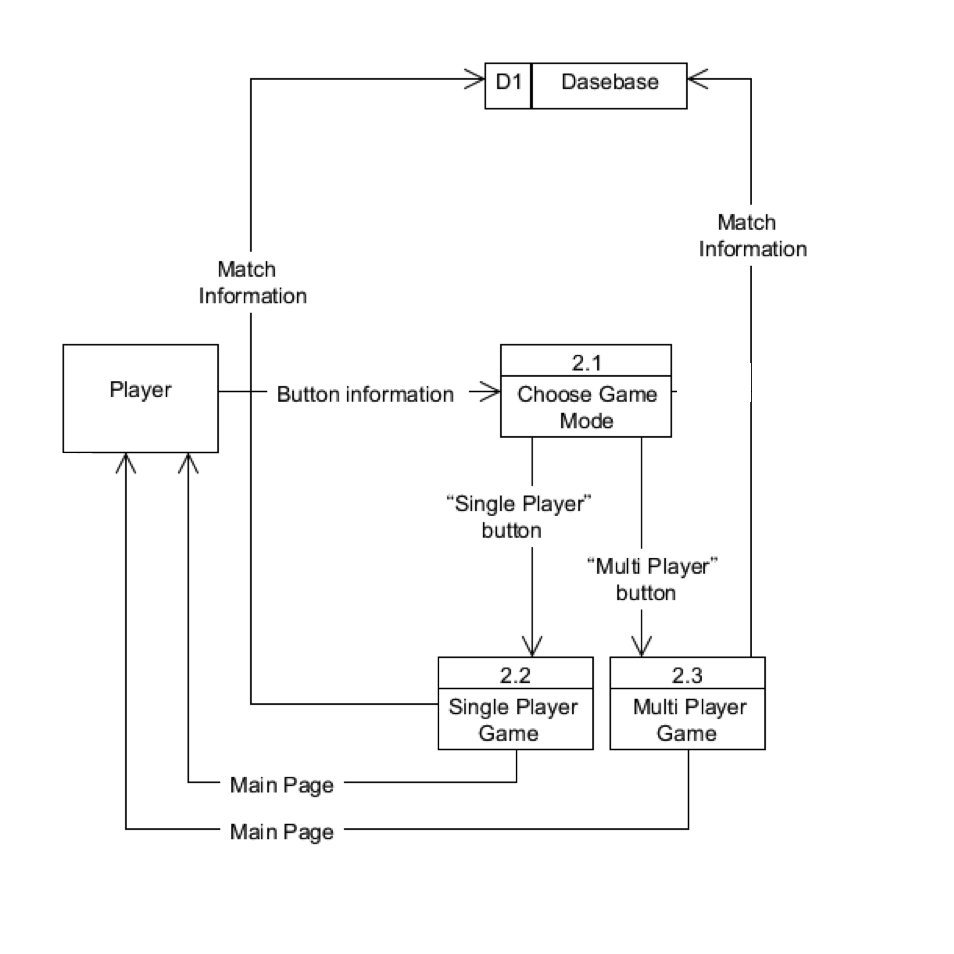
**Diagram 3.4.1 – Context Diagram**

Players provide account information, game information, and button information to the system (Zero pollution). The system will give back the current game information, match information and pages to the players.

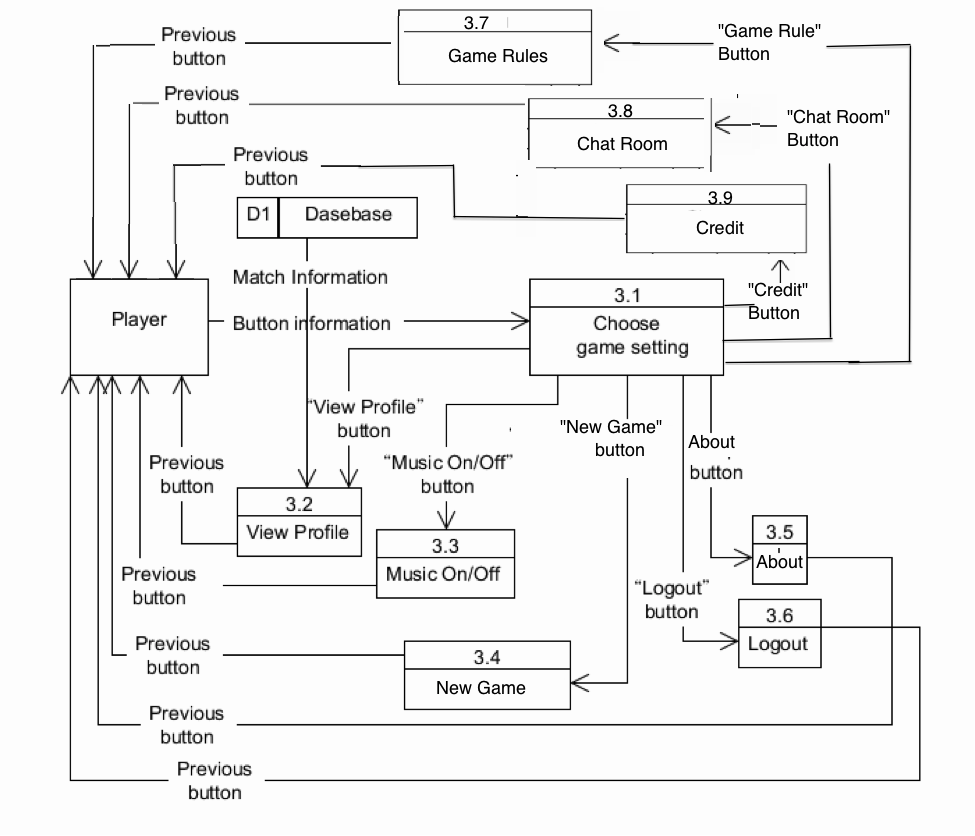
**3.4.2 Level 1 Diagram**

**Diagram 3.4.2(a) - Login to information system**

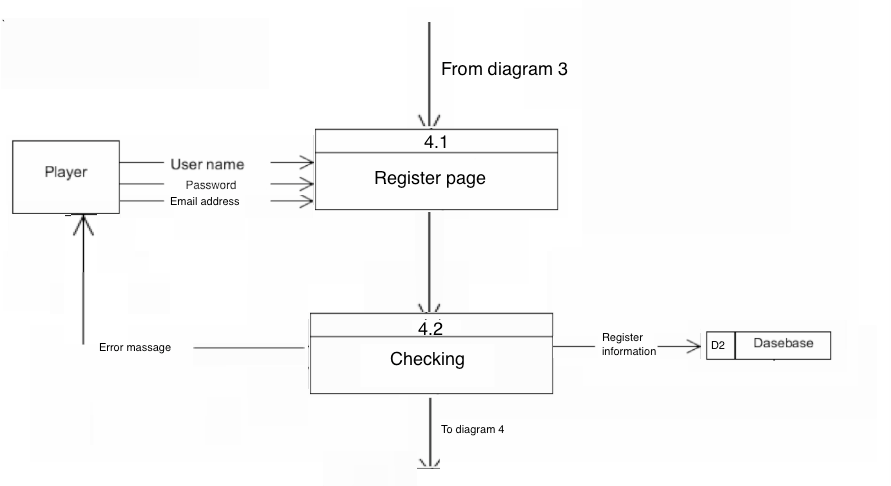
The first diagram is “Login to information system”. The primary actor is player. This use case describes the process of player's login to the system. After player opens the game, player will enter this use case. If players have not registered before, player has to click Register" button and enter "Register" page. If players have registered before, player needs to enter user name and password and click "Login" button to enter "Main menu" page. The system will locate player information in database and import player information into the game.

**  
Diagram 3.4.2(b) - Start a new game**

The second diagram is “Starting a new game”. The primary actor is player. This use case describes the process of players starting a new game. After player logins to the game, it will enter this use case and display main menu. If the player wants to start a single player game, click "Single Player" button and he/she will enter "Single Player" page. If the player wants to start a multiplayer game, click "Single Player" button to enter "Multi Player" page. System will record all players' step and action and update match information in database.

**  
Diagram 3.4.2(c) - View and edit game setting**

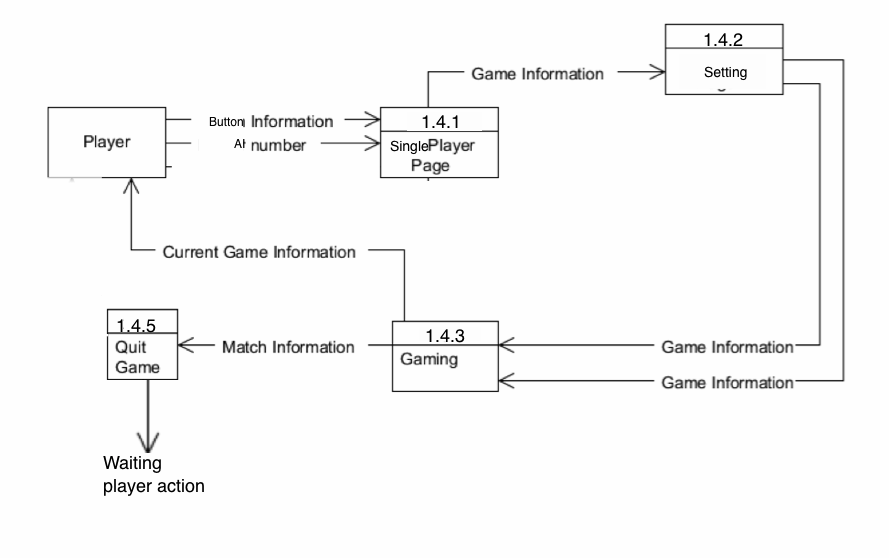
The third diagram is “View and edit game setting”. The primary actor is player. This use case describes the process of how players view or edit game setting after player press “Setting”. It will enter this use case and the Setting menu. Players have some choices to choose, including Home, New game, Profile, Chat, Room, Game, Rules, About, Credit, Music ON/OFF and Logout. Players can press the corresponding button to the page.

****

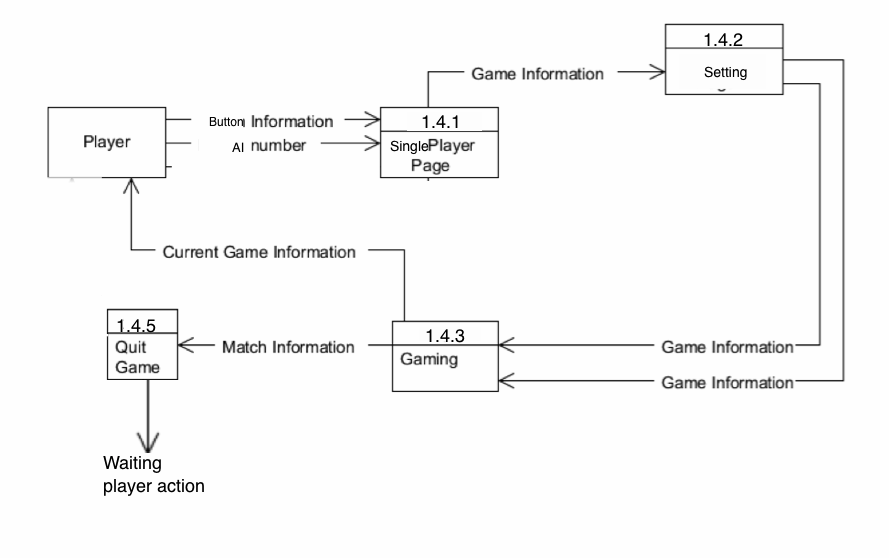
**Diagram 3.4.2(d) - Register a new account**

The forth diagram is “Register a new account”. The primary actor is player. This use case describes the process of new players register their account. When Players press “Register” button, they will enter this diagram and “Register page”. Users must provide name, password, and email address. They have some rule and the system will check the input is valid or not. The error massage will be showed immediately. After typing and the system have checked the information is valid, players can press “Send” button and all person information will be sent to the database. All data can be cleared by pressing “Clear” button.

**3.4.3 Level 2 Diagram**

**  
Diagram 3.4.3(a) - Single Player Game**

The fifth diagram is “Single-player game”. The primary actor is player. This use case describes the process of playing single-player game. When Players press "Single-player “button, they will enter this diagram and “Single player menu” page. System will send game information to the players. When players are playing a game, system keeps on tracking current game information. After the game ends, the player will quit from this use case.

**  
Diagram 3.4.3(b) - Multiplayer Game**

The last diagram is “Multi-player game”. The primary actor is player. This use case describes the process of a players play single-player. When Players press "Multi-player “button, they will enter this diagram and “Select table” page. If player chooses the table which has no players, player becomes host of that table automatically. If the player chooses the table that has no players, player becomes participant of that table. One the host presses “Start the game” button. The game starts and the system gives game information to the players. When players are playing the game, the system keeps on tracking current game information. After the game ends and player quit from the current game, match information will send back to the database for record. The players will send to “Main page”.

3

**4. System Design**

**4.1 Physical Architecture**

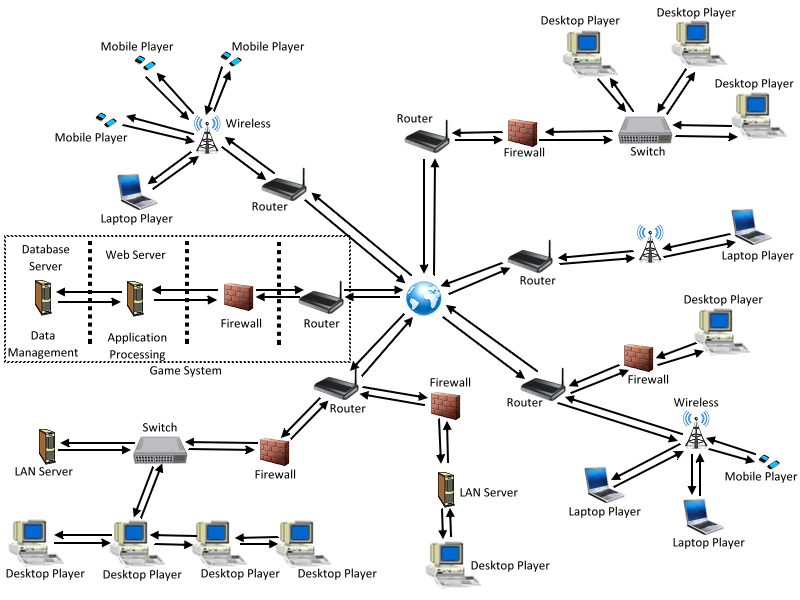
In order to ensure players have a better gaming experience, we have decided to use three- tier system architecture. It is a client-server architecture which is logically separated into three functions, including presentation, application processing, and data management, by divided into three tiers, presentation tier, data access tier and data tier. As there are 3 different separated tiers, we can upgrade or replace each tier independently.

The greatest benefit of using this system architecture is to prevent system containing any unrelated functional change when development team upgrades or replaces one function in the future.

The other benefit is that application servers can be deployed on many machines. It means that database no longer requires a connection from every client. The scalability is very high.

The drawback of the using three–tier system is lack of tools. Using fewer tools will make the process of building system much harder. Yet, we will obtain more research to overcome this shortcoming.

Connection between users and system will be set as the following diagram:

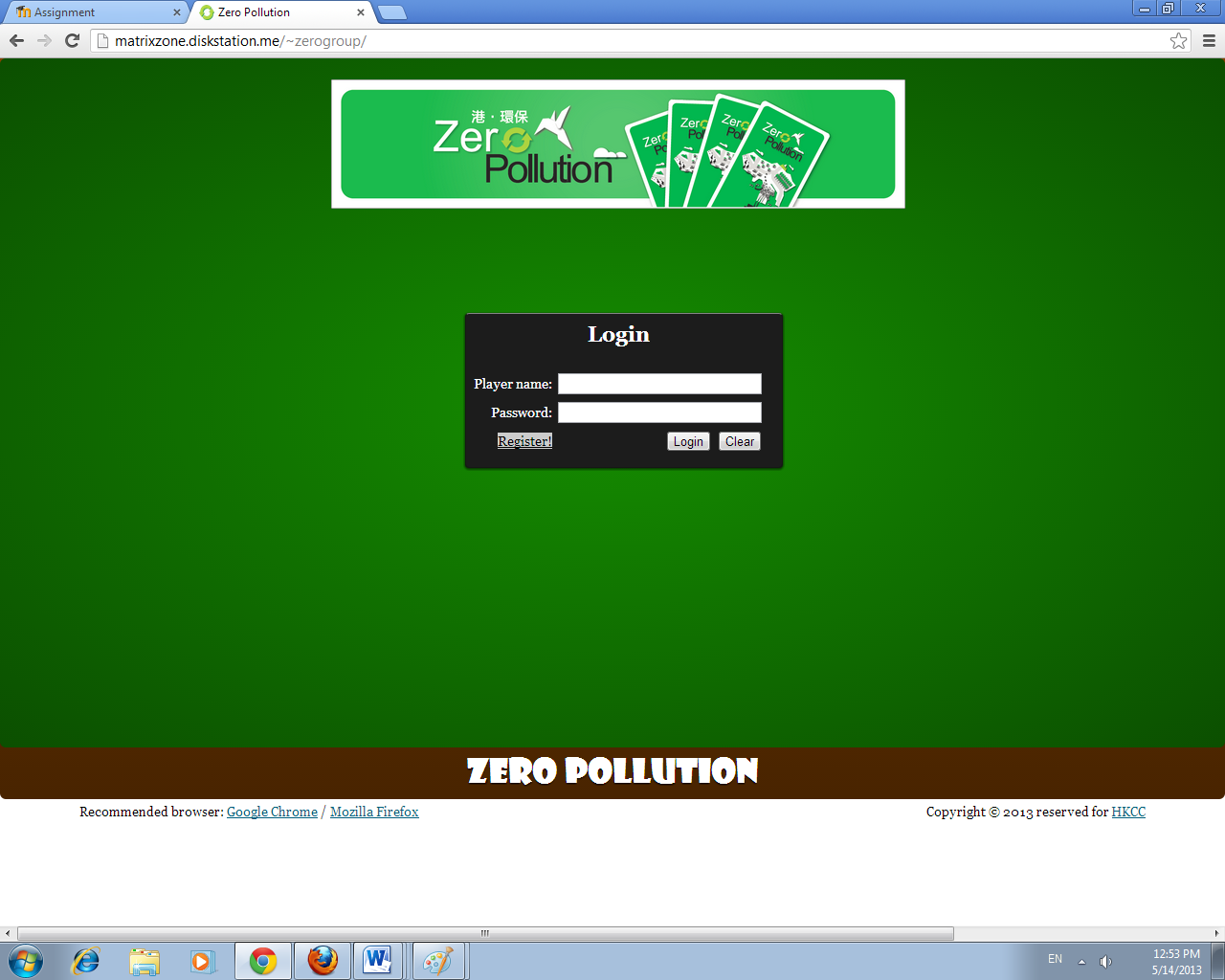


**Diagram 4.1 - Connection between users and system**

At first, users can login to the workstation by entering their own ID and password. The workstation is connected to the server though the Internet. There should be a firewall located between workstation and server to prevent any unauthorized attack. The sever will then check the user’s ID and password. If they are correct, the system will allow users access to the database.

**4.2 Functionality Screen Design**

**4.2.1 Login Screen**



1g

1a

1b

1d

1c

1e

1f

**Diagram 4.2.1(a) - Login Screen**

|  |  |
| --- | --- |
| **Name:**  Login Screen | |
| **Description:** The first page of the game system. Players are required to login before entering the game. | |
| **Button / Input / Output:** | |
| **1a. User ID Input** | Players can input their user name here to login to the system. The system would check whether user ID exists or not. |
| **1b. Password Input** | Players can input their password here to login to the system. The system would check whether the password match the user ID or not. |
| **1c. Register Button** | This button would move to create account page, which allows players to create a new account. |
| **1d. Login Button** | Players can login to the system after providing correct information (user ID and password) and clicking on it. |
| **1e. Clear Button** | This button would clear all the input placed in User ID Input (1a.) and password Input(1b) |
| **1f. Recommended Browser Button** | Mozilla Firefox and Chrome are the suggested browsers. Here is provided two hyperlink that link to their website for users download the browsers. |
| **1g. Copyright Button** | As the game is designed by HKCC, here is a hyper link that link to HKCC for users to get more information about this game. |

**Table 4.2.1(b) - Login Screen**

**4.2.2 Register screen**



2b

2h

2c

2d

2e

2f

2g

2a

2k

2j

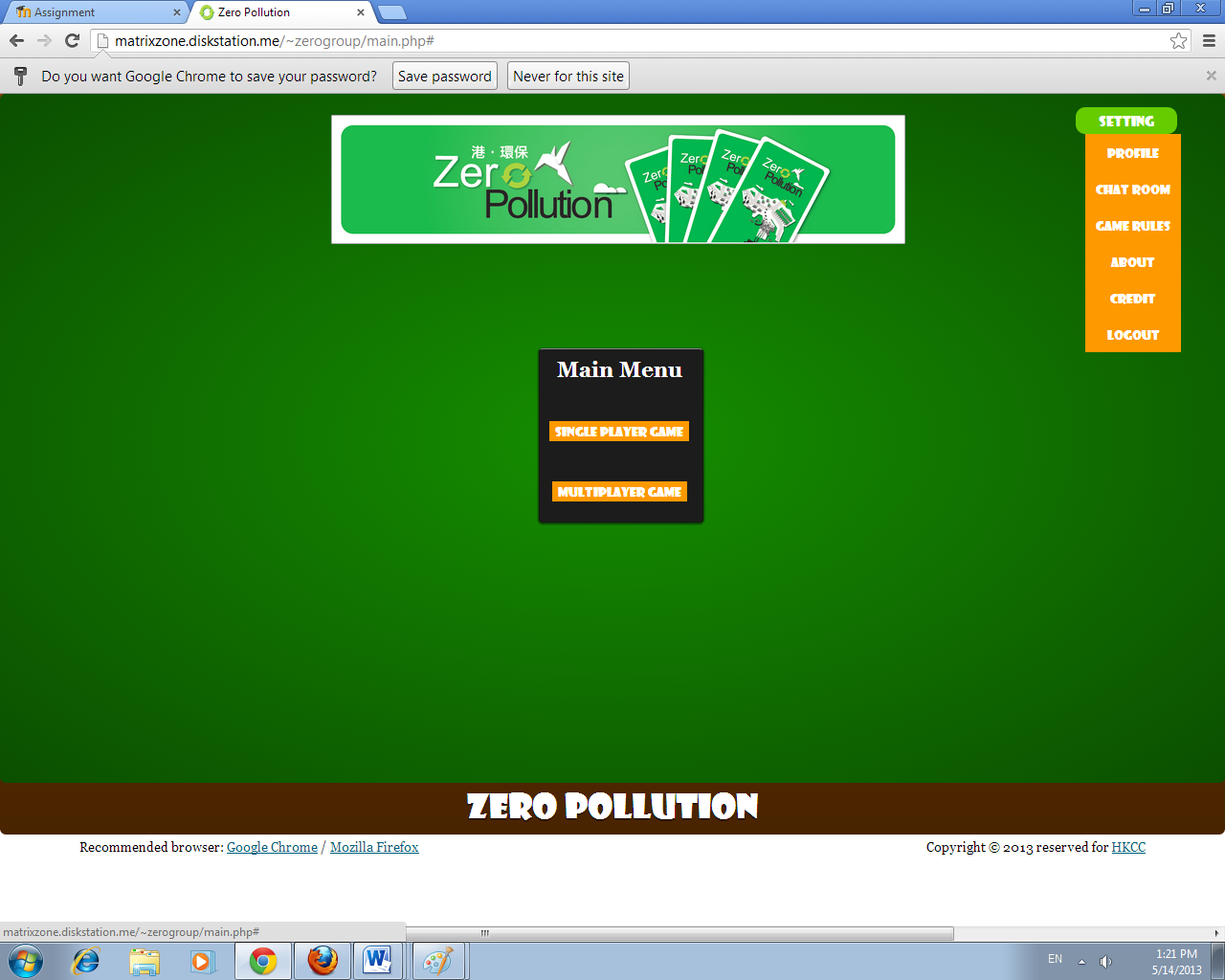
2i

**Diagram 4.2.2(a) - Register screen**

|  |  |
| --- | --- |
| **Name:** Register screen | |
| **Description:** This page allows players to create a new account in order to login to the system. | |
| **Button / Input / Output:** | |
| **2a. Back button** | Player can back to the Login Screen (1). All input will not save and the register process is not completed. |
| **2b. Player Name Input** | Players can input their login name here. The system would check whether there is same user name or not in the database, so that repeated user name is not allowed. The name should be at least 5 characters. |
| **2c. Password Input** | Players can input their own password here. They can type in alphabets (case-sensitive), numbers and symbols. The password should be at least 6 characters. |
| **2d Conform password Input** | Player can re-input the password here. The system will check the password inputted is same as Password Input (2c). |
| **2e. E-mail Input** | Players need to input their E-mail address here. The system would check whether the entered address control space and at sign or not. |
| **2f. Your real name Input** | Players can type his/her real name here. It is not a necessary process that players can choose type or not by his/her own will. |
| **2g. Phone number Input** | Players can type his/her phone number here. It is not a necessary process that players can choose type or not by his/her own will. |
| **2h. Home address Input** | Players can type his/her home address here. It is not a necessary process that players can choose type or not by his/her own will. |
| **2i. Checking Output** | The system will check the input is valid or not. The error massage will be showed immediately. |
| **2j. Submit Button** | Players can click this button to create a new account after providing valid information. |
| **2k. Clear Button** | This button would clear all the input placed in above input boxes. (2b. - 2h.) |

**Table 4.2.2(b) - Register screen**

**4.2.3 Main Menu Screen**



3b3i2i

3a2h

3c

3d

3e

3f

3g

3h

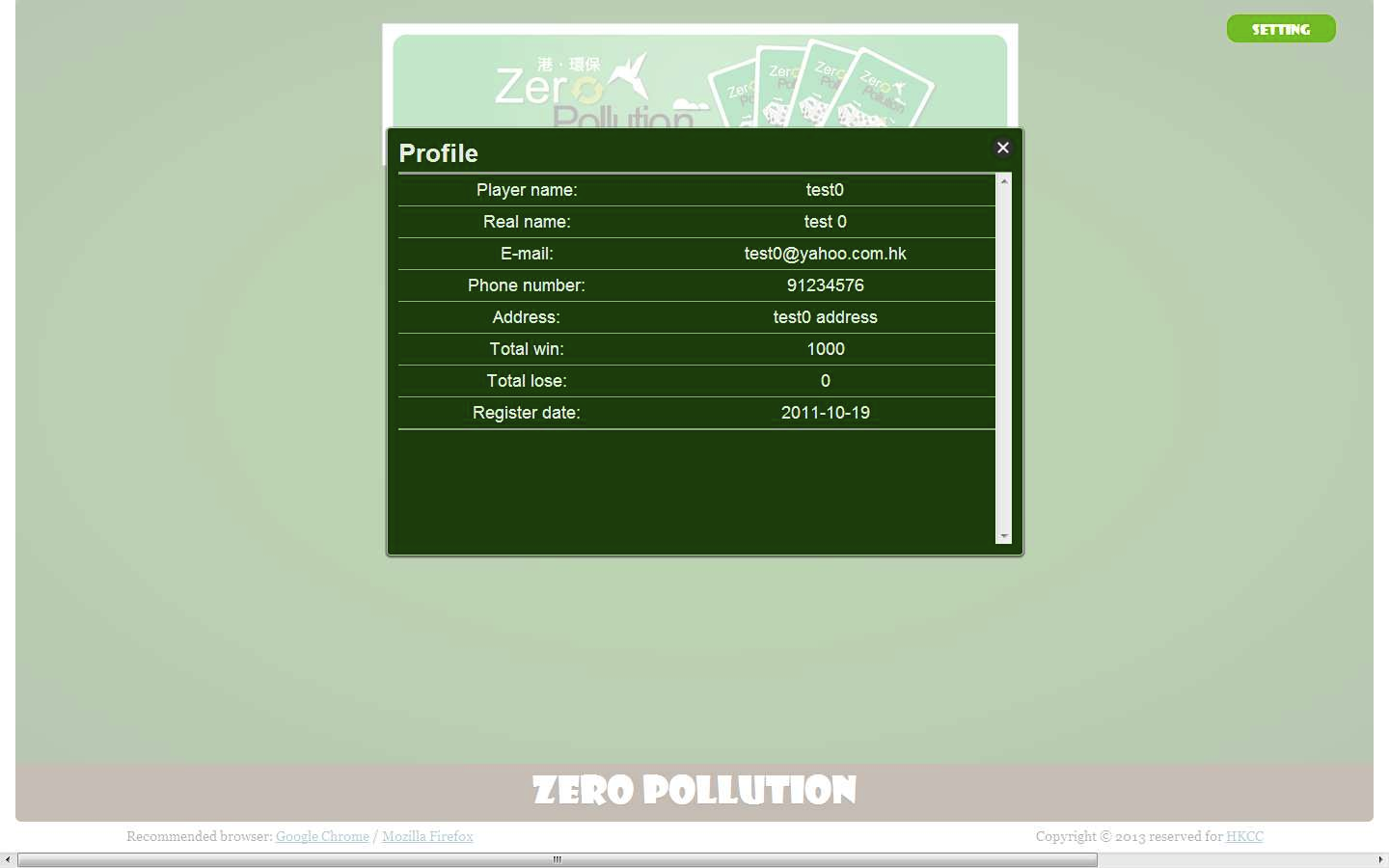
3i

**Diagram 4.2.3(a) - Main Menu Screen**

|  |  |
| --- | --- |
| **Name:** Main Menu Screen | |
| **Description:** The main page of the game system. Players can select play in single or play with the others. AI is developed for only in single player | |
| **Button / Input / Output:** | |
| **3a. Single Player Button** | Players would move to the single player page after clicking this button. They can conduct a new single game, which all opponents are played by artificial intelligence. |
| **3b. Multi Player Button** | Players would move to the multiplayer page after clicking this button. They can conduct a new multiplayer game which the opponents should have at least one human. |
| **3c. Setting Button** | Players can choose different option other than open a new game. |
| **3d. Profile Button** | Players can click it to call a pop-up window to view their own profile. |
| **3e. Chat Room Button** | Players can click it to open chat room windows to start chatting with other players. |
| **3f. Game Rules Button** | Players can view game rules after clicking this button. |
| **3g. About Button** | Players can view the information about this game after clicking this button. |
| **3h. Credit Button** | Players can view the credit about this game after clicking this button. |
| **3i. Logout Button** | Players can click to logout from the game. |

**Table 4.2.3(b) - Main Menu Screen**

**4.2.4 Profile Screen**

****

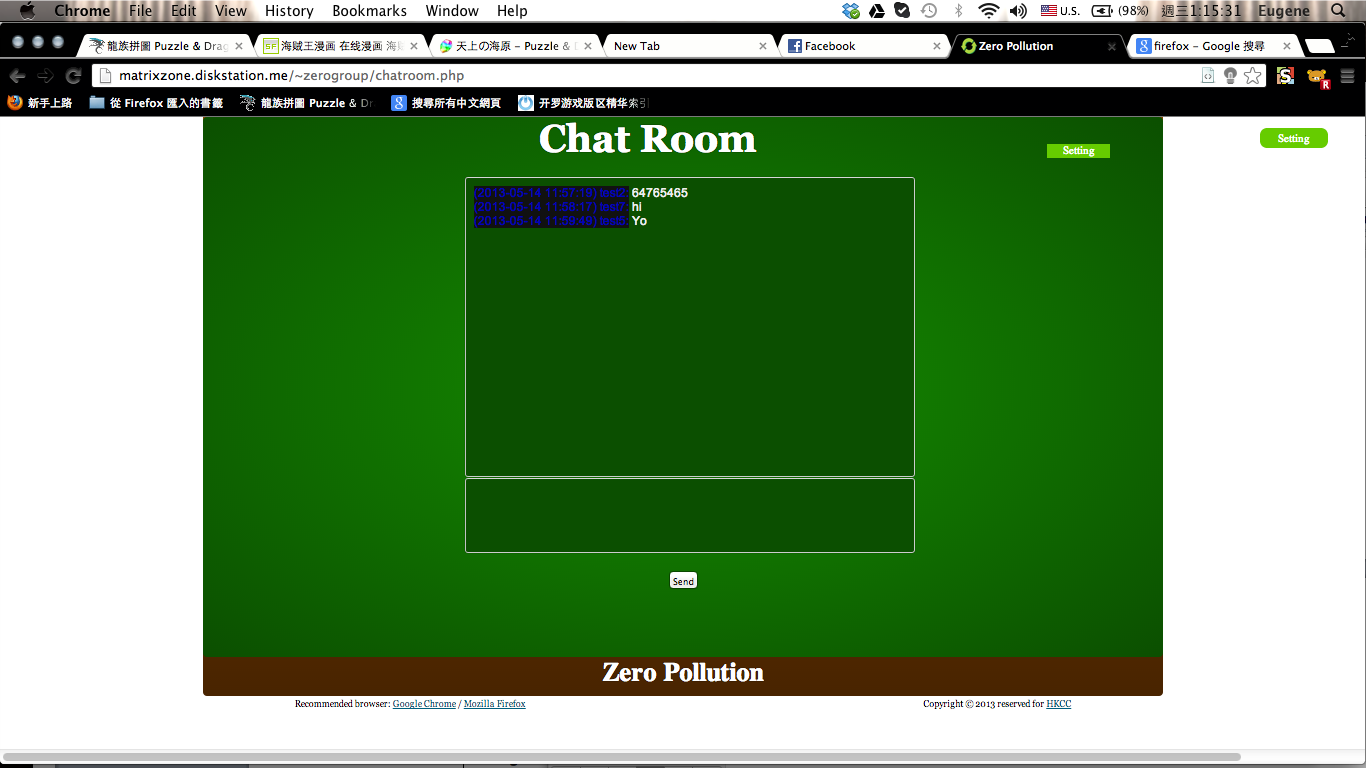
4a

**Diagram 4.2.4(a) - Profile Screen**

|  |  |
| --- | --- |
| **Name**: Profile Screen | |
| **Description:** The page of showing the user’s information, including name, real name, email address, phone number, address, total win, total lose and register date. | |
| **Button / Input / Output:** | |
| **4a. Close Button** | Players can close the window after viewing and continuous his/ her playing. |

**Table 4.2.4(b) - Profile Screen**

**4.2.5 Profile Screen**



5a4a

5b

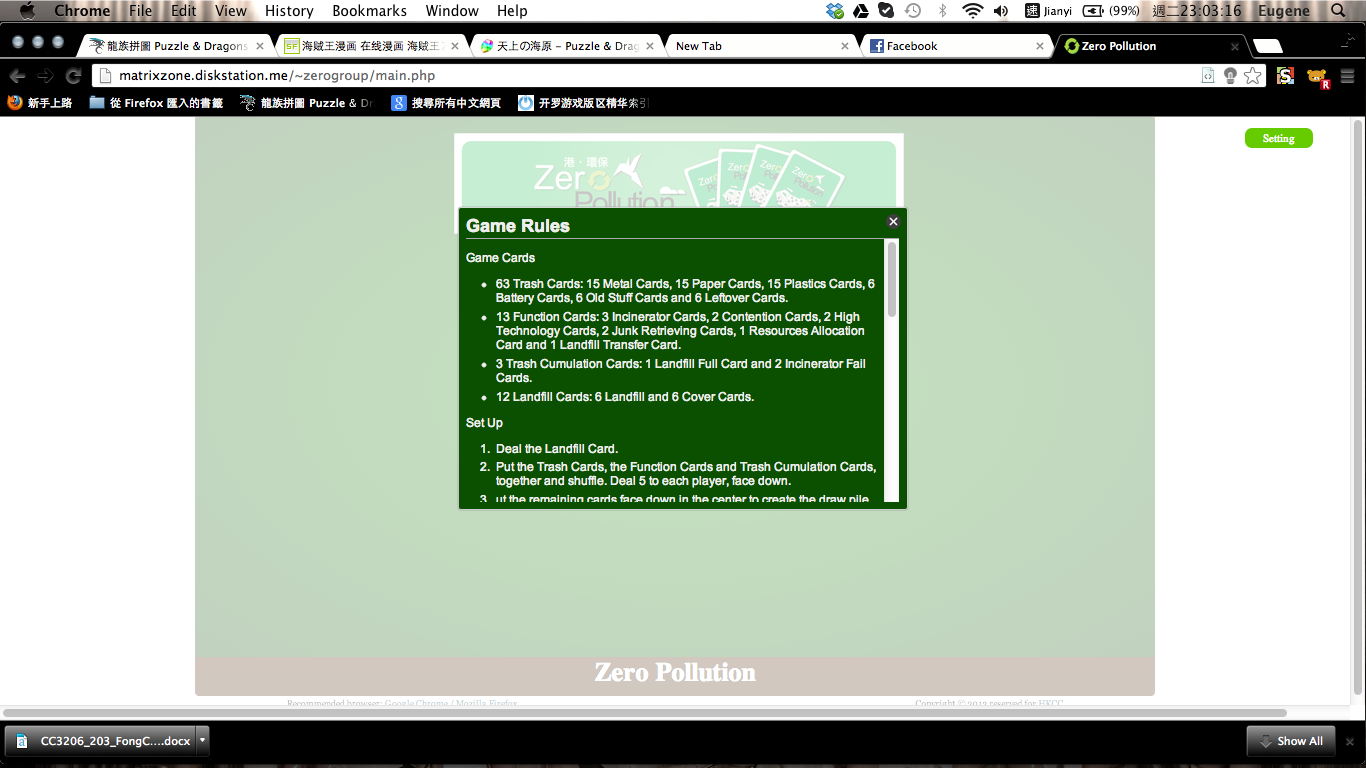
5c4a

**Diagram 4.2.5(a) - Chat room Screen**

|  |  |
| --- | --- |
| **Name:** Chat room Screen | |
| **Description:** The internal chat room for player communication will appear after players had login to the system. | |
| **Button / Input / Output:** | |
| **5a. Message** | This area would show the messages in the chat room. It also contains who and when the massages send. |
| **5b. Chat Room Input** | Players type text here to the others. |
| **5c. Submit Button** | Players click to send text to the others. |

**Table 4.2.5(b) - Chat room Screen**

**4.2.6 Game Rules Screen**



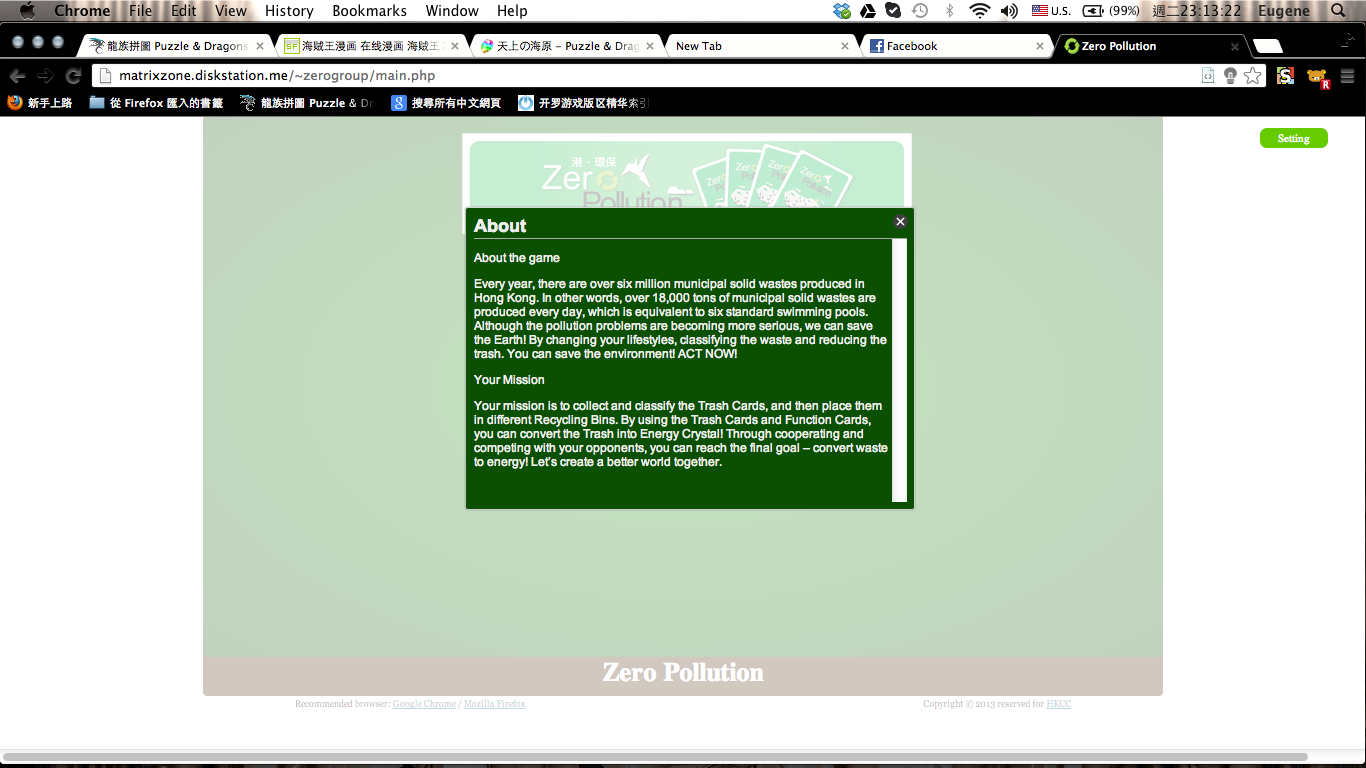
6a4a

**Diagram 4.2.6(a) - Game Rules Screen**

|  |  |
| --- | --- |
| **Name:** Game Rules Screen | |
| **Description:** The page of showing the game rules. Users can check the information of cards, set up, game process, when will the game end and how to win this game. | |
| **Button / Input / Output:** | |
| **6a. Close Button** | Players can close the window after viewing and continuous his/ her playing. |

**Table 4.2.6(b) - Game Rules Screen**

**4.2.7 About Screen**



**Diagram 4.2.7(a) - About Screen**

7a6a4a

|  |  |
| --- | --- |
| **Name:** About Screen | |
| **Description:** The page of showing about us. Users can check the information of this game and the mission of this game. | |
| **Button / Input / Output:** | |
| **7a. Close Button** | Players can close the window after viewing and continuous his/ her playing. |

**Table 4.2.7(b) - About Screen**

**4.2.8 Credit Screen**



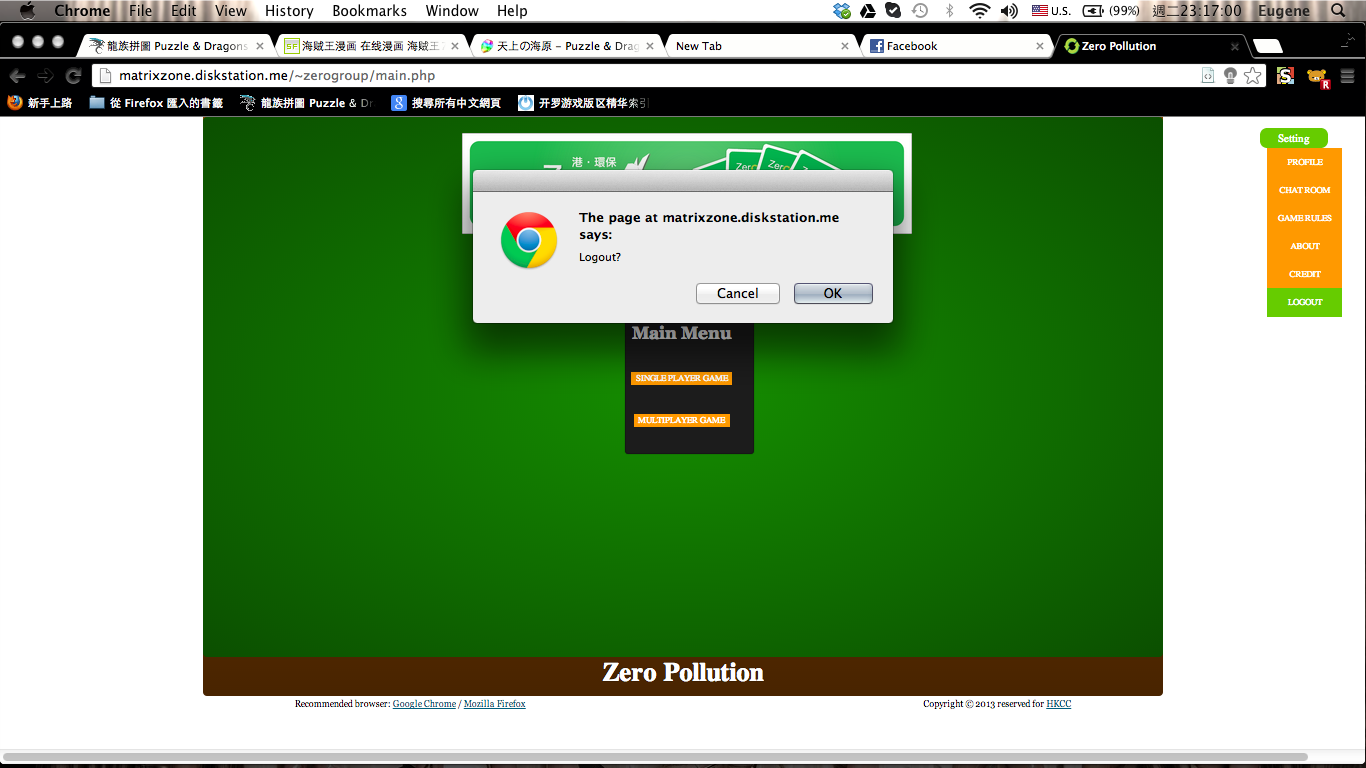
8a6a4a

**Diagram 4.2.8(a) - Credit Screen**

|  |  |
| --- | --- |
| **Name:** Credit Screen | |
| **Description:** The page of showing credit of this game. Users can check the background of this game, including design, coding and the source of this game. | |
| **Button / Input / Output:** | |
| **8a. Close Button** | Players can close the window after viewing and continuous his/ her playing. |

**Table 4.2.8(b) - Credit Screen**

**4.2.9 Logout Screen**



9a

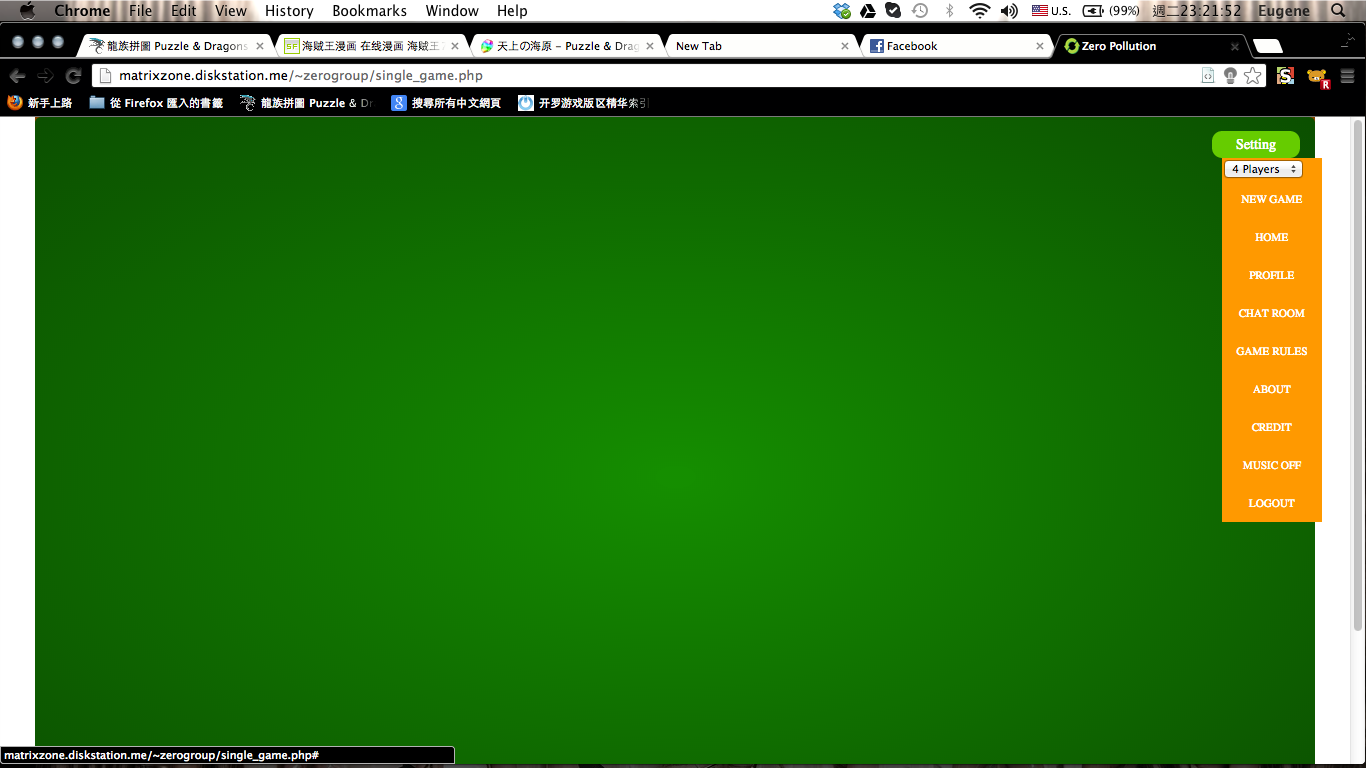
9b

**Diagram 4.2.9(a) - Logout Screen**

|  |  |
| --- | --- |
| **Name:** Logout Screen | |
| **Description:** This page will conform the players want to logout or not. | |
| **Button / Input / Output:** | |
| **9a. Cancel Button** | Players can choose not logout and continuous with his/her pervious page. |
| **9b. OK Button** | Players can choose logout and return to the Login Screen (screen 1). |

**Table 4.2.9(b) - Logout Screen**

**4.2.10 Game Main Menu Screen (Single Player)**



10c

10b

10a

**Diagram 4.2.10(a) - Game Main Menu Screen (Single Player)**

|  |  |
| --- | --- |
| **Name:** Game Main Menu Screen (Single Player) | |
| **Description:** Players choose what is the action next after choosing single player mode | |
| **Button / Input / Output:** | |
| **10a. New Game Button** | Players need to choose the players number before starting the game. After choosing and confirm, the game will start. |
| **10b. Home Button** | Players can back to the main menu screen (Screen 3). |
| **10c. Music Button** | Players can choose the music on or off here. |

**Table 4.2.10(b) - Game Main Menu Screen (Single Player)**

**4.2.11 Game Main Menu Screen (Single Player)**

11a

11b

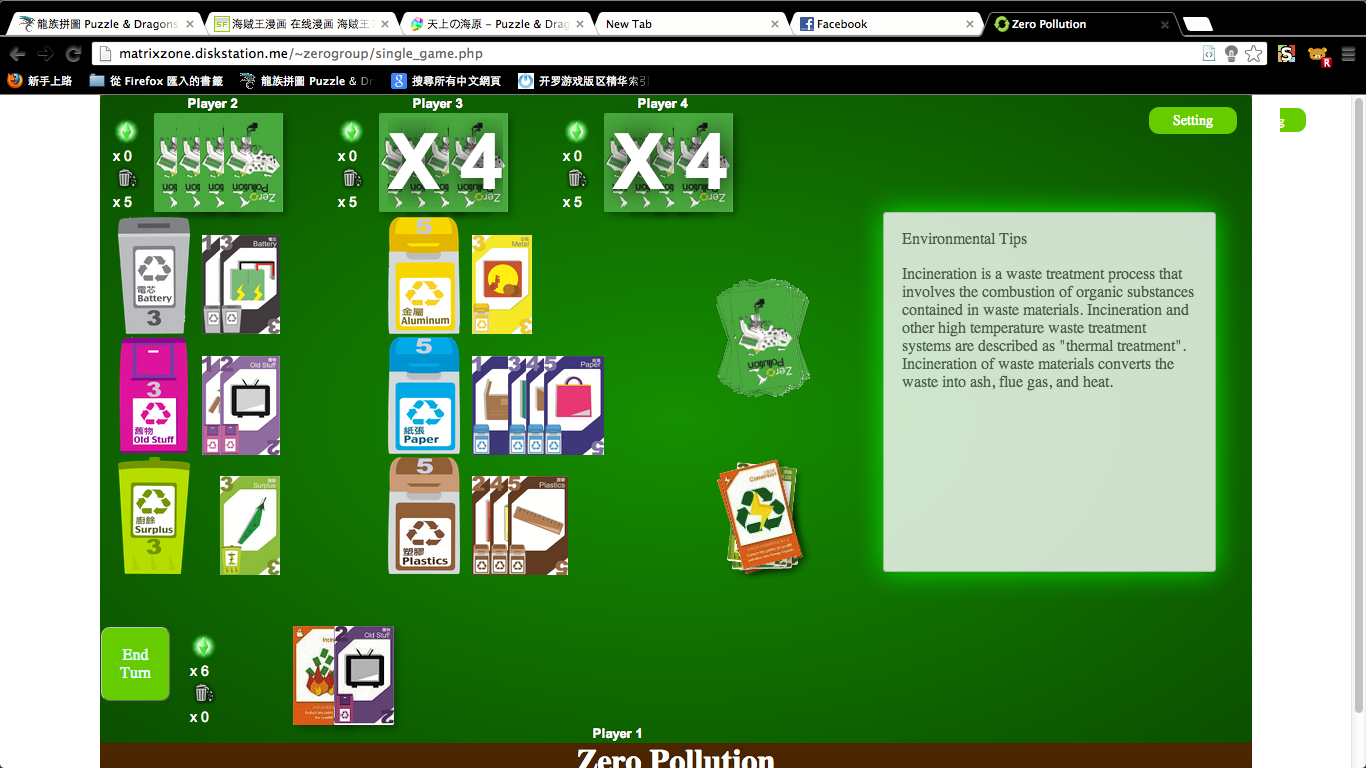
11c

11d

11e

11f

11g



**Diagram 4.2.11(a) - Main game page screen (Single Player)**

|  |  |
| --- | --- |
| **Name:** Main game page screen (single player) | |
| **Description:** This is the main game page for the system. Players would play the game in this page. | |
| **Button / Input / Output:** | |
| **11a. Game Board** | The game board built as the base of the game. Cards are placed inside and it makes action to different type of card. |
| **11b. Player Card** | Cards, which the player owned. |
| **11c. Environmental tips** | When the player selecting a specific card, the tips that about this card will be shown. All tips are about how to make our life green or the information about the rubbish. |
| **11d. End Turn Button** | If players cannot place any cards, they can click this button to end this turn. The system would then show all of the cards to other players and give two new cards to the players. The system would check whether players cannot place any cards or not. |
| **11e. Landfill and The Energy Crystals (Self)** | This area can count the landfill level of players and shows the energy crystals of players. |
| **11f. Information of opponents** | This shows remain numbers of cards, landfill level and crystals of each players. |
| **11g. Played Card area** | All played card will be placed in this area. Players can see which of the last card played. |

**Table 4.2.11(b) - Main game page screen (Single Player)**

**4.2.12 Select room Screen**



12b

12c

12a

**Diagram 4.2.12(a) - Select room Screen**

|  |  |
| --- | --- |
| **Name:** Select room Screen | |
| **Description:** This page will show the state, joined players’ name, information of each table. The players who first join the table will be the host and other can join the table. The maximum numbers of players in each table are 6. Only host can start the game. | |
| **Button / Input / Output:** | |
| **12a. Table** | The state, joined players’ name, information of each tables. |
| **12b. Start the game Button** | Host decides to start the game by pressing it. Only host have the right to start his/ her own tables game. |
| **12c. Leave the table Button** | Players can leave the table after joined any tables. |

**Table 4.2.12(b) - Select room Screen**

**4.2.12 Main game page Screen (multi-player)**

13a

13b

13c

13d

13e

13f

13g13e

****

**Diagram 4.2.13(a) - Main game page Screen (multi-player)**

|  |  |
| --- | --- |
| **Name:** Main game page Screen (multi-player) | |
| **Description:** This is the main game page for the system. Players would play the game in this page. | |
| **Button / Input / Output:** | |
| **13a. Game Board** | The game board built as the base of the game. Cards are placed inside and it makes action to different type of card. |
| **13b. Player Card** | Cards that the player owned. |
| **13c. Environmental tips** | When the player selecting a specific card, the tips that about this card will be shown. All tips are about how to make our life green or the information about the rubbish. |
| **13d. End Turn Button** | If players cannot place any cards, they can click this button to end this turn. The system would then show all of the cards to other players and give two new cards to the players. The system would check whether players couldn’t place any cards or not. |
| **13e. Landfill and The Energy Crystals (Self)** | This area can count the landfill level of players and shows the energy crystals of players. |
| **13f. Information of opponents** | This shows remain numbers of cards, landfill level and crystals of each players. |
| **13g. Played Card area** | All played card will be placed in this area. Players can see which of the last card played. |

**Table 4.2.13(b) - Main game page Screen (multi-player)**

**4.3 System Structure**

**4.3.1 Description of Source Code**

Here is a simple description of all of the source files, which provide a brief idea of how the game worked.

1. Create the MySQL database

* database\zeropollution.sql:

To create the database of this game, we can run this file under MySQL environment to insert the basic structure of the game database

* file inside database\zeropollution:

Created MySQL database of the system

2. Login to the system

* index.html:

Main page that provide login and register functions

* scripts\index.js:

Javascript that validate register function

* playerid.php:

Get player id to see if such player id existed or not

* connect.php:

Validate login function and bring to main page

3. Control panel functions of the game

* main.php:

Main page of game, allow player to choose playing in single or online

* get\_profile.php:

Get player profile from server

* chatroom.php:

Chat room of the game

* set\_message.php:

Send chat room message to server

* get\_message.php:

Get chat room message from server

4. Single player mode (refer Multi-player mode)

* single\_game.php:

Main page of single player mode, contain game element

* scripts\single\_game.js:

Handling game process

* scripts\single\_ui.js:

Handling game layout

* scripts\single\_card.js:

Giving out constant information of all cards

5. Multi-player mode (Before the game start)

* gameroom.php:

Game hall for multi-player mode, it contains two tables for different players play the game online

* scripts\main.js:

Handle the game room part of the system, bring players in game or leave table

* record\_selected\_table.php:

Record selected table by players

* set\_to\_started\_state.php:

Set table status to started when game is started

* tables\_status.php:

Keep polling the table status until players leave table or enter the game

* leave\_table.php:

Handle leaving a table by player control

6. Multi-player mode (Game started)

* initial\_game.php:

Initial game information, such as distribute the card, in server

* multi\_game.php:

Main page of multi-player mode, contain game element

* scripts\multi\_game.js:

Handling game process

* scripts\multi\_ui.js:

Handling game layout

* scripts\multi\_card.js:

Giving out constant information of all cards

* scripts\server\_request.js:

Handling communication between players and server

* update\_game.php:

Update game data to server after each play of card

* renew\_game.php:

Renew game status from server in every 0.1 second

* renew\_ui.php:

Renew user interface from server in every 0.1 second

* get\_enable\_status.php:

Keep polling to the server to check if a player can play card or not

7. Multi-player mode (Game ended)

* release\_table.php:

Empty and release table for other players

* set\_win\_lose.php:

Increment players win lose to the database

8. Others

* styles\style.css:

Styling the whole system

* scripts\jquery-1.9.1.js:

jQuery library

* icon.png:

Icon next to title

* file inside images:

All images showed inside the game

* music\bgm.mp3:

Background music of the game

**4.3.2 List of Front-end programs**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of program** | **Function** | **Input From** | **Responded By** |
| Index.html | 1. An interface for users to login  2. User can register their account | 1.Login not success:  index.html  Login success:  index.html  2. Register not success:  index.html  Register success:  index.html | 1.login not success:  connect.php  login success:  connect.php and  main.php  2. Register not success:  index.js  Register success:  index.js and  connect.php |
| main.php | A menu for user selects 1. “Single Player Game” or  2. “Multiplayer Game” | 1. main.php   2. main.php | 1. single\_game.php   2. gameroom.php |
| single\_game.php | A sinlge player game mode and interface for player | single\_game.php | single\_game.js  single\_ui.js single\_card.js |
| gameroom.php | Display the available table to user | gameroom.php | multi\_game.php |

|  |  |  |  |
| --- | --- | --- | --- |
| chatroom.php | A chat room which let player communicate | get\_message.php | set\_message.php |
| multi\_game.php | A multiplayer game mode and interface for player | multi\_game.php | multi\_game.js  multi\_ui.js multi\_card.js |

**Table 4.3.2 - List of Front-end programs**

**4.3.3 List of Back-end programs**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of program** | **Function** | **Input From** | **Responded By** |
| Index.js | javascript that validate register function | index.html | Playerid.php |
| connect.php | Connect to mysql database and validate login function and bring to main page | Index.html | main.php |
| single\_game.js | Arrange the process of single player game | sinlge\_game.html | Screen |
| single\_ui.js | Adjust the user interface of single game | single\_game.html | Screen |
| single\_card.js | A function to store all the card for single player game | N/A | single\_game.php |
| set\_message.php | Send chat room message to server | chatroom.php | Table: chatroom |
| get\_message.php | Get chat room message from server | Table: chatroom | chatroom.php |

|  |  |  |  |
| --- | --- | --- | --- |
| main.js | Arrange the operation in main.php | main.php | Screen |
| multi\_game.js | Arrange the process of multiplayer game | multi\_game.html | Screen |
| multi\_ui.js | Adjust the user interface of multiplayer game | multi\_gmae.html | Screen |
| multi\_card.js | A function to store all the card for multiplayer game | N/A | multi\_game.php |
| server\_request.js | Handling communication between players and server) | multi\_game.php | multi\_game.php |
| get\_enable\_status.php | keep polling to the server to check if a player can play card or not | multi\_game.php | multi\_game.php |
| get\_profile.php | Get player profile from server | Table: player | Screen |
| initial\_game.php | Initial the game | single\_game.php multi\_game.php | single\_game.php multi\_game.php |
| leave\_table.php | Let player leave the table | gameroom.php | gameroom.php |
| playerid.php | Get player id to see if such player id existed or not | Table: player | Index.js |
| record\_selected\_table.php | Store the number of player in each table and not allow player get in when the table of game is started | gameroom.php | Table: tables |
| release\_table.php | Release table when the game is finished | multi\_game.php | gameroom.php |
| renew\_game.php | Renew game status from server in every 0.1s | multi\_game.php | multi\_game.php |
| renew\_ui.php | Renew user interface from server in every 0.1s | multi\_game.php | multi\_game.php |
| set\_to\_started\_state.php | Set table status to started when game is started | multi\_game.php | Table: tables |
| set\_win\_lose.php | Adjust the win player and lose player | multi\_game.php | Table: player |
| tables\_status.php | Get the status of table to player from database | Table: tables | gameroom.php |
| update\_game.php | update the information of game such as the number of card pile | multi\_game.php | Table: subtable1  subtable2 |

**Table 4.3.3 - List of Back-end programs**

**4.3.4 Flow of main programs and sub-programs**

Get message from database

Sent message to database

Arrange the game process

<- Register not success

index.js

index.html

connect.php

main.php

Login not success

Login success

playerid.php

Register success

single\_game.php

multi\_game.php

chatroom.php

get\_message.php

set\_message.php

gameroom.php

main.js

record\_selected\_table.php

set\_to\_started\_state.php

table\_status.php

leave\_table.php

initial\_game.php

multi\_game.js

multi\_ui.js

multi\_card.js

server\_request.js

update\_game.php

renew\_game.php

renew\_ui.php

get\_enable\_status.php

release\_table.php

set\_win\_lose.php

single\_game.js

single\_ui.js

single\_card.js

Game room operation

**Table 4.3.4 - Flow of main programs and sub-programs**

**4.3.5 Classes and programs involved in each module**

|  |  |  |
| --- | --- | --- |
| **Classes** | **Source files** | **Description** |
| game | multi\_game.js, single\_game.js | game class contains all the functions that used for processing the game flow |
| ui | multi\_ui.js, single\_ui.js | ui class contains all the functions that used for editing the user layout of the game |
| card | multi\_card.js, single\_card.js | card class contains all the functions that used for giving out constant information of all cards |

**Table 4.3.5 - Classes and programs involved in each module**

**4.4 Database Design**

**Table Objects**

The table below state the catalogue of each table belongs to and it’s brief description.

|  |  |
| --- | --- |
| **Table Name** | **Description** |
| chatroom | Chat room information |
| player | Player account information |
| subtable1 | table1 game information for animation |
| subtable2 | table2 game information for animation |
| table1 | table1 information |
| table2 | table2 information |
| tables | Multiplayer tables entrance information |

**Table 4.4(a) - Table Objects**

**Primary Key**

The table below states the primary key of each table.

|  |  |
| --- | --- |
| **Table Name** | **Primary Key** |
| chatroom | record\_id |
| player | player\_id |
| subtable1 | type\_id |
| subtable2 | type\_id |
| table1 | record\_id |
| table2 | record\_id |
| tables | table\_id |

**Table 4.4(b) - Primary Key**

**Foreign Key**

The table below shows the relationship and where the foreign key attributes are reference from.

|  |  |  |  |
| --- | --- | --- | --- |
| **Foreign Key**  **Name** | **Attribute in**  **(Primary Table)** | **Attribute in**  **(Foreign Key Table)** | **Relationship** |
| player\_id | player\_id(player) | player\_id(chatroom) | 1:M |
| player\_id | player\_id(player) | player\_id(table1) | 1:1 |
| player\_id | player\_id(player) | player\_id(table2) | 1:1 |

**Table 4.4(c) - Foreign Key**

**Full List of Entity – Attributes**

The table below shows full list of attributes in each entity table.

|  |  |
| --- | --- |
| **Table** | **Attributes** |
| chatroom | record\_id |
| chatroom | player\_id |
| chatroom | message |
| chatroom | time |

**Table 4.4(d) - Full List of Entity – Attributes 1**

|  |  |
| --- | --- |
| **Table** | **Attributes** |
| player | player\_id |
| player | passowrd |
| player | e\_mail |
| player | register\_date |
| player | real\_name |
| player | phone\_no |
| player | address |
| player | win\_count |
| player | lose\_count |

**Table 4.4(e) - Full List of Entity – Attributes 2**

|  |  |
| --- | --- |
| **Table** | **Attributes** |
| subtable1 | type\_id |
| subtable1 | existed |

|  |  |
| --- | --- |
| **Table** | **Attributes** |
| subtable2 | type\_id |
| subtable2 | existed |

**Table 4.4(f) - Full List of Entity – Attributes 3**

**Table 4.4(g) - Full List of Entity – Attributes 4**

|  |  |
| --- | --- |
| **Table** | **Attributes** |
| table1 | record\_id |
| table1 | player\_id |
| table1 | type |
| table1 | cardlist |
| table1 | cardcount |
| table1 | crystal |
| table1 | landfill |
| table1 | play\_card\_enable |
| table1 | timer |

**Table 4.4(h) - Full List of Entity – Attributes 5**

|  |  |
| --- | --- |
| **Table** | **Attributes** |
| table2 | record\_id |
| table2 | player\_id |
| table2 | type |
| table2 | cardlist |
| table2 | cardcount |
| table2 | crystal |
| table2 | landfill |
| table2 | play\_card\_enable |
| table2 | timer |

**Table 4.4(i) - Full List of Entity – Attributes 6**

|  |  |
| --- | --- |
| **Table** | **Attributes** |
| tables | table\_id |
| tables | playerlist |
| tables | playercount |
| tables | status |
| tables | joinenable |

**Table 4.4(j) - Full List of Entity – Attributes 7**

**Table Access**

The table below shows the tables affected when a certain transaction is carried out.

(Notation \* means it depends on which table is the game in, table1 or table2)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Create** | **Delete** | **Update** | **Action Target** | **Affected Table** |
| **√** |  |  | Register | player |
|  |  | **√** | Enter table | tables |
|  |  | **√** | Leave table | tables |
|  |  | **√** | Start game | tables, table\* |
|  |  | **√** | Play a card | table\*, subtable\* |
|  |  | **√** | End turn | table\* |
|  |  | **√** | Crystal increase/decrease | table\* |
|  |  | **√** | Landfill increase/decrease | table\* |
|  |  | **√** | Game End(Win/Lose) | player |
|  |  | **√** | Send chat room message | chatroom |

**Table 4.4(k) - Table Access**

**Attribute Constraints**

The table below shows the validation rules of the attributes in the database.

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Attribute Name** | **Validation Rules** |
| chatroom | record\_id | The value should be auto numbered by the system.  Value should not be null. |
| player\_id | Value should not be null. |
| message | Value should not be null. |
| time | Value should not be null. |

**Table 4.4(l) - Attribute Constraints 1**

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Attribute Name** | **Validation Rules** |
| player | player\_id | Value should not be null. |
| passowrd | Value should not be null. |
| e\_mail | Value should not be null. |
| register\_date | The value should be a date.  Value should not be null. |
| real\_name | Value can be null. |
| phone\_no | Value should not be null. |
| address | Value can be null. |
| win\_count | Value should not be null. |
| lose\_count | Value should not be null. |

**Table 4.4(l) - Attribute Constraints 1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Table Name** | **Attribute Name** | **Validation Rules** | |
| subtable1 | type\_id | Value should not be null. |
| existed | Value should not be null. Either 1/0. |

**Table 4.4(m) - Attribute Constraints 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Table Name** | **Attribute Name** | **Validation Rules** | |
| subtable2 | type\_id | Value should not be null. |
| existed | Value should not be null. Either 1/0. |

**Table 4.4(n) - Attribute Constraints 3**

|  |  |  |  |
| --- | --- | --- | --- |
| **Table Name** | **Attribute Name** | **Validation Rules** | |
| table1 | record\_id | | Value should not be null. |
| player\_id | | Value should not be null. |
| type | | Value should not be null. |
| cardlist | | Value should not be null. |
| cardcount | | Value should not be null. |
| crystal | | Value should not be null. |
| landfill | | Value should not be null. |
| play\_card\_enable | | Value can be null. |
| timer | | The value should be a time.  Value should not be null. |

**Table 4.4(o) - Attribute Constraints 4**

|  |  |  |  |
| --- | --- | --- | --- |
| **Table Name** | **Attribute Name** | **Validation Rules** | |
| table2 | record\_id | | Value should not be null. |
| player\_id | | Value should not be null. |
| type | | Value should not be null. |
| cardlist | | Value should not be null. |
| cardcount | | Value should not be null. |
| crystal | | Value should not be null. |
| landfill | | Value should not be null. |
| play\_card\_enable | | Value can be null. |
| timer | | The value should be a time.  Value should not be null. |

**Table 4.4(p) - Attribute Constraints 5**

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Attribute Name** | **Validation Rules** |
| tables | table\_id | Value should not be null. |
| playerlist | Value should not be null. |
| playercount | Value should not be null. |
| status | Value should not be null. “Started” / “Pending”. |
| joinenable | Value should not be null. Either 1/0. |

**Table 4.4(q) - Attribute Constraints 6**

**Detailed Attributes Description**

The table below shows the details of all attributes use in the database.

(The duplicated table1/table2 and subtable1/subtable2 have been reduced as they are exactly the same.)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute Name** | **Description** | **Mand-atory** | **Type** | **Storage Length** | **Default Value** | **Example** |
| record\_id(chatroom) | Record ID | **√** | int | 3 |  | 3 |
| message | Message | **√** | text | auto |  | Hello World! |
| time | Send time | **√** | timestamp | auto |  | 2013-05-15 20:18:37 |
| player\_id | Player ID | **√** | varchar | 30 |  | test0 |
| passowrd | Password | **√** | varchar | 30 |  | password0 |
| e\_mail | E-mail | **√** | varchar | 40 |  | test0@yahoo.com.hk |
| register\_date | Register date | **√** | date | auto |  | 2011-10-19 |
| real\_name | Player real name |  | varchar | 30 |  | test 0 |
| phone\_no | Player phone no. | **√** | int | 20 |  | 91234576 |
| address | Player address |  | varchar | 100 |  | test0 address |
| win\_count | Player win times | **√** | int | 4 | 0 | 1000 |
| lose\_count | Player lose times | **√** | int | 4 | 0 | 598 |
| type\_id | Card type | **√** | int | 3 |  | 12 |
| existed | Card existence | **√** | tinyint | 1 | 0 | 1 |
| record\_id(table) | Record ID | **√** | varchar | 30 |  | 5 |
| type | Record type | **√** | varchar | 7 |  | player |
| cardlist | Card list | **√** | text | auto |  | #50##1##68# |
| cardcount | Card number | **√** | int | 3 |  | 4 |
| crystal | Crystal number | **√** | int | 3 |  | 1 |
| landfill | Landfill number | **√** | int | 3 |  | 5 |
| play\_card\_enable | Turn indicator |  | tinyint | 1 |  | 0 |
| timer | Timer | **√** | timestamp | auto |  | 2013-05-15 19:20:04 |
| table\_id | Table ID | **√** | varchar | 6 |  | table1 |
| playerlist | Player list | **√** | varchar | 255 |  | <li>test1</li> |
| playercount | Player number | **√** | int | 3 |  | 1 |
| status | Table status | **√** | varchar | 8 |  | pending |
| joinenable | Joinable indicator | **√** | tinyint | 1 | 0 | 1 |

**Table 4.4(r) - Detailed Attributes Description**

**4.5 Interfaces Samples**

**Case 1: Register**

Input screen:



1b

1h

1c

1d

1e

1f

1g

1j

1i

1a

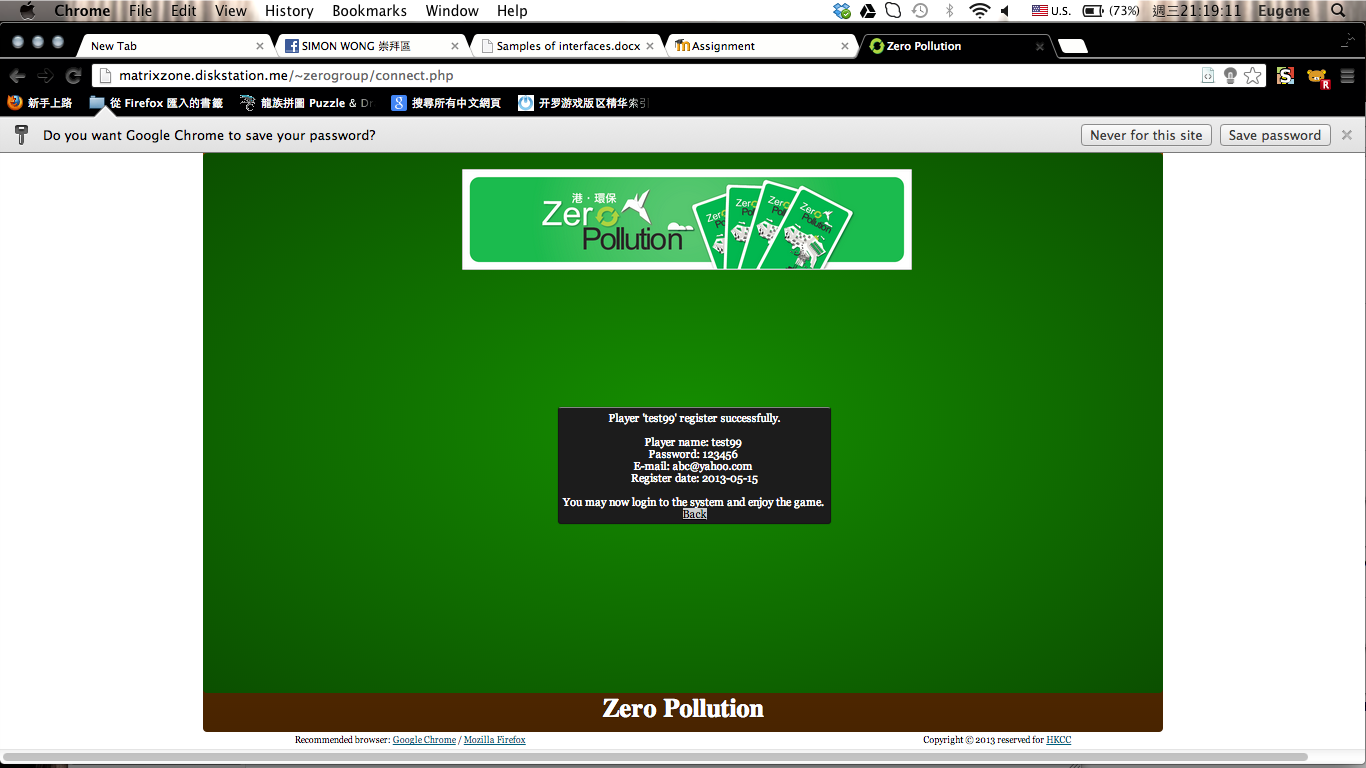
**Diagram 4.5(a) - Case 1: Register Input**

|  |  |
| --- | --- |
| **Description:** This page allows players to create a new account in order to login to the system. | |
| **Button / Input / Output:** | |
| **1a. Checking Output** | The system will check the input is valid or not. The error massage will be showed immediately. The system will check the inputted value are valid or not in every 0.2second. This can ensure users have no need to wait the system checking. |

|  |  |
| --- | --- |
| **1b. Player Name Input** | Players can input their login name here. The name should be at least 5 characters. This can enforce every new players take a longer users name, thus the probability of repeating players’ name will be lower. |
| **1c. Password Input** | Players can input their own password here. They can type in alphabets (case-sensitive), numbers and symbols. The password should be at least 6 characters. This can enforce users have a higher security’s password. As the account including some personal information (e.g. address or phone number), we hope every users’ account have a high protection. |
| **1d Conform password Input** | Player can re-input the password here. We hope users remember the password by re-typing his/ her password. Also, we want to double check if the password above is correct or not. |
| **1e. E-mail Input** | Players need to input their E-mail address here. The system would check whether the entered address control space and at sign or not. We want to prevent user type a wrong email address and make us cannot not send any information in the future. |
| **1f-1h. Your real name Input, Phone number Input, Home address Input** | Players can type his/her real name here. It is not a necessary process that players can choose type or not by his/her own will. Some players may want to meet some friends that have same interest (e.g. like environmental protection), these boxes is suitable for them. Also, there are some players may just want to try the game and they can choose finish the register by not providing those information |
| **1i. Submit Button** | Players can click this button to create a new account after providing valid information. |
| **1j. Clear Button** | This button would clear all the input placed in above input boxes. (2b. - 2h.). If the users want to re-type the information again, they need not to delete boxes one by one, but just press the “Clear “button. |

**Table 4.5(b) - Case 1: Register Input**

Output Screen:



**Diagram 4.5(c) - Case 1: Register Output**

|  |  |
| --- | --- |
| **Description:** This page informs players register the account successfully. | |
| **Button / Input / Output:** | |
| **Account Info window** | To remind the players, the system will show the player’s name, password, email address and the register date. |
| **Back button** | Players back to main menu by clicking this button. |

**Table 4.5(d) - Case 1: Register Output**

**Case 2: Host starts a game without other player(s)**

Input screen:

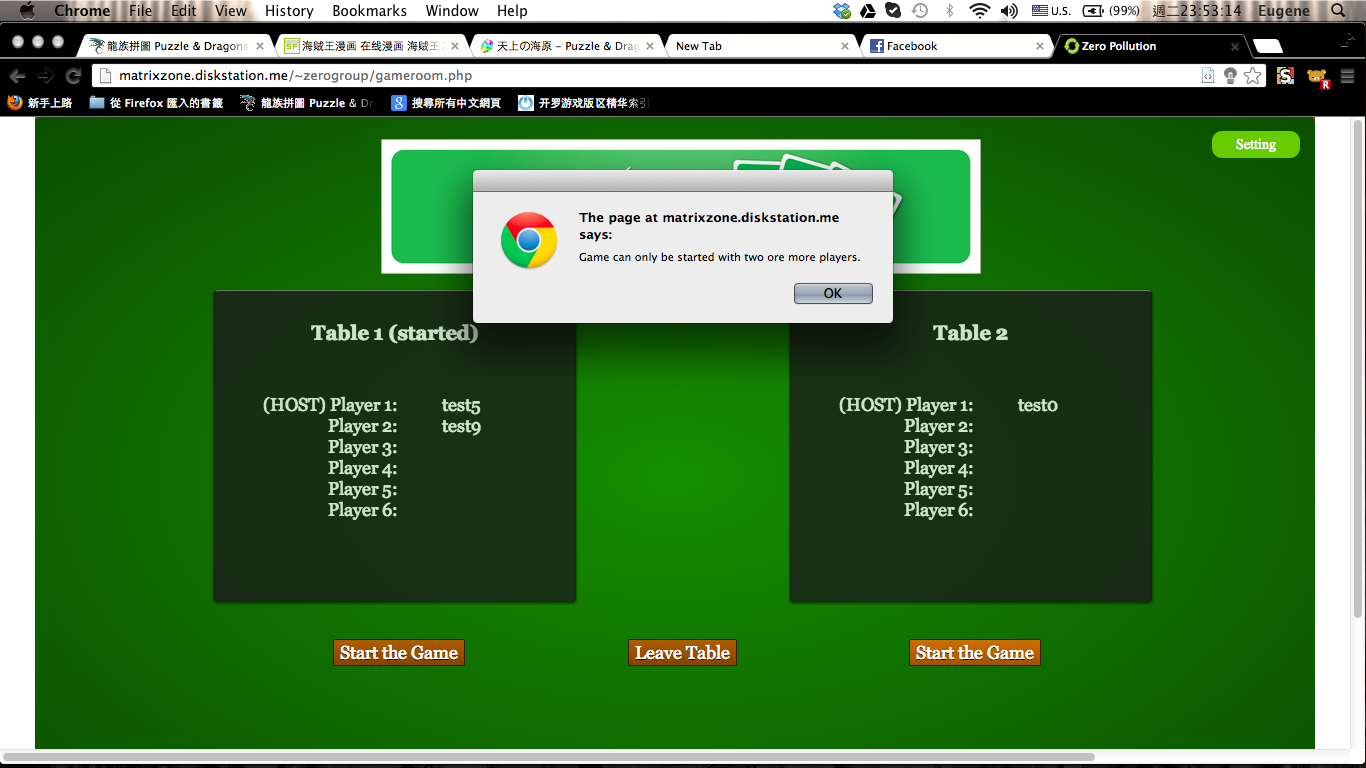


**Diagram 4.5(e) - Case 2: Host starts a game without other player(s) Input**

|  |  |
| --- | --- |
| **Description:** This page will show the state, joined players’ name, information of each table. | |
| **Button / Input / Output:** | |
| **12a. Table** | The state, joined players’ name, information of each table. The players who first join the table will be the host and other can join the table. We want to make the process of preparing as simple as we can. Every player wanted to be a host can enter a new table. This can prevent players that in the same table need to choose who is host and have the right to start the game. |
| **12b. Start the game Button** | Host decides to start the game by pressing it. Only host have the right to start his/ her own tables game. |
| **11c. Leave the table Button** | Players can leave the table after joined any tables. |

**Table 4.5(f) - Case 2: Host starts a game without other player(s) Input**

Output screen:



**Diagram 4.5(g) - Case 2: Host starts a game without other player(s) Output**

|  |  |
| --- | --- |
| **Button / Input / Output:** | |
| **Window pop out** | The system will control the host starting the game at least 2 players (including himself/herself). As this is a multi-player mode, there should be one more human player play with. We do not allow players play “single player mode” in “multi- player mode” because the number of table is limited. If many players play “single player mode” in “multi- player mode” in one time, those players who want to “multi-player mode” in “multi- player mode” will have no table. We cannot provide unlimited tables to prevent this condition happened as the server durability is limited. |

**Table 4.5(h) - Case 2: Host starts a game without other player(s) Output**

**Case 3: Players that non- host start the game**

Input screen:

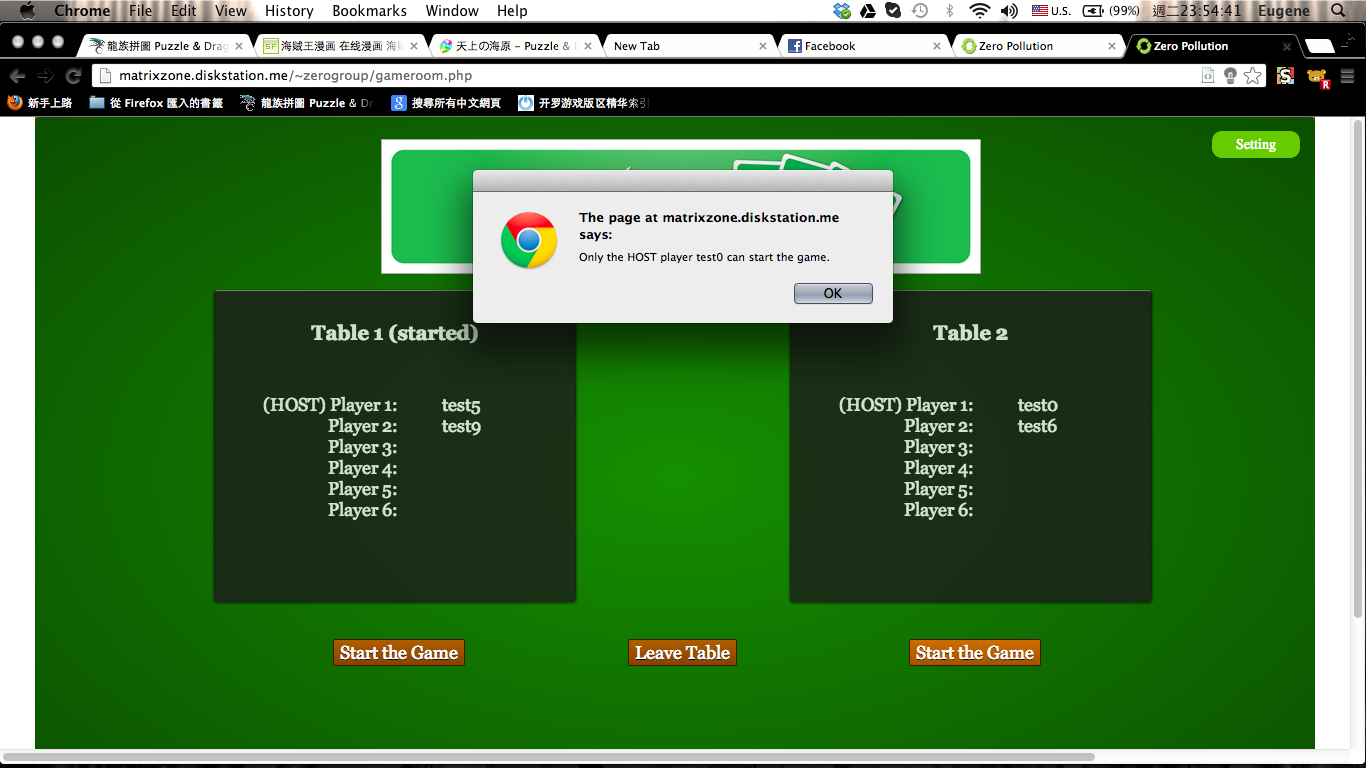


**Diagram 4.5(i) - Case 3: Players that non- host start the game Input**

|  |  |
| --- | --- |
| **Description:** This page will show the state, joined players’ name, information of each table. | |
| **Button / Input / Output:** | |
| **12a. Table** | The state, joined players’ name, information of each tables. The players who first join the table will be the host and other can join the table. We want to make the process of preparing as simple as we can. Every players if he /she want to be a host can enter a new table. This can prevent players that in the same table need to choose who is host and have the right to start the game. |
| **12b. Start the game Button** | Host decides to start the game by pressing it. Only host have the right to start his/ her own tables game. |
| **11c. Leave the table Button** | Players can leave the table after joined any tables. |

**Table 4.5(j) - Case 3: Players that non- host start the game Input**

Output screen:



**Diagram 4.5(k) - Case 3: Players that non- host start the game Output**

|  |  |
| --- | --- |
| **Button / Input / Output:** | |
| **Window pop out** | The system will control the only host can start the game. As there are 6 players in each table, if all the players have the right to start the game, this may cause a lot of problems. Some players may want to start the game but the host may want to wait more players to join. Some unnecessary argument may occur. Only host have the right to start the game can prevent this condition happen. |

**Table 4.5(l) - Case 3: Players that non- host start the game Output**

**5. Software packages and tools used**

**Software tools we used to develop our project:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Software Name** | **Version** | **Function in System** | **Main File used** |
| Notepad++ | 6.3.2 | Programming | All source codes |
| Adobe Dreamweaver | CS5 | Checking errors | All source codes |
| Google Chrome | 26.0.1410.67 | Browsing website | All source codes |
| Mozilla Firefox | 21.0 | Browsing website | All source codes |
| Microsoft Word | 2010 | Documentation | All text documents |
| Apache httpd | 2.4.4 | Setting up server | All web documents |
| MySQL | 5.6.10 | Setting up server | All web documents |
| phpMyAdmin | 4.0.1 | Setting up server | All web documents |
| Xampp | 1.8.1 | Maintaining server | All web documents |
| Adobe Photoshop | CS5 | Editing images | All images |

**Programming languages we used to develop our project:**

|  |  |
| --- | --- |
| **Language Name** | **Version** |
| Javascript | 1.8.5 |
| Cascading Style Sheet 3 | 3 |
| HTML5 | 5 |
| SQL | 2011 |
| JQuery | 2.0.0 |
| PHP | 5.4.15 |

**List of external program and packages used:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Package Name** | **Function in System** | **Source** | **Main File used** |
| jquery-1.9.1 | Provide standardized function for JavaScript | http://www.jquery.com | All JavaScript documents |

**6. Installation Guideline**

The Web-based “Zero-Pollution” is supported by three system, the Apache HTTP Web server, MySQL database, and a web browser (Google Chrome/ Mozilla Firefox is recommended). We also recommend installing the XAMPP package to simplify the installation of Apache, MySQL and phpmyadmin.

**Installation Procedures for the XAMPP package**

System Requirements:

There is no specific minimum system requirement for the XAMPP package, but the official page stated that the ram size would be the key of running speed.

The XAMPP package will run on Linux, Windows, Mac OS X and Solaris.

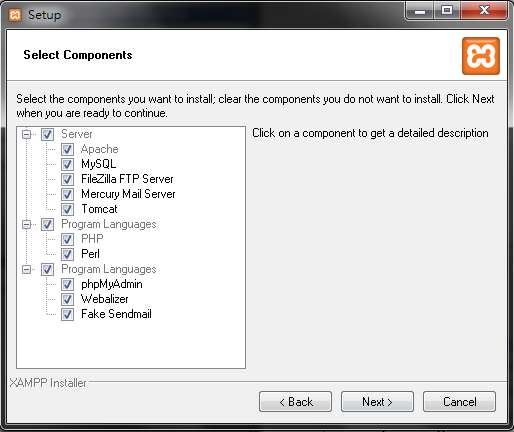
In addition, the Apache HTTP Web server has no specific minimum system requirements. Also, MySQL needs to run on x86 or x86\_64 platform.

**Download and Install XAMPP**

1. Go to the website: <http://www.apachefriends.org/en/xampp.html>
2. Select the version according to the OS of the platform (The following steps will be exampled by the Windows version)
3. Select “XAMPP Windows 1.8.1 Installer (97MB)” and download it
4. Run the installer “xampp-win32-1.8.1-VC9-installer.exe”
5. Click “Next” on the Setup window

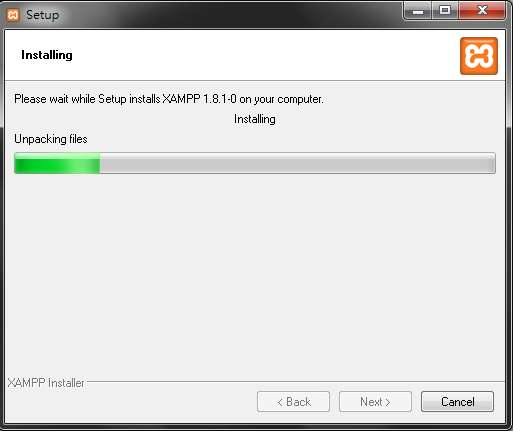


1. Make sure Apache, MySQL and phpMyAdmin have been checked before clicking “Next”



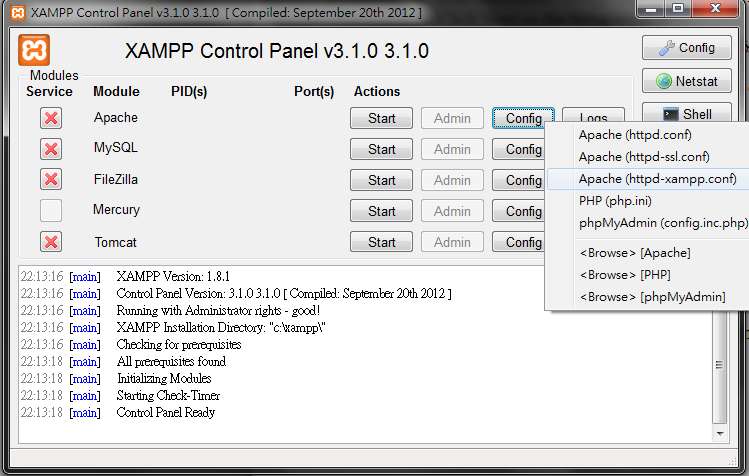
1. The installer default the installation path as “C:\xampp”, you can manually change the location to another directory

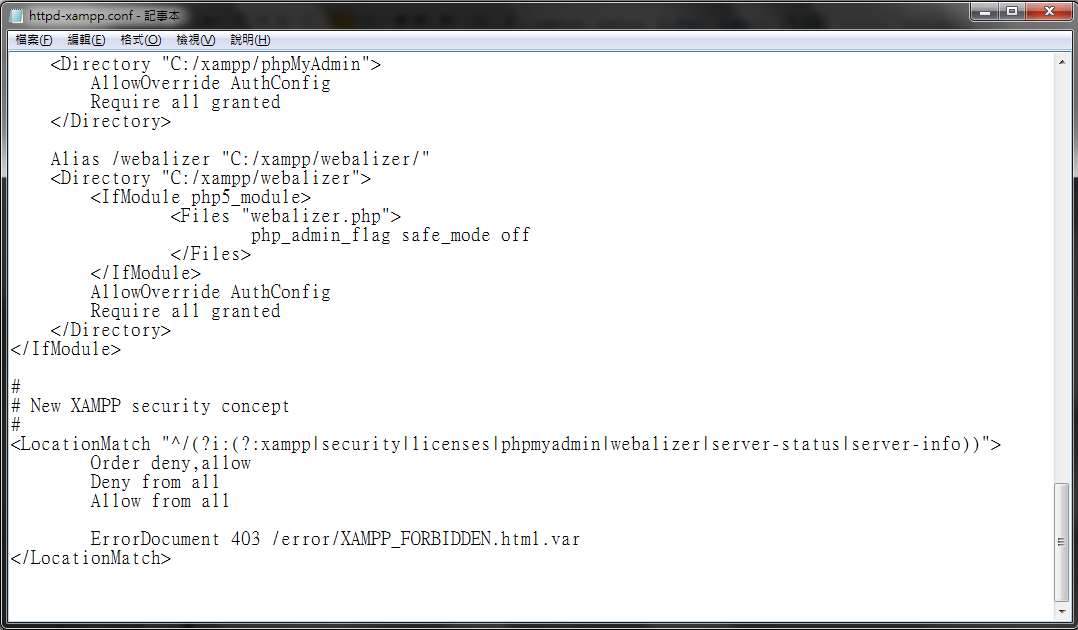


1. Uncheck the box “Learn more about BitNami for XAMPP” and click “Next”
2. Click “Next” again to ready the installation  
   
3. Wait for the completion of installation
4. Click “Finish”

**Configuration of XAMPP**

Click the button “Config” of the “Apache” row on the Control Panel of XAMPP and select Apache (httpd-xampp.conf)



1. Scroll down the document to the section “New XAMPP security concept”
2. Modify the row of “Allow from”, adding the IP address that you wish it to connect to the server, or simply use “Allow from ALL” to allowing the public   
   

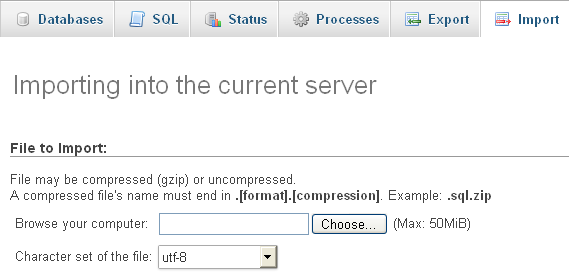
This allows the server to be connected by device from other network.

**Database and program configuration for running the game**

1. Copy all program code files of the Web-based “Zero-Pollution” to the path C:\xampp\htdocs (or the manual input path while installing the XAMPP)
2. Open a web Browser
3. Enter localhost/phpmyadmin on the address bar
4. Click on “Database”, under “**Create database”** input “zeropollution” and click “Create”



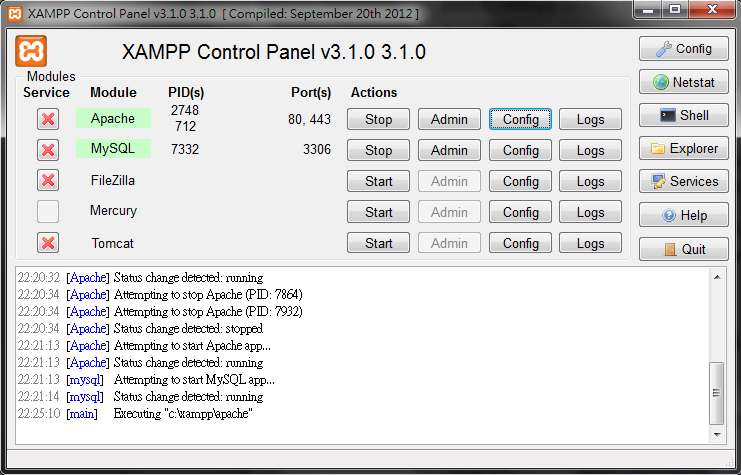
1. Click on the button “Import” and select the file zeropollution.sql from the program code files, then click “GO”



The database and Apache server should be now configured and the files of Web-based “Zero-Pollution” are imported.

**Starting up Procedures for the game**

1. Click the button “Start” of the “Apache” and “MySQL” row on the Control Panel of XAMPP



1. Open a browser, enter <http://localhost/index.html> or <http://(server_ip)/index.html> to open the game

**7. Samples of source code**

**7.1 multi\_game.js - playCardProcessor**

**Code:**

|  |
| --- |
| // Do the checking task after the player played a card  playCardProcessor : function(cardObject) {  var currentPlayer = parseInt(playerPosition);  if(card.isFunctionalCard(cardObject)) {  ui.addCardToFunctionalArea(cardObject);  if(card.isConversionCard(cardObject)) {  // Convert the points of Landfill pollution into Energy Crystals  playerCrystal[currentPlayer] += playerLandfill[currentPlayer];  playerLandfill[currentPlayer] = 0;  game.checkNextTurn();  } else if(card.isDishonestTraderCard(cardObject)){  // Steal 1 Energy Crystal from your opponent, increasing 2 point of Landfill  // Pop a window and let him select a suit  ui.clearToast();  var suitWindow = $('#suit-selector');  var background = $('#popup-background');  suitWindow.fadeIn('fast');  background.fadeIn('fast');  ui.clearToast();  suitWindow.bind('click', function(event) {  var target = event.target;  if(target.nodeName.toLowerCase() === 'button') {  suitWindow.unbind('click');  suitWindow.hide();  background.hide();  var playerId = parseInt(target.id.split('-', 2)[1]);  playerLandfill[playerId] += 2;  playerCrystal[currentPlayer] += 1;  if (totalCrystal != 0)  totalCrystal--;  ui.bindCardClickListener();  game.checkNextTurn();  }  });  } else if(card.isExcessLandfillCard(cardObject)){  // Increase two points from the Landfill  for(var i = 0; i < playerCount ;i++) {  playerLandfill[i] += 2;  }  game.checkNextTurn();  } else if(card.isHighTechnologyCard(cardObject)){  // As part of any Trash combination  // If it is a human player, pop a window and let him select a suit  ui.clearToast();  var suitWindow = $('#rubblish-selector');  var background = $('#popup-background');  suitWindow.fadeIn('fast');  background.fadeIn('fast');  ui.clearToast();  suitWindow.bind('click', function(event) {  var target = event.target;  if(target.nodeName.toLowerCase() === 'button') {  suitWindow.unbind('click');  suitWindow.hide();  background.hide();  var cardId = parseInt(target.id);  ui.addCardToDiscardPile(cardObject);  game.addCardToDiscardPile(cardObject);  cardObject.typeId = cardId;  ui.addCardToRubblishArea(cardObject);  cardTypeIsExist[cardObject.typeId] = true;  game.checkRubblishCollect(currentPlayer);  ui.bindCardClickListener();  game.checkNextTurn();  }  });  } else if(card.isIncineratorFailureCard(cardObject)){  // increase one point from the Landfill  for(i=0; i < playerCount ;i++) {  playerLandfill[i] += 1;  }  game.checkNextTurn();  } else if(card.isIncineratorCard(cardObject)){  // deduct one point from the Landfill  if (playerLandfill[currentPlayer] != 0)  playerLandfill[currentPlayer] -= 1;  game.checkNextTurn();  } else if(card.isJunkRetrievingCard(cardObject)){  // get a card from your oppenent's hand  // pop a window and let him select a suit  ui.clearToast();  var suitWindow = $('#suit-selector');  var background = $('#popup-background');  suitWindow.fadeIn('fast');  background.fadeIn('fast');  ui.clearToast();  suitWindow.bind('click', function(event) {  var target = event.target;  if(target.nodeName.toLowerCase() === 'button') {  suitWindow.unbind('click');  suitWindow.hide();  background.hide();  var playerId = parseInt(target.id.split('-', 2)[1]);  var numOfCard = players[playerId].length;  var cardId = Math.floor(Math.random() \* numOfCard);  var cardObject = players[playerId][cardId];  game.removePlayerCard(playerId, cardObject.id);  ui.removeCardFromOtherPlayerTray(playerId);  players[currentPlayer].push(cardObject);  ui.addCardToPlayerTray(cardObject);  ui.bindCardClickListener();  game.checkNextTurn();  }  });  } else if(card.isLandfillTransferCard(cardObject)) {  // Every player transfers their Landfill to next opponent in anti-clockwise  var temp = playerLandfill[0];  for(var i = 0; i < (parseInt(playerCount) - 1); i++) {  playerLandfill[i] = playerLandfill[(i+1)];  }  playerLandfill[parseInt(playerCount) - 1] = temp;  game.checkNextTurn();  }  } else {  ui.clearToast();  cardTypeIsExist[cardObject.typeId] = true;  ui.addCardToRubblishArea(cardObject);  game.checkRubblishCollect(currentPlayer);  game.checkNextTurn();  }  } |

**Explanation:**

This is function to perform different operations after a player play a card and then pass a card object to here.

In here, the system will identify which card type does the card object passed belongs to. The identification is based on a if … else … structure such as following.

If (the card type is x) {

perform x’s operations

} else if (the card type is y) {

perform y’s operations

} else {

perform z’s operations

}

When identification is finished, the system will detect which player plays this card and perform operations according to the player id and the card id in the system

For example, if the system receives a selection signal that player plays a functional card, and it will add the card into functional area. Then, system will continue to use if cases to check whether it is a Conversion Card, Dishonest Trader Card, Excess Landfill Card, High Technology Card, Incinerator Failure Card, Incinerator Card, Junk Retrieving Card or Landfill Transfer Card. After each card operation the system will run a ‘checkNextTurn()’ function which belong to the game class, to check whether this player’s turn should end after performing the operations.

Moreover, for single player mode, the end turn button will pass a card object with type id ‘32’. This is to virtualizing the human player to play a card and the operation of this card is to end this turn.

**7.2 multi\_game.js – intervalFunction2**

**Code:**

|  |
| --- |
| // Interval function2  function intervalFunction2() {  var json\_string = JSON.parse(server.renewUi(currentTable));  var cardTypeIsExist\_temp = new Array();  var stock\_temp = '';  var discardpile\_temp = '';    // Get played card information  for (var i = 0; i < cardTypeIsExist.length; i++){  if (json\_string.type[i] == 1) {  cardTypeIsExist\_temp[i] = true;  } else if (json\_string.type[i] == 0)  cardTypeIsExist\_temp[i] = false;  }    // Get player crystal and landfill information  for (var i = 0; i < playerCount; i++){  playerCrystal[i] = parseInt(json\_string.crystal[i]);  playerLandfill[i] = parseInt(json\_string.landfill[i]);  playerEnable[i] = parseInt(json\_string.enable[i]);  }    // Get stock and discard pile information  var patten = /#(.\*?)#/g;  stock\_temp = json\_string.cardlist[0].match(patten);  discardpile\_temp = json\_string.cardlist[1].match(patten);  if (stock\_temp != null){  for (var i = 0; i < stock\_temp.length; i++){  stock\_temp[i] = stock\_temp[i].replace(/#/g, "");  }  }  if (discardpile\_temp != null){  for (var i = 0; i < discardpile\_temp.length; i++){  discardpile\_temp[i] = discardpile\_temp[i].replace(/#/g, "");  }  }  var temp;  var numberOfCardPerType = card.getMaxCard() / card.getMaxCardType();  if (stock\_temp != null) {  for (var i = 0; i < stock\_temp.length; i++){  temp = new Object();  temp.id = stock\_temp[i];  temp.typeId = Math.floor(stock\_temp[i] / numberOfCardPerType);  stock\_temp[i] = temp;  }  }  if (discardpile\_temp != null) {  for (var i = 0; i < discardpile\_temp.length; i++){  temp = new Object();  temp.id = discardpile\_temp[i];  temp.typeId = Math.floor(discardpile\_temp[i] / numberOfCardPerType);  discardpile\_temp[i] = temp;  }  }    // Update player crystal and landfill  ui.updateCrystalArea();    // Update current player title  for (var i = 0; i < playerCount; i++) {  if (playerEnable[i] == 1 && i != playerPosition) {  for (var j = 1; j < playerCount; j++) {  if (playerName[i] == document.getElementById("player" + (j+1) + "-title").innerHTML) {  document.getElementById("player" + (j+1) + "-title").innerHTML = document.getElementById("player" + (j+1) + "-title").innerHTML + " <- (Current Turn)";  document.getElementById("player" + (j+1) + "-title").setAttribute("style", "color:#F00");  if (j != 1) {  document.getElementById("player" + (j) + "-title").innerHTML = document.getElementById("player" + (j) + "-title").innerHTML.replace(" <- (Current Turn)", "");  document.getElementById("player" + (j) + "-title").setAttribute("style", "color:#FFF");  }  }  }  }  }    // Update stock and discardpile  if (stock\_temp != null) {  if (stock.length != stock\_temp.length) {  ui.removeCardFromStockPile();  stock.pop();  }  }  if (discardpile\_temp != null) {  if (discardpile.length != discardpile\_temp.length) {  ui.addCardToDiscardPile(discardpile\_temp[discardpile\_temp.length - 1]);  discardpile.push(discardpile\_temp[discardpile\_temp.length - 1]);  }  }  // Update played card on table  for (var i = 0; i < cardTypeIsExist\_temp.length; i++) {  if (cardTypeIsExist[i] != true && cardTypeIsExist\_temp[i] == true) {  for (var j = 0; j < playerCount; j++) {  for (var k = 0; k < players[j].length; k++) {  if (players[j][k].typeId == i) {  ui.addCardToRubblishArea2(players[j][k]);  cardTypeIsExist[i] = true;  return;  }  }  }  }  }  } |

**Explanation:**

This function is used to collect data from the server in every 0.1 second. At the beginning, the function gets server data by calling server function renewUi(currentTable) in server\_request.js as following:

|  |
| --- |
| var json\_string = JSON.parse(server.renewUi(currentTable)); |

Then the function transfers and stores the json string into temporary variable cardTypeIsExist\_temp, stock\_temp and discardpile\_temp. The number of crystal, landfill level and enable status of each player are updated automatically. Those card information in temporary variable is also translated into card object with card ID and type ID.

|  |
| --- |
| var numberOfCardPerType = card.getMaxCard() / card.getMaxCardType();  if (stock\_temp != null) {  for (var i = 0; i < stock\_temp.length; i++){  temp = new Object();  temp.id = stock\_temp[i];  temp.typeId = Math.floor(stock\_temp[i] / numberOfCardPerType);  stock\_temp[i] = temp;  }  }  if (discardpile\_temp != null) {  for (var i = 0; i < discardpile\_temp.length; i++){  temp = new Object();  temp.id = discardpile\_temp[i];  temp.typeId = Math.floor(discardpile\_temp[i] / numberOfCardPerType);  discardpile\_temp[i] = temp;  }  } |

Afterwards, the function would update the number of crystal and landfill level first:

|  |
| --- |
| // Update player crystal and landfill  ui.updateCrystalArea(); |

After that, the function would check who the current player is. If that current player is not the same as client data, the function would update the title of that current player.

|  |
| --- |
| // Update current player title  for (var i = 0; i < playerCount; i++) {  if (playerEnable[i] == 1 && i != playerPosition) {  for (var j = 1; j < playerCount; j++) {  if (playerName[i] == document.getElementById("player" + (j+1) + "-title").innerHTML) {  document.getElementById("player" + (j+1) + "-title").innerHTML = document.getElementById("player" + (j+1) + "-title").innerHTML + " <- (Current Turn)";  document.getElementById("player" + (j+1) + "-title").setAttribute("style", "color:#F00");  if (j != 1) {  document.getElementById("player" + (j) + "-title").innerHTML = document.getElementById("player" + (j) + "-title").innerHTML.replace(" <- (Current Turn)", "");  document.getElementById("player" + (j) + "-title").setAttribute("style", "color:#FFF");  }  }  }  }  } |

If the stock pile or discard pile are changed, this function would update the layout of that two pile also. This can make the play card effect more realize.

|  |
| --- |
| // Update stock and discardpile  if (stock\_temp != null) {  if (stock.length != stock\_temp.length) {  ui.removeCardFromStockPile();  stock.pop();  }  }  if (discardpile\_temp != null) {  if (discardpile.length != discardpile\_temp.length) {  ui.addCardToDiscardPile(discardpile\_temp[discardpile\_temp.length - 1]);  discardpile.push(discardpile\_temp[discardpile\_temp.length - 1]);  }  } |

At last, the function would compare the cardTypeIsExist variable in client side and that get from the server. If the existed status is not the same (e.g. card type number 1 is not exist in client variable but it is existed in server variable), the program would update the table to display that card.

|  |
| --- |
| // Update played card on table  for (var i = 0; i < cardTypeIsExist\_temp.length; i++) {  if (cardTypeIsExist[i] != true && cardTypeIsExist\_temp[i] == true) {  for (var j = 0; j < playerCount; j++) {  for (var k = 0; k < players[j].length; k++) {  if (players[j][k].typeId == i) {  ui.addCardToRubblishArea2(players[j][k]);  cardTypeIsExist[i] = true;  return;  }  }  }  }  }  } |

**8. Conclusion**

In this project, we have completed the whole web-based version of “Zero Pollution” board game. During this project, we can learn how to get start in doing a project from nothing, system analyzing, code programming and interface designing included.

In the part of system analysis, we have analysis current situation of as-is system, making some use cases and functional requirements. In the part of system design, we have produced system structure and a number of interface designs in different actions that users have made for reader reference. We have also learnt how to select appropriate software or coding packages to help developing the program. We need to how to make a document with some clear explanations and guidelines.

Besides, we have also faced different challenges which make some development processes are being encountered obstacles such as timing, communication between server and clients, animation synchronization and creativity. We have to solve different problems found in the development phrase within a limited time. Although we have solved those problems lastly, there is a waste of too much time in problem solving. Thus, we have learnt more about the way of how to solve problem in a limited time.

**9. Appendices**

**9.1 Game Rules**

In order to implement the game to become funnier, the web-based Zero Pollution will change or add some of the game rules to the game. The following instruction is the modified version of the rule based on the printed version of Zero Pollution.

**About the game**

Every year, there are over six million municipal solid wastes produced in Hong Kong. In other words, over 18,000 tons of municipal solid wastes are produced every day, which is equivalent to six standard swimming pools. Although the pollution problems are becoming more serious, we can save the Earth! By changing your lifestyles, classifying the waste and reducing the trash.

**You can save the environment! ACT NOW!**

**Your Mission**

Your mission is to collect and classify the Trash Cards, and then place them in different Recycling Bins. By using the Trash Cards and Function Cards, you can convert the Trash into Energy Crystal! Through cooperating and competing with your opponents, you can reach the final goal – convert waste to energy! Let’s create a better world together.

**Game Cards**

**72 Trash Cards**: 20 Metal Cards, 20 Paper Cards, 20 Plastics Cards, 6 Battery Cards, 6 Old Stuff Cards and 6 Leftover Cards.

**22 Function Cards**: 4 Incinerator Cards, 3 Contention Cards, 2 High Technology Cards, 2 Junk Retrieving Cards, 3 Resources Allocation Card, 1 Conversion Card,2 Landfill Transfer Card and 3 Junk Collection Card and 2 illegal Dumping Card.

**5 Trash Cumulation Cards**: 2 Landfill Full Cards and 3 Incinerator Fail Cards.

**12 Landfill Cards**: 6 Landfill and 6 Cover Cards.

**Set Up**

1. Deal the Landfill Card.
2. Put the Trash Cards, the Function Cards and Trash Cumulation Cards, together and shuffle. Deal 5 to each player, face down.
3. Put the remaining cards face down in the center to create the draw pile.
4. Prepare the Energy Crystals. (10pieces for two players, 17 pieces for three players, 15 pieces for players, 17 pieces for five players and 19 for six players)
5. Decide who goes first (you may start from the youngest player).

**Card Details**

**Landfill Cards**

Record the pollution level of the Landfill

**Function Card**

Incinerator

Deduct one pollution point from the Landfill

Landfill Transfer

Every player transfers his/her Landfill to the next player in anti-clockwise direction

Contention

Get 1 Energy Crystal from one of the other players, but increase 1 pollution point of your Landfill

High Technology

Used as one type of Trash Cards

Junk Retrieving

Get a card from one of the players

Resource Allocation

Get 3 cards from the draw pile, choose one card for yourself and distribute the remaining cards to other players

Conversion

Convert every 2 points of Landfill pollution into Energy crystal (at most you can convert 4 points)

Junk Collection  
Get 2 cards from the draw pile

Illegal Dumping  
Disable one opponent Function Card

**Trash Cumulation Cards**Incinerator Failure  
Increase one point in the Landfill of each player  
Landfill Full  
Increase two points in the Landfill of each player

**Game Process**

1. Distribute one Landfill Card and one Cover Card to each player. The Cover card covers the Landfill Card for the calculation of pollution level.
2. Start the game from the first player. Other players play the game in clockwise direction.
3. Play the cards in your hand. Every player can play 1 to 3 cards (at least one). If a trash card is played, it is put on the corresponding position on the board. The played card must be different from the cards currently on the board. If a Function Card is played, follow the instruction on that card.
4. Collect the Trash Cards in the Recycle Bin. The player who collects a whole set of trash can get one Energy Crystal.
5. If a player cannot play any card in a turn, the player shoes all the cards in hand to all players, chooses at most 2 cards in hand to discard and draw the same number of cards. Then, the next player started.
6. Take two cards from the draw pile each round and put it in the cards in your hand. Then, the next player plays. If you have more than seven cards in your hand at the end of your turn, each extra card will increase the stated pollution level of the Landfill.
7. When a player gets a trash cumulation card, show it immediately and all players increase the stated pollution level.
8. If a player has no cards in hand, draw 5 cards immediately. Then, the next player plays.
9. If all cards are drawn in the draw pile, collect all discarded Trash cards, Function cards and Trash cumulation cards and shuffle.

**When will the Game End**

When all Energy Crystals are distributed, the game ends.

**How to win**

For every pollution point a player has, remove one Energy crystal. The player who has the most Energy Crystals wins!

If players have the same number of Energy Crystals, the player who has lower pollution points wins.

If more than one player has the same number of Energy Crystals, it is a draw game.

**9.2 References**

|  |
| --- |
| **\* sorted by alphabetical order** |
| **\*\* bold parts are the references of "environmental tips"** |
| admin. (2009, Nov 30). Free Buttons To Use On Your Website. FreeBannersandbuttons.com. Retrieved from http://www.freebannersandbuttons.com/free-buttons.htm |
| Adobe.com. (2012). Adobe Creative Suite 6 Design & Web Premium. Adobe. Retrieved from http://shop.adobe.com/store/adbehap/zh\_HK/DisplayProductDetailsPage/ThemeID.25225600/productID.247439100 |
| Autodesk. (2013). Autodesk 3ds Max Products. Autodesk. Retrieved from http://usa.autodesk.com/3ds-max/how-to-buy/ |
| Azharuddin@TWC. (2012, March). 5 Open Source Database: Overview, Comparison and Features. The Windows Club. Retrieved from http://www.thewindowsclub.com/open-source-database |
| Ballard, P., and Moncur, M. (2009). Sams teach yourself Ajax, JavaScript, and PHP all in one. Indianapolis, Ind.: Sams Pub. |
| Blanchard, B. S., & Fabrycky, W. J. (2006). Systems engineering and analysis (4th ed.). New Jersey: Prentice Hall. |
| Botelho, C., Leung, P., Wan, C., & Wong, C. (2012). Zero Pollution [board game]. HK: People On Board. |
| **Baidu. (n.d.). Cooking Oil. Baidu Baike. Retrieved from http://baike.baidu.com/view/1752448.htm** |
| Carey, P. (2012). New Perspectives on Creating Web Pages with HTML, XHTML and XML (2nd ed.). USA: Thomson Course Technology. |
| Connolly, T. and Begg, C., E. (2010). Database Systems: A Practical Approach to Design, Implementation and Management (5th e.d.). USA: Pearson. |
| Dalibor, D., Dvorski. (2007, Mar). INSTALLING, CONFIGURING, AND DEVELOPING WITH XAMPP. Skills Canada - Ontario. Retrieved from http://dalibor.dvorski.net/downloads/docs/InstallingConfiguringDevelopingWithXAMPP.pdf |
| Dell. (2012). Shop for Small Business & Office. Dell. Retrieved from http://www.dell.com/us/soho/p/?scat=root&~ck=mn&isredir=true |
| Dennis, A., Wixom, B, H., Roth, R, M. (2010). System Analysis and Design. (4th ed.). USA: John Wiley & Sons. |
| Derk. (2012, Apr 23). The 8-Step Guide To Interface Design for iPhone Games. Paladin. Retrieved from http://www.paladinstudios.com/2012/04/23/the-8-step-guide-to-interface-design-for-iphone-games/ |
| Desi, Q. (2013, Jan 22). Game UI By Example: A Crash Course in the Good and the Bad. gamedev.tutsplus.com. Retrieved from http://gamedev.tutsplus.com/tutorials/aesthetics/game-ui-by-example-a-crash-course-in-the-good-and-bad |
| **Digital Butter. (2011). What is PDP?. Plastic Disclosure Project. Retrieved from www.plasticdisclosure.org** |
| **Dictionary.com. (2013). Ruler. Dictionary.com. Retrieved from http://dictionary.reference.com/browse/ruler** |
| Dodge, B, J. (n.d.). First Steps in Board Game Design. sdsu.edu. Retrieved from http://edweb.sdsu.edu/courses/edtec670/boardgame/BoardGameDesign1.html |
| Downs, K. (2007, Dec 17). Database Skills: Complete Contents. The Database Programmer. Retrieved from http://database-programmer.blogspot.hk/2007/12/database-skills-complete-contents.html |
| eHow Contributor. (n.d.). How to Choose Database Software. eHow. Retrieved from http://www.ehow.com/how\_11478\_choose-database-software.html |
| **Environmental Protection Department. (2013). Environmental Protection department. Hong Kong Government. Retrieved from http://www.epd.gov.hk/epd/eindex.html** |
| **Gov.hk. (2013, Mar). Waste Reduction & Separation. Gov.hk. Retrieved from http://www.gov.hk/en/residents/environment/public/green/wastereduction.htm** |
| **Green Sense. (n.d.). 紙袋？膠袋？「環保袋」？？？. Greensense.org.hk. Retrieved from http://www.greensense.org.hk/plasticbag/subpage1h.php** |
| guest7af47. (2009, Feb 9). Principles Of Good Screen Design. slideshare.net. Retrieved from http://www.slideshare.net/guest7af47/principles-of-good-screen-design |
| Gupta, M. (2010, Nov 27). What are Data Models. SQL Server Training. Retrieved from www.sqlserver-training.com/what-are-data-models/ |
| H., W., Joshua. (n.d.). User account | Joshua W. Herring. Drupal. Retrieved from http://joshuawherring.com/?q=user/password |
| Hermida, L. (2007, Aug 27). Introduction to Database Systems, Data Modeling and SQL. EMBnet. Retrieved from http://ch.embnet.org/CoursEMBnet/Basel07\_II/Introduction%20to%20Database%20Systems,%20Data%20Modeling%20and%20SQL.pdf |
| **HKBU - School of Chinese Medicine - Medicinal Plant Images Database. (2007). Daucus carota L. Hong Kong Baptist University Library. Retrieved from http://libproject.hkbu.edu.hk/was40/detail?channelid=1288&lang=en&searchword=herb\_id=D00713** |
| **Home Affairs Department. (2004). Community Used Clothes Recycling Bank Scheme. Had.gov.hk. Retrieved from http://www.had.gov.hk/en/public\_services/com\_clo\_rec\_ban\_sch/** |
| **Hung Hom Society for Community Organization. (2009). 家居小百科. Hung Hom Society for Community Organization. Retrieved from http://hunghomsfco.com/home\_advice.html** |
| Ian Sommerville. (2011). Software Engineering (9th ed.). Boston, MA, USA: Pearson Education, Inc. |
| Indrayanto, A. (2010, Jun 22). Introduction to Browser-Based Online Game Design and Development. Retrieved from http://anemonesoft.com/files/tutorial-html/introduction\_to\_browser\_based\_online\_game\_design\_and\_development.html |
| Interactive Employment Service Labour Department. (n.d.). Job Salaries. Retrieved from http://www.jobs.gov.hk/1/0/webform/Default.aspx |
| jQuery. (2013). jQuery API Documentation. The jQuery Foundation. Retrieved from http://api.jquery.com/ |
| Kieras, D. (n.d.). User Interface Design for Games. University of Michigan. Retrieved from http://web.eecs.umich.edu/~soar/Classes/494/talks/User-interfaces.pdf |
| **Makita. (n.d.). RECYCLE! Your rechargeable batteries. MAKITA POWER TOOLS (HK) LIMITED. Retrieved from http://www.makita.com.hk/catalog/news\_info.php?nid=15&language=en** |
| Michael, V., Mannino. (2007). Database Design, Application Development, & Administration (3rd e.d.). USA: McGraw-Hill. |
| Microsoft. (2012). Microsoft SQL Server 2012 SP1. Microsoft. Retrieved from http://technet.microsoft.com/en-us/sqlserver/ff898410 |
| Moore, C. (n.d.). How to Create Your Own Game Server. eHow. Retrieved from http://www.ehow.com/how\_5016681\_create-own-game-server.html |
| Moseley, Erin. (n.d.). Technical Skills in System Analysis. eHow. Retrieved from http://www.ehow.com/list\_6218205\_technical-skills-system-analysis.html |
| msdn. (2012, Aug 2). Entity Data Model. Microsoft. Retrieved from http://msdn.microsoft.com/en-us/library/ee382825.aspx |
| MySQL. (2013). MySQL 5.6. MySQL. Retrieved from http://www.mysql.com |
| **news.gov. (2009, May 20). LCQ17: Reducing the amount of packaging materials. News.gov.hk. Retrieved from http://www.info.gov.hk/gia/general/200905/20/P200905200185.htm** |
| Oracle. (2013). Oracle Products and Services. Oracle. Retrieven from http://www.oracle.com/index.html |
| Ofni Systems. (2013). Functional Requirements. Ofni Systems. Retrieved from http://www.ofnisystems.com/services/validation/functional-requirements/ |
| Deital, P., Deital, H., and Deital, A. (2013). Internet & World Wide Web, How to Program (5th ed). USA: Pearson. |
| **Panasonic Taiwan Co. Ltd. (2007). 舊報紙被回收再生利用後會變成什麼？. Panasonic Taiwan Co. Ltd. Retrieved from http://discovery.panasonic.com.tw/science/library/lib10recy/l10007.html** |
| Prisantosao, Y. (2010, Jan 27). 30 AJAX Tutorials For Smart Web Developers. 1st Web Designer. Retrieved from http://www.1stwebdesigner.com/tutorials/ajax-tutorials-smart-web-developers |
| **recycle.epa. (n.d.). 廢容器 問與答. recycle.epa.gov.tw. Retrieved from http://recycle.epa.gov.tw/recycle/epa/ShowPage2.aspx?sno=30&subsno=106** |
| **recycling.org. (n.d.). 環保@助人店. 職工盟回收助人中心 recycling.org.hk. Retrieved from** **http://www.recycling.org.hk/object.htm** |
| **Rogers, S. (2007, Mar 31). Man and the Biological World. USA: Lightning Source Incorporated.** |
| Rosenberg, D., & Scott, K. (2001, Feb 1). Top Ten Use Case Mistakes. Dr Dobbs. Retrieved from http://www.drdobbs.com/top-ten-use-case-mistakes/184414701 |
| Rouse, M. (2010, Aug). data modeling. Search Data Management. Retrieved from http://searchdatamanagement.techtarget.com/definition/data-modeling |
| Salen, K., Zimmerman, E. (2004). Rules of Play: Game Design Fundamentals. USA: MIT Press |
| Seidler, O., K. (2013, Mar 1). Apache Friends - Xampp. Apachefriends.org. Retrieved from http://www.apachefriends.org/en/xampp.html |
| shirock. (2007, Jan 25). Regular Expression (RegExp) in JavaScript. 遊手好閒的石頭成的部落格. Retrieved from http://blog.roodo.com/rocksaying/archives/2670695.html#comments |
| Softwarecentral. (2010, April 16). Software Quality Assurance and Testing. Slideshare.net. Retrieved from http://www.slideshare.net/Softwarecentral/software-quality-assurance-and-testing |
| Stack Overflow. (2013). Stack Overflow Question. Stack exchange inc. Retrieved from http://www.clker.com/clipart-11768.html |
| Stoy, Ada. (2012, Feb 3). Learn the Basics of Functional Requirements. Bright Hub PM. Retrieved from http://www.brighthubpm.com/project-planning/11666-learn-the-basics-of-functional-requirements/ |
| **Tai Po Environmental Association. (n.d.). Advanced Recycling Community. Tai Po Environmental Association. Retrieved from http://www.taipoea.org.hk/arc/arc.htm** |
| **Tech Blind. (2004, Oct 20). 藍廢紙、黃鋁罐、啡膠樽. Human-Centered Design in Hong Kong. Retrieved from http://techblind.blogspot.hk/2004/10/blog-post\_109827717950030880.html** |
| **Tetra Pak. (2013). Carton Structure And Purpose. Tetra Pak® UK Ltd. Retrieved from http://www.tetrapakrecycling.co.uk/tp\_structure.asp** |
| Thvg. (n.d.). Movies Icon. VeryIcon. Retrieved from http://www.veryicon.com/icons/folder/shine-2/movies-58.html |
| troels. (n.d.). Comparison of different SQL implementations. arvin.dk. Retrieved from http://troels.arvin.dk/db/rdbms |
| **U&O你和地球. (2007). 一张纸的诞生过程背后. 中国素食文化传播网. Retrieved from http://www.veg520.com/html/200907/sushi\_124824684023511.html** |
| Villalobos, R. (2012, Apr 11). A JSON Tutorial. Getting started with JSON using JavaScript and jQuery. I View Source. Retrieved from http://iviewsource.com/codingtutorials/getting-started-with-javascript-object-notation-json-for-absolute-beginners |
| xiaozhenghua. (2009, May 25). 齒輪矢量圖. nipic. Retrieved from http://www.nipic.com/show/3/83/bc6c306f7d19b197.html |
| w3schools.com (n.d.). php. w3schools.com. Retrieved from http://www.w3schools.com/php/default.php |
| Wikipedia. (n.d.). Software requirements specification. Wikipedia. Retrieved from http://en.wikipedia.org/wiki/Software\_Requirements\_Specification |
| Zero Pollution. (2012, Jun 16). Zero Pollution「港‧環保」桌上遊戲 Facebook 專頁. Hong Kong Community College. Retrieved from https://www.facebook.com/ZeroPollutionBoardGame |
| Zero Pollution. (2012, Sep 9). 桌上遊戲 Zero Pollution「港‧環保」玩法介紹 Youtube 影片. Hong Kong Community College. Retrieved from http://www.youtube.com/watch?v=IdwtE2FKwvM |
| Zero Pollution. (2012, Oct 9). 20121006 港環保桌上遊戲@TVB「激優一族」專訪 Youtube 影片. Hong Kong Community College. Retrieved from http://www.youtube.com/watch?v=lLRzsy6GZ9Q |
| 大爺. (2011, May 19). 不負責任UML筆記- 使用案例圖 (use case diagram). 痞客邦 PIXNET. Retrieved from http://ckwsteven.pixnet.net/blog/post/28605355-%E4%B8%8D%E8%B2%A0%E8%B2%AC%E4%BB%BBuml%E7%AD%86%E8%A8%98--%E4%BD%BF%E7%94%A8%E6%A1%88%E4%BE%8B%E5%9C%96-%28use-case-diagram%29 |
| **火神学说. (2007). 甜橙. tcmlib.com. Retrieved from http://www.tcmlib.com/zy/html43/showdetail-343237312ce7949ce6a9992c7a79.html** |
| 平凡的世界. (2007, March 25). Document对象内容集合. ccvita.com. Retrieved from http://www.ccvita.com/80.html |
| **彭琬馨. (2012, Sep 11). 綠色塑膠 看不見的環境危機. 節能減排故事賞. Retrieved from http://savearth.nctu.edu.tw/index.php/environment/206-2012-09-11-16-23-30.html** |
| 飛爾酥 philsu. (n.d.). 50個超棒的AJAX應用教學 . 飛爾酥創意設計. Retrieved from http://philsu.tw/index.php/deisgn-treasures/inspiration/web-design/item/50-excellent-ajax-tutorials |
| 達MiNG. (2009, April 21). jQuery教學(1) - GridView/Table中使用Selectors. {明式}. Retrieved from http://tatmingstudio.blogspot.hk/2009/04/jquery1-gridviewtableselectors.html |

**9.3 Assumption**

In order to fulfill the main purpose of this project, which is to develop a web-based version of the board game called Zero pollution, we need to build up some business or system assumptions before development process. Here are the assumptions made to control development flow of program:

1. This project is assumed to be worked in the situation of bidding a project. It means that target reader of this project is company manager of Zero Pollution. We need to promote our development work and to write a detailed report to them.

2. The information of this document gives a detail design plan of the system, which can control the development flow of the system effectively with company managers, developers, programmers and users. As the whole structure of the system is not developed yet, since there may have some layout design, programming, database improvements, so that the final design may be slightly different than the plan in this document.

**9.4 Test Plan**

**9.4.1 Login**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test procedure** | **Expected Result** | **Pass/ Fail** | **Remark** |
| 1 | Press "Login" button to submit  information | Database update information | Pass |  |
| 2 | Sign up by correct ID and password | Users can login to the system | Pass |  |
| 3 | Sign up by correct ID and incorrect password | Users cannot login to the system | Pass |  |
| 4 | Sign up by incorrect ID and correct password | Users cannot login to the system | Pass |  |
| 5 | Press "Clear" button to clear  information | All data inputted will be cleared | Pass |  |
| 6 | Enter main menu after login successfully | Enter main menu | Pass |  |

**Table 9.4.1 – Login Test Plan**

Test by: Chan You Zhi Eugene, Fong Chi Fai Date:7-5-2013

**9.4.2 Register**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test procedure** | **Expected Result** | **Pass/ Fail** | **Remark** |
| 1 | Press "Register" button | Users can enter register page | Pass |  |
| 2 | Enter player’s name less than 5 character | Message of “Player name length less than 5” is shown | Pass |  |
| 3 | Enter player’s name only in number | No message is shown | Pass |  |
| 4 | Enter player’s name in number and characters | No message is shown | Pass |  |
| 5 | Enter player’s name only in characters | No message is shown | Pass |  |
| 6 | Enter player’s password less than 6 character | Message of “Password length less than 6” is shown | Pass |  |
| 7 | Enter player’s password in number and characters | No Message is shown | Pass |  |
| 8 | Enter player’s password only in characters | No Message is shown | Pass |  |
| 9 | Confirm player’s password with correct password | No message is shown | Pass |  |
| 10 | Confirm player’s password with incorrect password | Message of “Not same as password” is shown | Pass |  |
| 11 | Enter email with “@” | No message is shown | Pass |  |
| 12 | Enter email with no “@” | ‘@’ sign required | Pass |  |
| 13 | Register with no real name | Register successfully | Pass |  |
| 14 | Register with no phone number | Register successfully | Pass |  |
| 15 | Register with no home, address | Register successfully | Pass |  |
| 16 | Register with real name phone number and home address | Register successfully | Pass |  |
| 17 | Successful registration is being updated in database | Update successfully | Pass |  |
| 18 | Press "Clear" button to clear  information | All data inputted will be cleared | Pass |  |

**Table 9.4.2 – Register Test Plan**

Test by: So Chun Kit, So Tik Hang Date:7-5-2013

**9.4.3 Setting**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test procedure** | **Expected Result** | **Pass/ Fail** | **Remark** |
| 1 | Press “New Game” button | Start a new game | Pass |  |
| 2 | Press “Home” button | Back to main menu | Pass |  |
| 3 | Press “Profile” button | Enter the Profile page | Pass |  |
| 4 | Press “Chat Room” button | Enter the Chat Room page | Pass |  |
| 5 | Press “About” button | Enter the About page | Pass |  |
| 6 | Press “Credit” button | Enter the Credit page | Pass |  |
| 7 | Press “Music ON/ OFF” button | Turn on/OFF the music | Pass |  |
| 8 | Press “Logout” button | Logout and back to login page | Pass |  |

**Table 9.4.3 – Setting Test Plan**

Test by: Wong Ka Wai, Yeung Chi Shing Date:7-5-2013

**9.4.4 Single player and Multiplayer gameplay**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test procedure** | **Expected Result** | **Pass/ Fail** | **Remark** |
| 1 | Environmental Tips are shown when mouse on the card | Show the Environmental Tips | Pass |  |
| 2 | Card can played when mouse clicked | Card can played into right area | Pass |  |
| 3 | End turn by clicking “End Turn “ button | End the player’s turn | Pass |  |
| 4 | The description of functional card is shown when mouse on the card | Show the description of the functional card | Pass |  |
| 5 | Functional card can be active when it has been played | All functional card function well | Pass |  |
| 6 | The system will draw two card to a player when there is no card can be played | Draw two card to the user and end the player’s turn | Pass |  |
| 7 | The number of “Energy Crystal” will be increase when a collection of card is collected | The number of “Energy Crystal” increased | Pass |  |
| 8 | The number of card of a player is unlimited. | Player can have unlimited cards | Pass |  |
| 9 | The System will shuffle the card in discard pile back to the stock pile | There is enough card in stock pile all the time | Pass |  |
| 10 | A player will be a winner when he or she has the largest amount of “Energy Crystal” | The winner can be shown correctly | Pass |  |
| 11 | The game result will be display automatically when the game is ended | The game result will be displayed | Pass |  |
| 12 | The game board will be freeze when the game is ended | The game board is freeze | Pass |  |

**Table 9.4.4 – Single player and Multiplayer gameplay Test Plan**

Test by: Chan You Zhi Eugene, Fong Chi Fai, So Chun Kit Date:7-5-2013

**9.4.5 Chat room**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test procedure** | **Expected Result** | **Pass/ Fail** | **Remark** |
| 1 | The time of the message is match to the send out time | The time is correct | Pass |  |
| 2 | The previous message can be shown in the chat room | Message is shown | Pass |  |
| 3 | Player can be able to see the message send out by others | The message will be follow with the sender’s name and the message is correctly | Pass |  |
| 4 | The order of the message is correct | The order is correct | Pass |  |

**Table 9.4.5 – Chat room Test Plan**

Test by: So Tik Hang, Wong Ka Wai, Yeung Chi Shing Date:7-5-2013