Forsaken City

– COMP208 Group 53
Requirement Document

Group members:
Zeyu Bao; Yiyi Ju; Linshan Li; Zhaoxi Liu; Yunfan Sun; Xin Yan

Contents

1 Project Description

1.1 Target Audience

This game is rated as 12+ in PEGI rating system due to the violence elements and dirty words. This game aims at survival and zombie game fans.

After getting the license and internal testing, the game will be put on the "Google Play" store where users can pay 3.99 pounds for it and download. As a Buy to Play(B2P) mode game, the purchaser has the permanent ownership of the game without extra payment. After the game released, according to user feedback, there will be Downloadable Contents (DLC) to expand game content if users wish.

1.2 Mission Statement

The "Forsaken City" is a single-player mobile game which utilizes the most value of fragmentation time. It allows users to spend their spare time, it serves as a decision-making entertainment game as well, which encourages users to make just decisions to survive in a harsh environment with promoting the development of plots.

1.3 Mission Objectives

Essential

- Start of game
  - Have an instruction about the game for new users
  - Allow users to access and create archives
  - Introduce the background of the game at the beginning
  - Set a game leaderboard which displays top 20 players' scores in descending order
- While playing
  - Appropriate sound effects and containing switch to turn on/off the background music.
  - Simple and clear game interface
  - Pause, save and resume function
  - Recognizable symbol of the four attributes
  - Fluent card animation and dynamic properties value changes illustration
  - Involve plot into the game design perfectly
  - At least 10 different artworks of card.
  - At least one algorithm using in game balancing.

Desirable

- Achievement system.
- Loading scenes and tips
- Pictures for different endings
- Contain a bonus 2D pixel Coaster Rider game (if the value of special attribute reaches top)
- Import shooting function into the bonus game.
- A switch to choose whether decide randomly by system after 15 seconds idle.

2 Statement of Deliverables

2.1 Description of Anticipated Documentation

The character of the game is a survivor of the apocalypse trapped in a city by zombies. There will be different realistic scenario, and what users need to do is measuring the situation and making decisions by swiping the card left or right. Different choice will lead to different variations on properties, when any property reaches its bottom or top, the game will end. In any case, in order to survive, the first step is to learn how to make right decisions. Only controls all properties in dynamic balance can survival be achieved, otherwise there will lead to endless trouble.

2.2 Description of Anticipated Software

The users need to set a new archive or load exist archives if they want to start the game, the archive is used for storing player's information and history records and saving game progress.

Swipe the card left or right and hold, you can see two different strategies to a problem and which properties will be affected. Your decision will affect some of the attributes including "Health", "Resource", "Weapon", "Mental" and a bonus property "Kindness"; however, the system manages to lead player into a bad situation and make player fails after about 30 rounds. A background music will be set, and there will be an "Option" button in the bottom of the game interface to turn off/on the sound and control game process.

As for the desirable features, the bonus attribute will change if you choose to help others or not; when the special attribute reaches it's top, a 2D shooting bonus game that leads to hidden endings will be triggered, the "Health" and "Weapon" you get in the first game will be substituted into the following game.

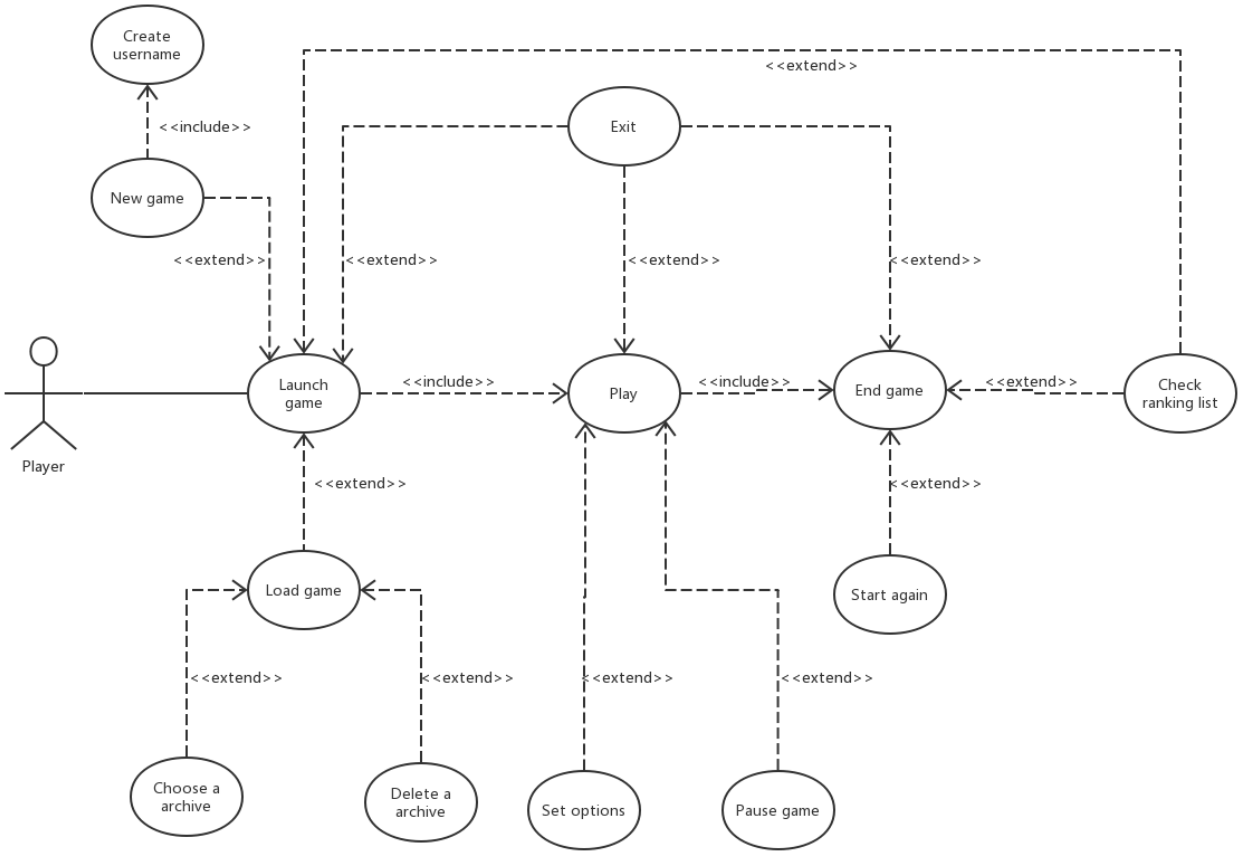
2.3 Description of Anticipated Experiment

To test whether the game is bug-free:

- A tester should download the installation package of game and install it on his smart phone. This can check the integrity of game.
- The tester should run the game and watch the background story displayed on the screen. This can check whether the game is executable and background story can be displayed.
- The tester should touch each functional button on the homepage, such as the button for the leaderboard, option, new game, load archive and the information about the game. This can check whether the buttons are functional.
- The tester should start a new game, create his own name. This can check whether the game can be started without bugs.
- During playing the game, the tester should test whether the pause button, option button, and save buttons are functional.
- The tester should play the game and achieve each ending. The tester should also pay attention to the change of each property, and the animation for delivering cards, making decisions and displaying endings.
- The tester should load an existing game and test whether the game can be loaded.
- The tester should check whether the game scores can be recorded on the rank list.

2.4 Description of Methods for Evaluation

2.4.1 Use Case Diagram



2.4.2 Description of Use Case

ID	UC1
NAME	Launch Game
DESCRIPTION	Player will enter a main interface.
PRE-CONDITION	App is running
EVENT FLOW	
POST CONDITION	
INCLUDES	UC 7
EXTENSION POINTS	UC2; UC5; UC13
TRIGGER	Player open this app

ID	UC2
NAME	Load game
DESCRIPTION	Load game interface
PRE-CONDITION	App is running
EVENT FLOW	Player presses “CONTINUE” button and the page jumps to a list of archives.
POST CONDITION	Could choose the archive
INCLUDES	
EXTENSION POINTS	UC3; UC4
TRIGGER	Player press “CONTINUE” button

ID	UC3
NAME	Choose an archive
DESCRIPTION	Allow player to choose an archive to play
PRE-CONDITION	App is running
EVENT FLOW	Choose and click an archive
POST CONDITION	Game enters an interface waits for player to choose
INCLUDES	
EXTENSION POINTS	
TRIGGER	Player clicks a archive

ID	UC4
NAME	Delete an archive
DESCRIPTION	Allow player to delete an archive
PRE-CONDITION	App is running
EVENT FLOW	Choose an archive to delete
POST CONDITION	Game enters an interface waits for player to choose
INCLUDES	
EXTENSION POINTS	
TRIGGER	Player press “delete” button

ID	UC5
NAME	New game
DESCRIPTION	Allow player to start a new game
PRE-CONDITION	App is running
EVENT FLOW	1. Include UC 6 2. Start playing
POST CONDITION	Game enters an interface waits for player to choose
INCLUDES	UC6
EXTENSION POINTS	
TRIGGER	Player presses “NEW GAME” button

ID	UC6
NAME	Create username
DESCRIPTION	Player will create a username.
PRE-CONDITION	App is running
EVENT FLOW	Player creates a new username
POST CONDITION	Player in a new game
INCLUDES	
EXTENSION POINTS	
TRIGGER	

ID	UC7
NAME	Play
DESCRIPTION	Player will enter a play interface.
PRE-CONDITION	App is running
EVENT FLOW	Player enters playing interface and plays
POST CONDITION	Player chooses an archive to play
INCLUDES	UC10
EXTENSION POINTS	UC8 UC9 UC12
TRIGGER	Player chooses an archive or starts a new game.

ID	UC8
NAME	Pause game
DESCRIPTION	Allow player to pause game without exiting app.
PRE-CONDITION	App is running
EVENT FLOW	1. Player chooses an archive successfully 2. Player enters playing interface 3. Player presses “pause” button
POST CONDITION	Player is playing game
INCLUDES	
EXTENSION POINTS	
TRIGGER	Player press “pause” button during game

ID	UC9
NAME	Set options
DESCRIPTION	Allow player to set game options like volume
PRE-CONDITION	App is running
EVENT FLOW	Player can set options like volume of sound
POST CONDITION	Player is playing game
INCLUDES	
EXTENSION POINTS	
TRIGGER	Player press “set” button in the game

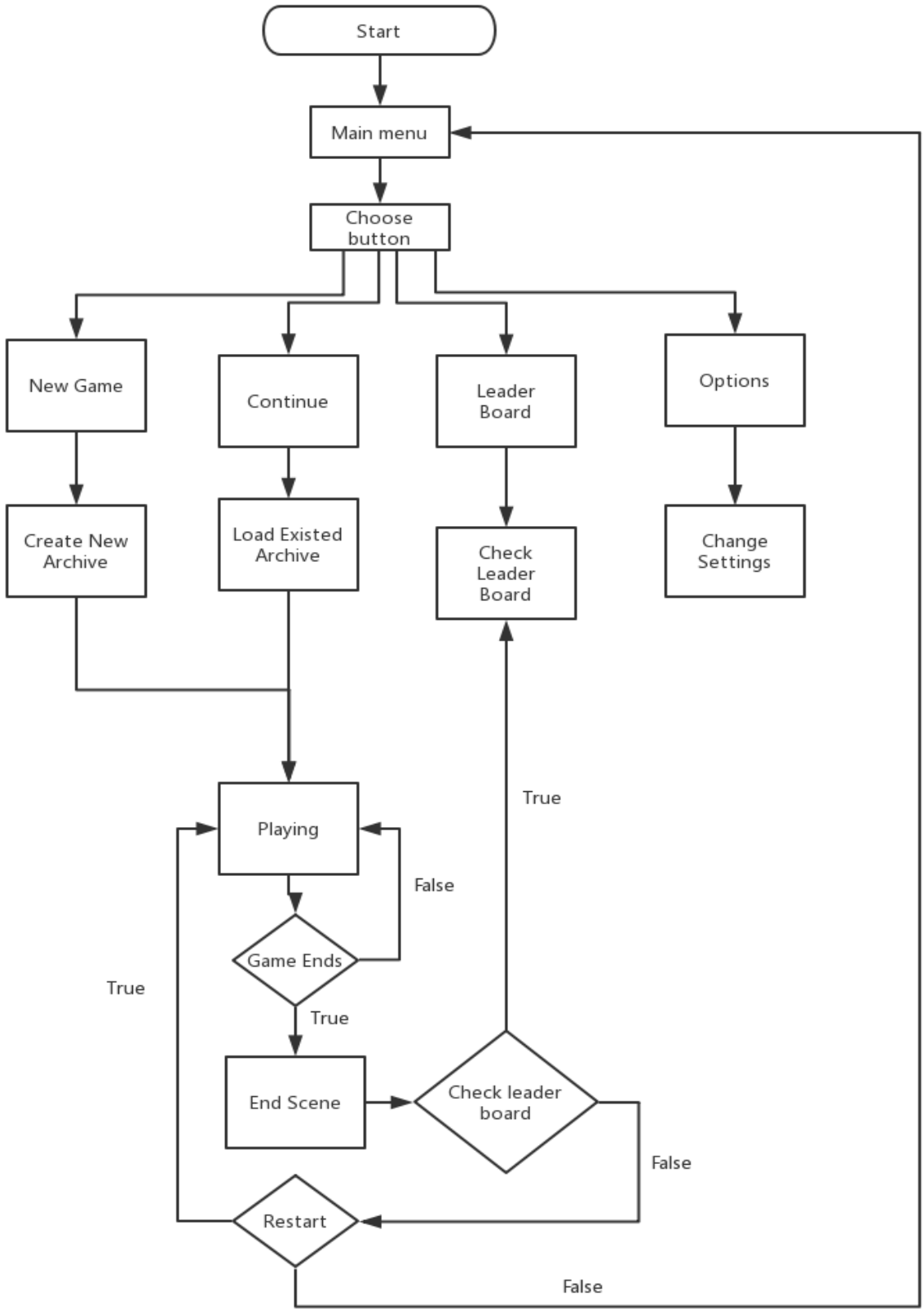
ID	UC10
NAME	End game
DESCRIPTION	End Interface.
PRE-CONDITION	App is running
EVENT FLOW	When any attribute equals to zero, the game will end.
POST CONDITION	Game calculates the number of player’s survival days.
INCLUDES	
EXTENSION POINTS	UC11; UC12; UC13
TRIGGER	Any attribute is equal to 0

ID	UC11
NAME	Start again
DESCRIPTION	Allow player to play again.
PRE-CONDITION	App is running
EVENT FLOW	If the game ends, the screen will show "Do you want start again?" If click "Yes," game restarts. Else, the game goes back to main interface.
POST CONDITION	Game ends or game restart
INCLUDES	
EXTENSION POINTS	
TRIGGER	Player presses "Yes" or "No" button

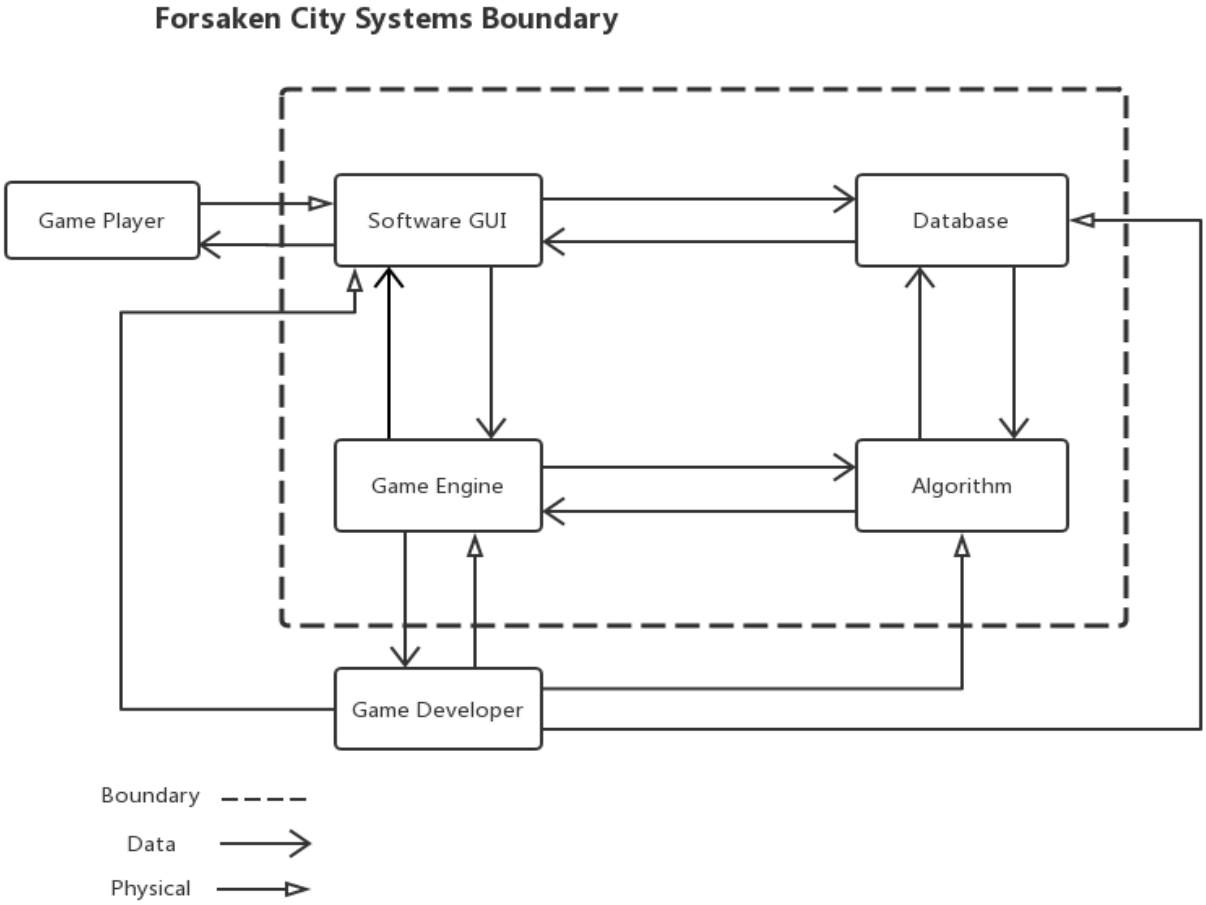
ID	UC12
NAME	Check leaderboard
DESCRIPTION	Allow player to check leaderboard.
PRE-CONDITION	App is running
EVENT FLOW	1. Player presses "leaderboard" button 2. Player enters leaderboard interface
POST CONDITION	
INCLUDES	
EXTENSION POINTS	
TRIGGER	Player press "leaderboard" button

ID	UC13
NAME	Exit
DESCRIPTION	Allow player to exit this app at these three statements.
PRE-CONDITION	App is running
EVENT FLOW	1. Player press "Exit" button shown in the interface 2. Game shut down.
POST CONDITION	
INCLUDES	
EXTENSION POINTS	
TRIGGER	Player presses "exit" button

2.4.3 Flow Chart



2.4.4 System Boundary Diagram



2.4.5 Functional Requirement

1) Set player's name

When player decides to start a new game, he/she should set a username. The username can be duplicated with previous ones.

Test method:

Create a new name and check whether this action is successful. Then create a name which is same as previous. Check whether this action is valid or not.

2) Pause

This requirement is set for letting user could pause the process during game. In the game interface, the game designer should provide an icon to pause the game. Once the icon been pushed, the game will be paused.

Test method:

Enter playing interface, press "pause" button. Check whether whole process is paused.

3) Main menu

At the beginning of the game, a main menu should be provided to users. This menu provides users with continue game(choose archive), start new game and change settings functions.

Test method:

Launch this program, check whether there is a main menu. Press any buttons of this menu.

4) Load game

This requirement is set for users to continue their last game. This project needs to set archive's unique id to identify archive's information.

Test method:

Enter main menu, press "continue" button. Check whether there are archives player saved before.

5) Sound effect

To enhance the interests of the game, game designer set a background music, there will be a switch on the lower right corner of the game interface to choose whether turn off the sound.

Test method:

Press "option" button. Check whether the volume is changed as desire.

6) GUI

A well-designed and tidy game interface can motivate users' desire to play the game. Red is the main color creates an atmosphere of tense. Besides, a variety of images of icons and buttons have been set for providing players with more convenient operation.

Test method:

Get a few players to experience the game and ask them how they feel.

2.4.6 Non-Functional Requirement

1) Security

This non-functional requirement is set for protecting document and data. The game is an offline game, all the game information will be stored in the local device, and all the game files are read only and cannot be accessed by user as well.

Test method:

Confirm whether this document could edit by anyone who do not have the authority.

2) Usability

This non-functional requirement is set for making users feel easy to play this game. There is an instruction when new user launches the game, therefore, a user who play this game for the first time shall be familiar with this game after 5 minutes .

Test method:

Invite serval testers to play this game. Record their average time of being familiar of this game. Programmer should collect their opinions after testing and improve their program based on test results.

3) Performance

This non-functional requirement is set for making game more efficient. Suppose there are more than one archives on a device. Then the game shall get data in 2 seconds successfully.

Test method:

Do “load” action for 20 times for different archives. Then record the average time of loading archive successfully.

4) Capacity

This non-functional requirement is set for requiring this software have plenty of space for update users’ data. The database should has the capacity of at least 3 archives. Moreover, the leaderboard can display at most top 20 users’ scores. (descending)

Test method:

Let players create 3 different archives. Then check whether this program could load 3 different archives correctly. Then set 20 records in the leaderboard database. Check whether these 20 records can be shown correctly in leaderboard. (descending)

5) Reliability

This non-functional requirement is set for requiring the system maintains its functions for a long time. The game can be played any time in a day. The time of game crashing should not exceed 2 times in 100 tests.

Test method:

Run this program for 100 times. Record times of crash.

6) Recoverability

This non-functional requirement is set for protecting game data and players’ previous data. The recovery of failure won’t affect the data stored in the system before.

Test method:

Run this game and force it shutdown. Then check whether user's data and archive are destroyed by this action. Repeat this test for 10 times.

2.4.7 Constraint

As a group project, it would be more efficient and effective to group sources code with the readable, well-formatted and elegant code. This is accomplished by constraints like proper naming conventions and statement formatting. Therefore, the project will consider the soft and hard constraints that would need to be adhered to.

Soft constraint

1) Naming convention: using lowercase with hyphens separating words
2) Have clearly designed framework: based on framework, implement the service module and the monitoring interface module.
3) Try to minimize the response time.
4) Try to optimize reading data time from database tables.

Hard constrain

1) Each choice must trigger the next card or ending.
2) After a pause, suspension, interrupt or hibernating, the data of the player is not lost.

3 Conduct of Project and Plan

3.1 Preparation

3.1.1 Background Research

This project aims to make a complete game that uses Unity 3D [1] as engine and uses C# as the game development language. Using merge sort algorithm to order players according to their number of survival days in the leaderboard. Besides, the narrative will encourage the player to stay on the golden path [2] without unnatural felling. The game must comply with the soft constrains and the hard constrains where the hard constraints must be adhered to or else the game won't start or crash.

Based on trigger mechanism, this game will not contain some complex algorithms to optimize.

Unity 3D is a free game engine, which provides theoretical background on topics

like level design, interface design and tools needed. Group members will become familiar with core design method in it and design and implement the project base on the narrative.

Golden path is an essential principle of designing game, which is an optimum path a player would take through the game in order to experience the game as intended and to experience the maximum rewards. The game includes completely story and balance attributes in each branch. Above all, the narrative makes sure each branch could go back to the main line to continue.

3.1.2 Data Required:

1) The attributes of player including health, mental, resources and weapon
2) All the cards contend
3) All the players names
4) The number of days the player survives
5) Players rating of the rank list, based on survival days
6) Background music
7) Volume of sound

3.2 Design Stage

The main methods of the project software design combine Object-Oriented method and Agile method [3] [4].

Agile

The method used is Scrum:
Scrum Team: Members of Development Department
Scrum Master: Members of Planning Department

Weekly meeting

There is a weekly meeting which every member talk about the progress has been achieved since last meeting, what tasks will be finished before next meeting and problems in the work.

Product analysis

Involving all group members and test the product by testing department. The members of testing department give advice and improve suggestion. After analysis, the planning department gives improved methods and requirements to development department and adjust the product.

Task assignment

After deciding tasks, making a timetable to arrange deadline for each task. Each member picks up tasks they want. During the process of completing the task, each member could update task information.

Objective-Oriented

Using bottom-up design, solve the problem of each component first then combine all the components together to get a complete subsystem. After iterations, combine all the subsystems into a complete system.

Required documents

1) Class diagrams to demonstrate relations between different entities of the project and each entity's function and properties.
2) Pseudo codes of functions which including the pseudo codes of main functions, such as card generation, card swipe and scene change.
3) Data structures to store information of entities.
4) Subsystem division and the within components description.

3.3 Implementation Stage

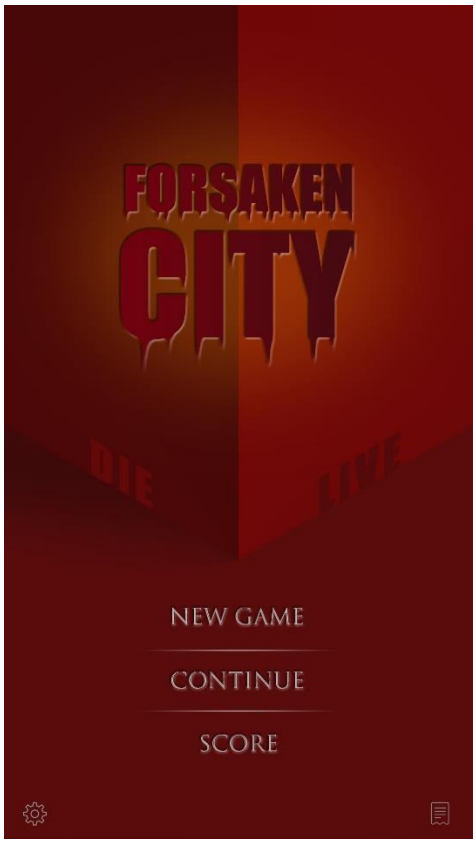
Software: Unity, Visual Studio, SQLite
Hardware: Android Smart Phone, PC

Test:

1) Code test: using IDE's automatic error check to correct syntax and logical errors.
2) Method test: test whether the input and output of each function conform to the design.
3) Component test: test functions, such as drag function, overturn function, etc
4) Sub-system test: test sub-systems, such as login system, card control system, etc.
5) Whole system test: test the whole system.

3.4 Design Preview [5] [6] [7] [8] [9] [10]

Home Page

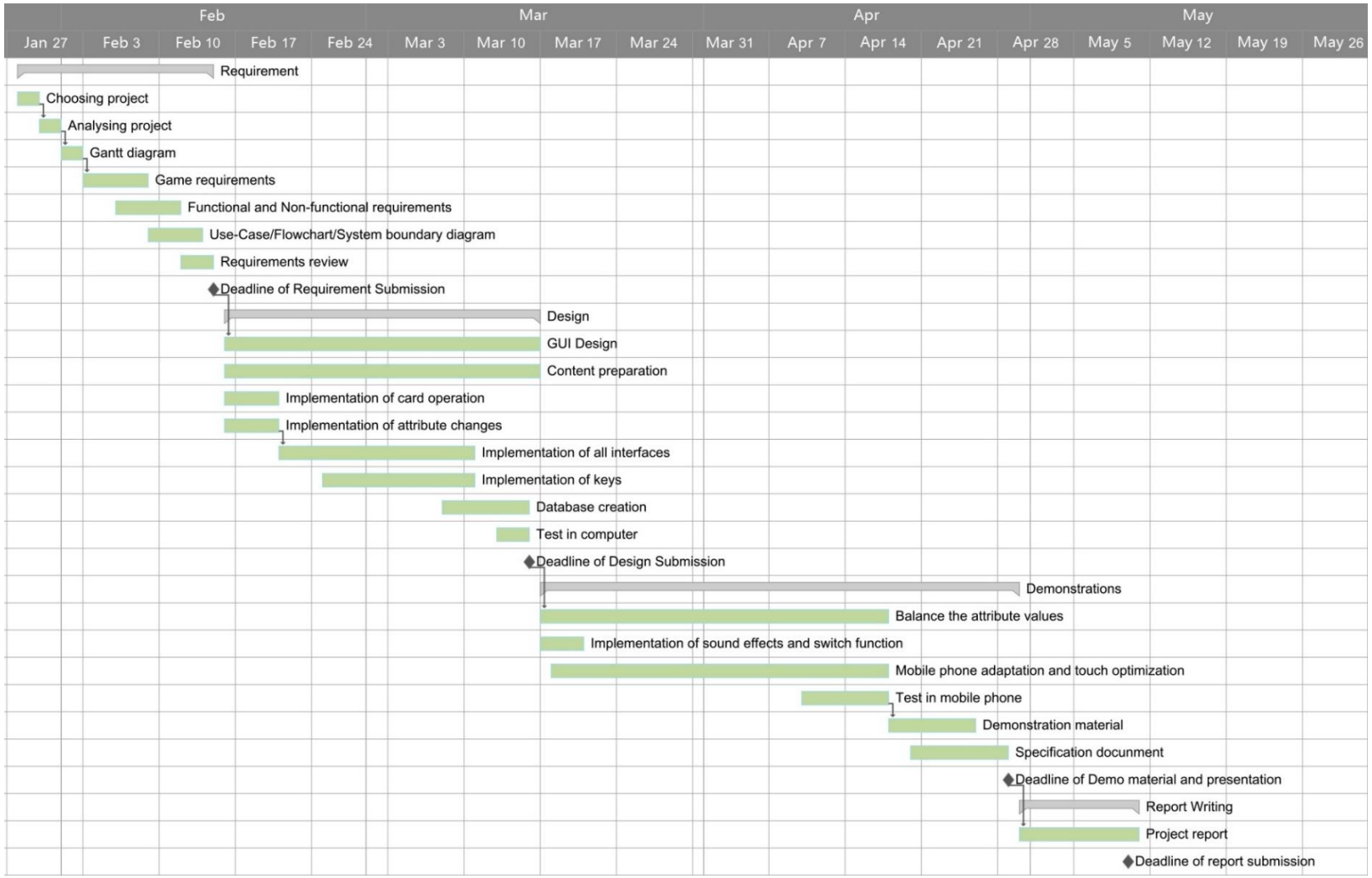


Rank list

Create new player

In the game

3.5 Gantt Chart



3.6 Risk Assessment

3.6.1 Major Challenges

1) Database failure:

Database failure that causes data loss or game crush.

2) Compatibility

Game may be incompatible with some devices. There will be problems such as flashbacks or games become uncontrollable.

3) Time Costs

Using unity 3D which is unfamiliar with programmers as game engine costs massive time to learn and master. The game implementation could also be challengeable.

4) Game interesting

The difficulty of the game is hard to balance, if the game is hard and no rewarding, player would be tired of it soon.

3.6.2 Possible Solution

1) To avoid data loss, using system log to record data changes and recover the data after failure.
2) Optimizing code and reduce the use of system memory which may improve performance on low-end devices.
3) After project design stage, some members transfer from requirement department to program department and share the load of development.
4) Provide demos to players and collect feedback to adjust the narrative logic. Add rewarding system such as level system or collectible items to improve game interesting.

4 Bibliography

[1] M. Jere, *Unity 3D and PlayMaker essentials [electronic book]: game development from concept to publishing* / Jere Miles, Boca Raton: CRC Press, 2016

[2] S. Rabin, *Introduction to game development*, Boston: Course Technology Cengage Learning, 2010.

[3] S. Christoph, *Agile software development teams [electronic book]: the impact of agile development on team performance* / Christoph Schmidt, New York: Springer, 2016

[4] Z. Ma, *"Introduction to Scrum"*, lecture note.

[5] (2013, June. 24) *Options* [Online]. Available: http://www.sucaibar.com/icon/7923.html

[6] (2013, June. 24) *Volume of sound* [Online]. Available: http://www.sucaibar.com/icon/7941.html

[7] (2018, April. 14) *Building* [Online], Available: http://599ku.com/sc/9894062.html

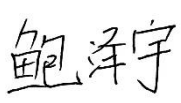
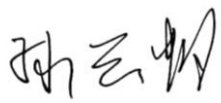
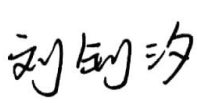
[8] *Heart* [Online], Available: http://www.mypng.cn/view/468594.html

[9] (2013, June.15) *Pistol* [Online], Available: http://www.haipic.com/item/9292/

[10] *Brain* [Online], Available: http://www.51miz.com/sucai/962976.html

Signatures of Group Members

Zeyu Bao, Yunfan Sun, Zhaoxi Liu

Yiyi Ju, Linshan Li, Xin Yan