

CSE 344 Software Engineering
Requirement Analysis Report

Covid Bird

by

Ann Nedime Neşe Rende
Aleyna Polat
Arda Ayvataş
Burcu Ece Kartal
Can Erdoğan
Temel Metehan Çınar

Yeditepe University
Faculty of Engineering
Department of Computer Engineering
Spring 2021

TABLE OF CONTENTS

1. Introduction.....	3
1. Purpose.....	3
2. Background.....	3
3. Motivation.....	4
1. Statement of problems with the existing system.....	4
2. The new system.....	5
4. Structure of the document.....	6
2. Functional Requirements.....	7
1. Description of the system functionalities.....	7
2. Description of the system users.....	7
3. Specific Requirements.....	8
1. Use Case Diagrams.....	8
2. Use Case Priority List.....	9
3. Use Case Specifications.....	10
3. Non-functional Requirements.....	21
1. Volere Templates.....	21
4. System Models.....	31
1. Object and class model.....	31
2. User interface.....	33
5. Definitions, Acronyms and Abbreviations.....	37
6. Glossary & References.....	37

1.INTRODUCTION

1.1. Purpose

This document aims to describe the functional and non-functional requirements of the game designed, as well as the basic design of the object and class model. Added to that, user interface elements are provided.

1.2. Background

While designing Covid Bird, we were mainly inspired by Flappy Bird. With the help of analyzing similar games existing in the market, we shaped our game to include the parts that the Flappy Bird was missing.

First of all, we analyzed Dinosaur Game which is included in Google Chrome web browser. The game was created by Sebastien Gabriel in 2014. It can be accessed by pressing the space bar while the user is offline in Google Chrome. Also, it is played with the spacebar and down arrow keys to avoid obstacles. The creation of the game refers to the prehistoric times offline. We were inspired by this and designed our game Covid Bird with a similar intention, to refer to Covid-19.

Another game that we checked is Jetpack Joyride which is an adventure game. It is a mobile game and was released in 2011. It was made in Australia and produced by Halfbrick Studios. The game aims to go to the farthest point by overcoming obstacles. One of the biggest helpers in increasing travel distance is the supplements collected. The chapters are changing and certain missions are completed. This way, you can have higher success in the game. Unlike Flappy Bird, graphics and supplements can be collected. It can be played on Android and iOS platforms.

Flappy Dunk is a game that combines Flappy Bird and basketball, published in 2017 by Voodoo. The game had more than 10,000,000 downloads on Google Play Store and it can be seen that it is still popular as it had more than 100,000 worldwide downloads, on iOS and Android platforms, in February 2021. Instead of going through pipes, the main goal is to get the winged ball through the loops as much as the player can. The way to control the ball is the same-by tapping on the screen. It also has 51 challenges, which makes it different from the simple Flappy Bird. It can be played on Android, iOS and Windows platforms.

Subway Surfers is a single-player endless runner mobile game. The game was developed by Kiloo and SYBO Games in 2012. It is developed using Unity game engine. It can be played on Android, iOS, Kindle, and Windows Phone platforms. The game had more than 1,000,000,000 downloads on Google Play Store in February 2019. The game has very easy gameplay. Players can swipe the screen up, down, left or right to move. Another thing that makes the game special is the animations. Subway Surfers' main goal is to avoid crashes and reach a high score. Players avoid crashing into subways, poles, walls and barriers. The game speeds up every second. If the player hits objects, the game is over.

Unlike the inspired Flappy Bird, whose developers were Mid Therox and Em Lazer-Walker and contributed by Zach Gage, Flappy Royale is a game where the last one out of 100 players wins without hitting a pipe. The game was released in July 2019 and while it was originally developed as an open-source web demo, it has also been developed for Android and iOS platforms, taking into account the likes and suggestions. Embodying the competitive atmosphere of the battle royale concept, the game features a free-for-all mode with up to 100 players and a daily top-score system to obtain which player passes the most pipes daily where players have the right to try at a limited number. The main character is a bird-like in the game Flappy Bird and starting with entering a username then the game is played by touching the screen or with the spacebar on the keyboard, and when the pipe is hit, the game ends and shows the player the ranking among 100 players.

Last but not least, Jetpack Pou is another game that we were inspired by. Pou was first released on August 23, 2012. Pou, which is developed by Paul Salameh, is a pet game that runs on BlackBerry, iOS and Android operating systems. There are also mini-games to achieve certain goals in Pou, where a triangular brown creature can be fed, cleaned, exercised and put to sleep. Pou means "bit" in French. The game is similar to Tamagotchi, which is about a creature trying to live in real-life conditions. The game is very popular. It is a game that appeals to children and the player group is also usually children. We are especially inspired by the gold that Pou collects while passing through column-like wood like the Flappy Bird.

1.3. Motivation

1.3.1. Statement of problems with the existing system

- A worldwide leaderboard does not exist, so users can not see the top scores in the world. As a consequence, users do not aim for a high score, and their will to play decreases.
- Flappy bird is not providing a user login system. People want to register themselves before playing the game. Because they may want to use a nickname or their original names that they prefer. People can be demoralized when they see themselves being named as player 1 player 2, and sometimes this situation can reduce their sympathy for the game.
- The path in the game is always at the same speed (the ordinary speed of colons). There is no difficulty level, and this situation means people have no difficulty playing. As a result, they can get bored of the game quickly. Games should not bother people because people play games to relieve stress or for fun.
- There are no items that can prevent the game from ending. Because of the problem, users can die easily. This problem can interfere with the fluency of the game. In a non-fluent game, users are bored after a while by the game also, the target score range will be lower. The problem will prevent the sustainability of the game.
- Flappy bird, Jetpack Joyride etc. games contain a pause button, but the games we inspired do not include Google Dino and Flappy Dunk. The problems caused by these games not including the stop button are as follows. The user may have to stop the game due to any situation during the game. The score may be lost. In this case, it is possible for the user to get bored with the game and not to prefer the game. In another case, if the user leaves the game again, his current score may be lost again.

- In the Flappy Royale game, in contrast to Flappy Bird, character customization was tried to be solved, except for the Flappy Bird's original bird, which can be changed without any prerequisites in the character customization section, skins such as cats, dogs, ghosts, different types of birds and hats and crowns that can be worn have been added. In this way, although the game has become more enjoyable, the size differences in the skins, the height of the hats and the skins of 100 players overlap in places, causing confusion and misleading, turning into an advantage or disadvantage for the player during the game.
- Both Flappy Bird and Flappy Dunk lack a detailed explanation of the game. In Flappy Bird, it only says, “tap” but doesn’t explain that the player has to go through the pipes or how the score will increase. The player has to learn that after trying and failing. In Flappy Dunk, the basic game rules are explained in the first play, but an option to read them later again doesn’t exist.
- There are not enough animations in Flappy Bird, Google Dinosaur Game and Jetpack Pou. Animations provide playability and visuality. It is also very necessary to be able to play the game continuously. That's why these games might not be appealing to everyone.
- There are not enough sound effects in Flappy Bird.

1.3.2. Our solutions to these problems

- With the addition of a leaderboard, a user can see the top scores all around the world. Therefore, the user can aim for a high score to see their username on the leaderboard and the game can get more competitive.
- We will put a login system that will allow people who want to put their own nickname or their own name. People want to register themselves before playing the game. Because they may want to use a nickname or their original names that they prefer. Sometimes some nicknames become so famous that people who use that name can be recognized around the world for their success. People sometimes want to know who they are competing with because each of them is a character. Also, when the system announces the winner, instead of giving names (such as player 1, player 2) players are more honored when they win with their nickname.
- We will put different levels of difficulty in the game, and the game will speed up over time, causing people to have difficulty playing. With this, people wonder how far they can go, which increases their sympathy for the game. Although there are people who say the opposite, as the difficulty and speed of the game increase, they become more ambitious and want to play more. This makes them love the game more.
- The user will not have a single life while playing the game. There will be an immunity bar as a can system. This immunity bar will be divided into features according to the items it collects. These features will fill the user's life or make the user immortal for a short time. In this way, the user will not die easily and users will reach high scores. will be able to use new skins. Thanks to the skins, the game will be out of uniformity. All this will be thanks to the items that will be added to the game.
- With the features we are inspired by the mentioned games, we can prevent the game from being stopped and losing the current score in our game in case the character of the user has any problems. The user can adapt to sustainability with the addition of this feature. For example, as a problem, if the user leaves the game, he will not lose his score
- In the character customization system, the player can open different characters by accumulating coins that can be opened on skins depending on his score each time he plays the game. With this situation, it is aimed to be able to play the game again and again. At the same

time, the hitboxes of the characters are kept fair so that the characters do not have advantages or disadvantages.

- When the user is in the main menu or the game is paused, there is a help button in our game to remind the user of the main rules and how the elements collected will result in an increase in their health percentage or gaining immortality for a short period of time. Moreover, different types of coronaviruses and how they will affect the health percentage are explained. In addition, there is another help button in the shop where the user can change their character's look.
- The issue regarding animation is solved by adding new animations to the game. For example, death animations, collision animations, character-specific animations, animations for items (vaccine, fruit and mask), cloud animations, morning-evening animations, motion animations. We will provide visuality and playability to our game with these animations. In addition, we will make it appeal to more mobile players with these animations.
- The problem about sound is solved by adding new sound effects in the game such as the sound when an item (a fruit, mask or vaccine) is collected or when the warning of the second wave comes up.
- Covid Bird has a real story from real life. Covid Bird's main goal is the player will try to avoid the Covid-19 virus-like in real life. Thus, Covid Bird not only has a true story but also creates awareness. This scenario behind the game adds a story to the game. Also with this story, unlike Flappy Bird, everything in Covid Bird has a reason. The player knows why to move the character because they encounter real-life things.

1.4. Structure of the document

Starting the document with the introduction, we analysed some of the similar games in the market, pointed out their problems and explained what we added into our game to solve them.

We have listed the functional requirements and mentioned the specific requirements such as use case diagram, use case priority list and use case specifications using models.

With the help of Volere templates, we have explained the non-functional requirements.

After that, we have made our system's object and class model and user interface, so that the system can be expressed more clearly.

In order to express the system more clearly, we included our system's object and class model and user interface.

We finished the document with adding definitions, acronyms, abbreviations, glossary and references.

2. FUNCTIONAL REQUIREMENTS

2.1. Description of the system functionalities

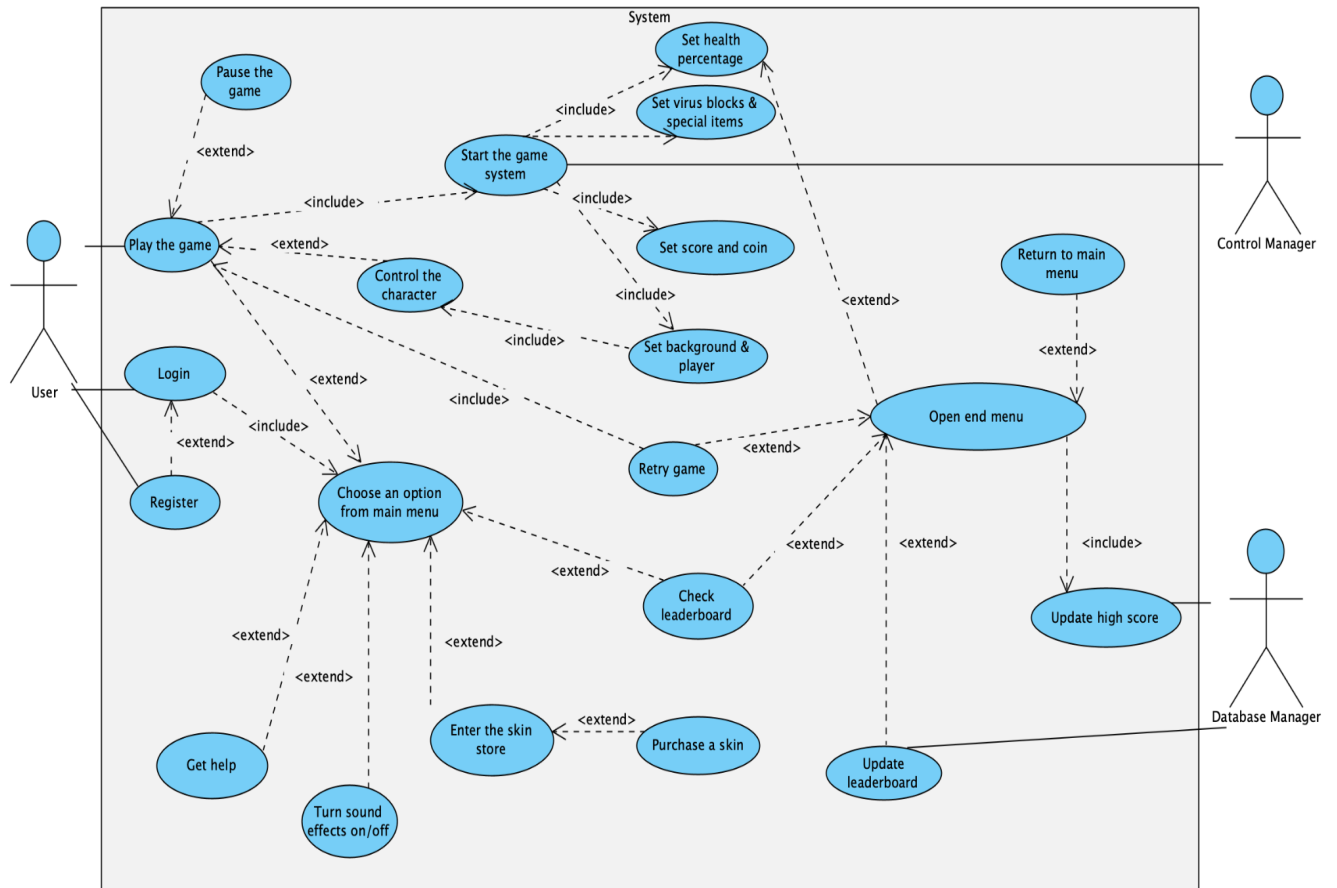
- 2.1.1:** The program shall have a GUI for user interaction.
- 2.1.2:** The program shall have a method of keeping user data.
- 2.1.3:** The program shall create a variety of elements randomly that increase the health percentage such as vaccines and fruits. Vaccines increase health by 100% and fruits increase it by 20%.
- 2.1.4:** The program shall have a sound button that can turn the sound effects on and off.
- 2.1.5:** The program shall increase the value of the y coordinate of the bird when the user presses space.
- 2.1.6:** The program shall stop the game when the P key on the keyboard is pressed.
- 2.1.7:** The program shall create coronavirus blocks repeatedly.
- 2.1.8:** The program shall delete coronavirus blocks, which it previously created, every 10 seconds.
- 2.1.9:** The program shall display the top 5 high scores when the leaderboard button is pressed.
- 2.1.10:** The program shall demonstrate how to play the game when the help button is clicked.
- 2.1.11:** The program shall ask for user information, and if the information is correct then the program shall be personalized after the user presses the login button.
- 2.1.12:** The program shall increase the speed of blocks every 15 seconds.
- 2.1.13:** The program shall change the background between night mode and day mode if the gameplay reaches 150 seconds.
- 2.1.14:** The program shall change the background between night mode and day mode over and over again for 5 seconds, every 500 seconds in the game.
- 2.1.15:** The program shall decrease health by 5% if the bird crashes the green virus blocks, 15% for yellow blocks, 20% for red blocks and 30% for purple blocks.
- 2.1.16:** The program shall start the game when the user presses the restart button.
- 2.1.17:** The program shall start the game when the user presses the play button on the main menu.
- 2.1.18:** The program shall create a mask randomly.
- 2.1.19:** The program shall provide immortality when the user collects a mask or a vaccine. If the object collected is a mask, then the player will be immortal for 3 seconds. Otherwise, if it is a vaccine, the duration of immortality will be 5 seconds.
- 2.1.20:** The program shall end the current game when the health is equal to 0%.
- 2.1.21:** The program shall display an end menu that contains “retry”, “leaderboard” and “main menu” options when the game ends.
- 2.1.22:** The program shall keep the total scores as gold for each user.
- 2.1.23:** The program shall show several skins when the user enters the skin store.
- 2.1.24:** The user should be able to change skins if the user's gold is equal to or higher than the skin's cost.
- 2.1.25:** The program shall decrease the amount of gold the user has by the cost of the skin when new skin is purchased.
- 2.1.26:** The program shall resume the game after the user presses the P key on the keyboard in the pause state.

2.2. Description of the system users

The system users of the game who love arcade/casual games with a simple structure, based on the reward system, score-based with details that do not tire the player.

2.3. Specific Requirements

2.3.1. Use Case Diagram



2.3.2 Use Case Priority List (1->Highest Priority, 10->Lowest Priority)

Use case	Priority	Explanation
Login	1	Must be done before playing the game
Register	1	Must be done before playing the game
Play the game	1	Must be done to start the game system
Pause the game	3	Needed to pause the game
Turn sound effects on and off	10	Could be done for improving the user experience
Get help	9	Could be done for learning how to play the game
Enter the skin store	6	Should be done for purchasing a skin
Purchase a skin	6	Should be done for specializing the characters
Control the character	1	Must be done to be able to play the game
Check the leaderboard	4	Needed for competition
Retry game	3	Needed for playing the game again without visiting the main menu
Choose an option from the main menu	2	Must be done for navigating in the game
Open end menu	2	Must be done for navigating in the game
Return to main menu	2	Must be done to get to main menu's options
Start the game system	1	Must be done to play the game
Set score & coin	1	Must be done to be able to purchase a skin in skin store
Set virus blocks & special items	1	Must be done for creating the game environment
Set health percentage	1	Must be done for ending the game
Set background & player	1	Must be done for creating the game environment
Update leaderboard	2	Must be done for displaying the current high scores
Update high score	2	Must be done for displaying the current data

2.3.3 Use Case Specifications

1) Login

Use case name	Login
Actor	User
Description	For users to log in to the game using their username and password.
Precondition	The user must open the game and should have been registered before.
Postcondition	The user will be logged in.
Main flow	<ol style="list-style-type: none">1. System shows a login page.2. User fills the username and password fields with the correct information.3. The user will be directed to the main menu. (inclusion point: Choosing an option from the main menu)
Alternative path	<ol style="list-style-type: none">2.a. -User enters the wrong username or password. -System displays a warning. -Use case resumes at step 1.

2) Register

Use case name	Register
Actor	User
Description	For users to register for the game by specifying a username and password.
Precondition	The user must open the game.
Postcondition	The user will have created an account.
Main flow	<ol style="list-style-type: none">1. System shows a register page.2. User fills the username and password fields according to his choice.3. The user will be directed to the login page. (extension point: Login)
Alternative path	<ol style="list-style-type: none">2.a. -User enters a username that is previously taken. -System displays a warning. -Use case resumes at step 1.

3) Play the game

Use case name	Play the game
Actor	User
Description	For users to play the game.
Precondition	The user must be logged in to the game and press the play button.
Postcondition	The user will play the game.
Main flow	<ol style="list-style-type: none">1. User presses the play game button.2. System starts the game system. (inclusion point: Start the game system)3. User controls the player by clicking the mouse. (extension point: Controlling the character)
Alternative path	<ol style="list-style-type: none">4.a. The user can pause the game by clicking the button. (extension point: Pause the game)

4) Pause the game

Use case name	Pausing the game
Actor	User
Description	For users to pause the game.
Precondition	The user must have started the game.
Postcondition	The user will pause the game.
Main flow	<ol style="list-style-type: none">1. User presses the P key on the keyboard.2. System pauses the game system.3. If the user presses the P key on the keyboard while the game is in progress System pauses the game. Else System resumes the game.
Alternative path	-

5) Turn sound effects on and off

Use case name	Turn sound effects on and off
Actor	User
Description	For users to turn the sound effects on and off.
Precondition	The user should be in the main menu.
Postcondition	The user will turn the sound effects on or off.
Main flow	<ol style="list-style-type: none"> 1. User presses the speaker button. 2. If the speaker button appears turned on state System turns off the sound effects and the the speaker button appears turned off. Else System turns on the sound effects and the the speaker button appears turned on.
Alternative path	-

6) Get help

Use case name	Getting help
Actor	User
Description	For users to get help about how to play the game.
Precondition	The user should be in the main menu.
Postcondition	The user can see and learn how to play the game.
Main flow	<ol style="list-style-type: none"> 1. The user clicks the 'Help' button. 2. The system shows the rules.
Alternative path	-

7) Enter the skin store

Use case name	Entering the skin store
Actor	User
Description	For the user to enter the skin store and wear the skin what the user has.
Precondition	The user should be in the main menu.
Postcondition	The user can see the available and previously purchased skins.
Main flow	<ol style="list-style-type: none"> 1. User presses the Skin Store button and enters the Skin Store 2. If the user has enough points to select skins, the user can choose the skin the user can afford. (extension point: Purchasing the skin)
Alternative path	-

8) Purchase a skin

Use case name	Purchasing a skin
Actor	User
Description	For the user to get the skin and wear.
Precondition	The user must have clicked the 'Skin Store' button.
Postcondition	The user can wear the skins that they have.
Main flow	<ol style="list-style-type: none"> 1. The system shows all the skins and the skins that the user has. 2. The user can choose and purchase the skin that the user can afford.
Alternative path	<ol style="list-style-type: none"> 2.a The user can go to the main menu without taking anything. (extension point: Main Menu)

9) Control the character

Use case name	Controlling the character
Actor	User
Description	For users to control the character.
Precondition	The game system must have started.
Postcondition	The user will control the character.
Main flow	<ol style="list-style-type: none">1. System starts the game system.2. User presses the “Space” key on the keyboard.3. System changes character positions.
Alternative path	-

10) Check leaderboard

Use case name	Checking leaderboard
Actor	User
Description	For users to check the leaderboard.
Precondition	The user must be in the main menu or end menu.
Postcondition	The user will see the leaderboard.
Main flow	<ol style="list-style-type: none">1. System shows the worldwide leaderboard.2. The user can view the top 5 players' high scores and the user's own score.
Alternative path	-

11) Retry game

Use case name	Retry game
Actor	User
Description	For users to retry the game.
Precondition	The end menu must be viewed in the game.
Postcondition	The essential game systems will be initialized again.
Main flow	<ol style="list-style-type: none"> 1. The user presses the “Retry” button. 2. System starts the game system again. (inclusion point: Play the game)
Alternative path	-

12) Choose an option from main menu

Use case name	Choose an option from main menu
Actor	User
Description	For showing the users the directions that they can choose.
Precondition	The user must be logged in to the game.
Postcondition	The user will be directed to their selection.
Main flow	<ol style="list-style-type: none"> 1. System shows the main menu. 2. The user will decide which option to take 3. The user will select “Play” (extension point: Play the game)
Alternative path	<ol style="list-style-type: none"> 3.a. The user will select “Help” (extension point: Getting help) 3.b. The user will select “Turn sound effects on/off” (extension point: Turning sound effects on/off) 3.c. The user will select “Skin store” (extension point: Entering the skin store) 3.d. The user will select “Worldwide leaderboard” (extension point: Checking leaderboard)

13) Open end menu

Use case name	Open end menu
Actor	User
Description	For showing the user's own scores, high scores and directions that they can choose.
Precondition	The user must die in the game.
Postcondition	The user will be directed to their selection.
Main flow	<ol style="list-style-type: none"> 1. The system shows the end menu. 2. Database Manager will update the user's high score if the user's score is higher than the user's high score. Current score will be the high score. (inclusion point: Update high score) 3. Database Manager will update the leaderboard which includes high scores.(extension point: Updating leaderboard) 4. The user will select "Main Menu"(extension point: Returning to main menu)
Alternative path	<ol style="list-style-type: none"> 4.a. The user will select "Retry"(extension point: Retry game) 4.b. The user will select "Worldwide leaderboard"(extension point: Checking leaderboard)

14) Return to main menu

Use case name	Returning to main menu
Actor	User
Description	For returning users to the main menu.
Precondition	The end menu must be viewed in the game.
Postcondition	The user will be directed to the main menu.
Main flow	<ol style="list-style-type: none"> 1. The user will select "Main Menu". 2. The user will be directed to the main menu.
Alternative path	-

15) Start the game system

Use case name	Start the game system
Actor	Control Manager
Description	For starting the essential game systems.
Precondition	The user must have pressed the play button.
Postcondition	The essential game systems will be initialized.
Main flow	<ol style="list-style-type: none"> 1. The Control Manager system will set the health percentage. (inclusion point: Setting health percentage) 2. The Control Manager system will set virus blocks and special items. (inclusion point: Set virus blocks & special items) 3. The Control Manager system will set score and coin. (inclusion point: Setting score and coin) 4. The Control Manager system will set the background and player.(inclusion point: Set background & player)
Alternative path	-

16) Set score & coin

Use case name	Set score and coin
Actor	Control Manager
Description	For changing the user's score and coin.
Precondition	The user earns coins and achieves the game score.
Postcondition	The score count and coin count will increase.
Main flow	<ol style="list-style-type: none"> 1. Move the y-coordinate of the user according to the user input. 2. Check the moved distance. 3. Check collected coins. 4. If the user has any distance. Set available score and coins.
Alternative path	-

17) Set virus blocks & special items

Use case name	Set virus blocks and special items
Actor	Control Manager
Description	For changing the virus block's location, move up and down and collect special items.
Precondition	The game system must have started.
Postcondition	The user can collect special items and during the game user can see virus blocks and special items
Main flow	<ol style="list-style-type: none"> 1. The system shows related objects(virus blocks or special items) 2. Moving the x or y coordinate of the blocks and items according to random. 3. Check related objects. 4. Affect the character's health.
Alternative path	-

18) Set health percentage

Use case name	Set health percentage
Actor	Control Manager
Description	For changing health percentage depending on the damage or special items.
Precondition	The game system must have started.
Postcondition	The user will see the health percentage.
Main flow	<ol style="list-style-type: none"> 1. Control manager checks the y-coordinate of the user according to the user input. 2. Control manager checks the health bar. 3. If the user collects a special item Control manager increases the health bar according to collecting specials. 4. If the user crashes any virus block. Control manager decreases the health bar according to crashes of any virus block.
Alternative path	-

19) Set background & player

Use case name	Setting background and player
Actor	Control Manager
Description	For changing the position of the player with respect to the user input and changing background in certain stages of the game
Precondition	The game system must have started.
Postcondition	The user and the background will get rendered.
Main flow	<ol style="list-style-type: none"> 1. Control manager changes the y-coordinate of the character according to the user input. (inclusion point: "Control the character") 2. Control manager checks the time. 3. If one of the time checkpoints is reached Control manager changes the background
Alternative path	-

20) Update leaderboard

Use case name	Update leaderboard
Actor	Database Manager
Description	For sorting the users with their latest high score saved in the database
Precondition	The game must have ended.
Postcondition	The leaderboard will contain the latest data and rankings.
Main flow	<ol style="list-style-type: none"> 1. Database Manager gets the user's high score. 2. If the high score has changed Database Manager places the user in the leaderboard with respect to their new high score
Alternative path	-

21) Update high score

Use case name	Update high score
Actor	Database Manager
Description	For keeping the highest score of the user in the database
Precondition	The game must have ended.
Postcondition	Database Manager will have the latest high score saved.
Main flow	<ol style="list-style-type: none">1. Database Manager obtains the calculated score.2. If the calculated score is higher than the high score saved in the database Database Manager saves the calculated score as the new high score
Alternative path	-

3. NON-FUNCTIONAL REQUIREMENTS

Requirement ID:1	Requirement Type:NFR (Performance)	Event/Use Case:#
Description: The CPU usage shall not be more than 20%.		
Rationale: The system should not be slowed down.		
Source: CBD Team		
Fit Criteria: We can obtain the CPU usage from the Windows Task Manager performance tab.		
Customer Satisfaction: 5	Customer Dissatisfaction: 5	
Priority: Essential	Conflicts:None	
Supporting Material: None.		Volere
History: Created March 24, 2021		

Requirement ID:2	Requirement Type:NFR (Performance)	Event/Use Case:#
Description: The RAM usage shall not be more than 1024MB.		
Rationale: The execution of other programs should not be affected.		
Source: CBD Team		
Fit Criteria: We can obtain the RAM usage from the Windows Task Manager performance tab.		
Customer Satisfaction: 3	Customer Dissatisfaction: 3	
Priority:Desirable	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

Requirement ID: 3	Requirement Type:NFR (Usability)	Event/Use Case:#
Description: The game shall have a simple interface and shall have no more than 5 distinct controls.		
Rationale: To be able to make user interaction easier and make the game easy to play.		
Source: CBD Team		
Fit Criteria: We can observe the number of control keys with using the game.		
Customer Satisfaction: 4	Customer Dissatisfaction:4	
Priority:Desirable	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

Requirement ID: 4	Requirement Type:NFR (Capacity)	Event/Use Case:#
Description: The disk usage shall not be more than 1024MB.		
Rationale: The system should use disk storage efficiently.		
Source: CBD Team		
Fit Criteria: Checking the total size of the folder in which the game was installed, for the hard disk space can test this requirement.		
Customer Satisfaction:2	Customer Dissatisfaction: 2	
Priority:Desirable	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

Requirement ID: 5	Requirement Type:NFR (Availability)	Event/Use Case:#
Description: The game shall be played on Windows 10.		
Rationale: The game should be available to more people.		
Source: CBD Team		
Fit Criteria: Installing the game on a Windows 10 platform and running the game.		
Customer Satisfaction:4	Customer Dissatisfaction: 4	
Priority:Essential	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

Requirement ID: 6	Requirement Type:NFR (Response Time)	Event/Use Case:#
Description: The average response time between click/press and reaction shall be less than 0.5 seconds.		
Rationale: The game should be able to be played smoothly.		
Source: CBD Team		
Fit Criteria: It can be measured by calculating: total of testing time / number of key presses.		
Customer Satisfaction:5	Customer Dissatisfaction: 5	
Priority:Essential	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

Requirement ID: 7	Requirement Type:NFR (Security)	Event/Use Case:#
Description: The game shall request no personal information from the user.		
Rationale: The game should be able to be played without the need to provide personal information.		
Source: CBD Team		
Fit Criteria: The only information kept in the database are the username, password and the score.		
Customer Satisfaction:2	Customer Dissatisfaction: 1	
Priority: Low	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

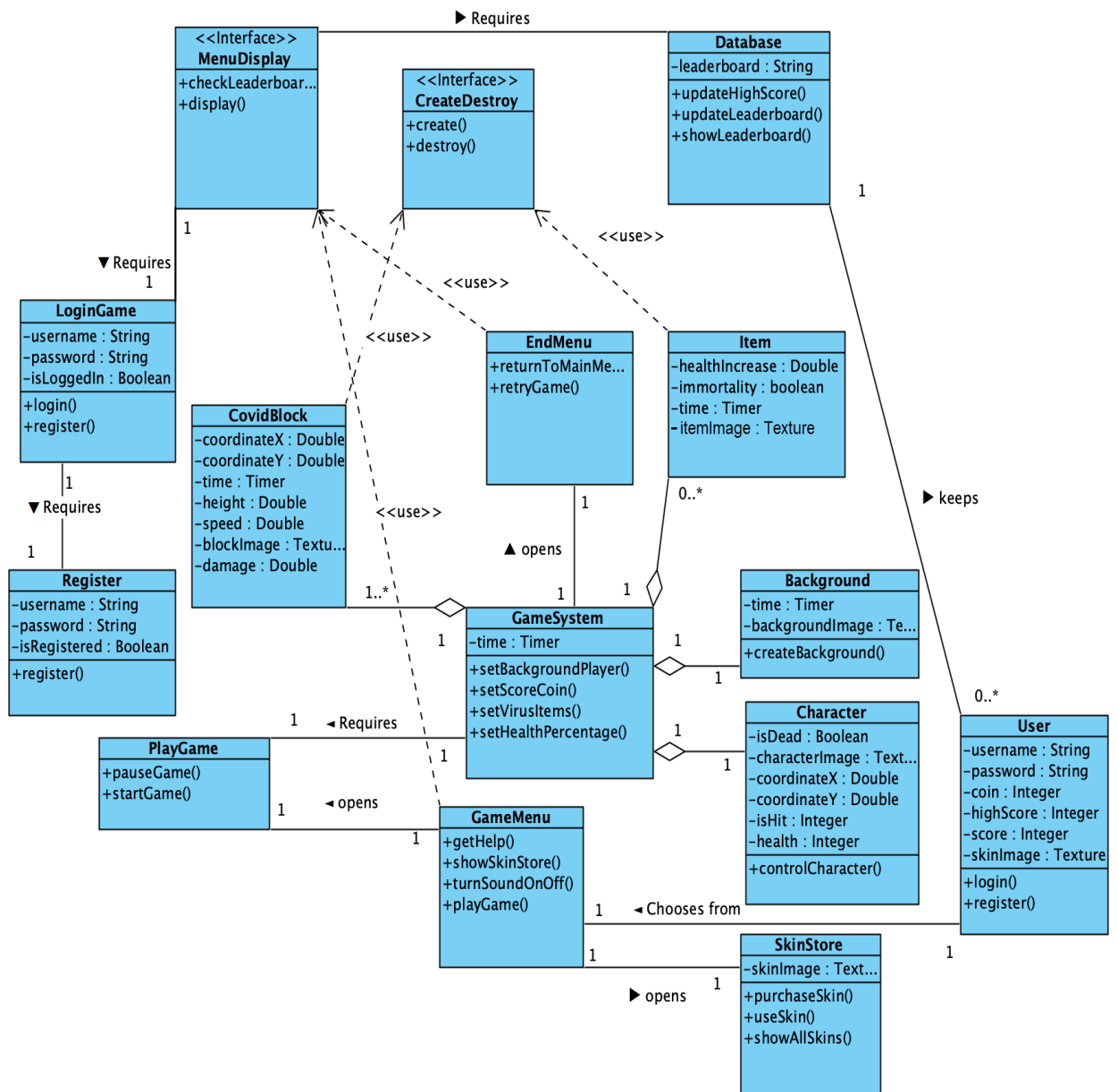
Requirement ID:8	Requirement Type:NFR (Performance)	Event/Use Case:#
Description: The game shall take no more than 10 seconds to launch.		
Rationale: The game should get opened quickly in order to not lose the user interest.		
Source: CBD Team		
Fit Criteria: We can observe how many seconds have passed while running the game using a timer.		
Customer Satisfaction:4	Customer Dissatisfaction:4	
Priority:Desirable	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

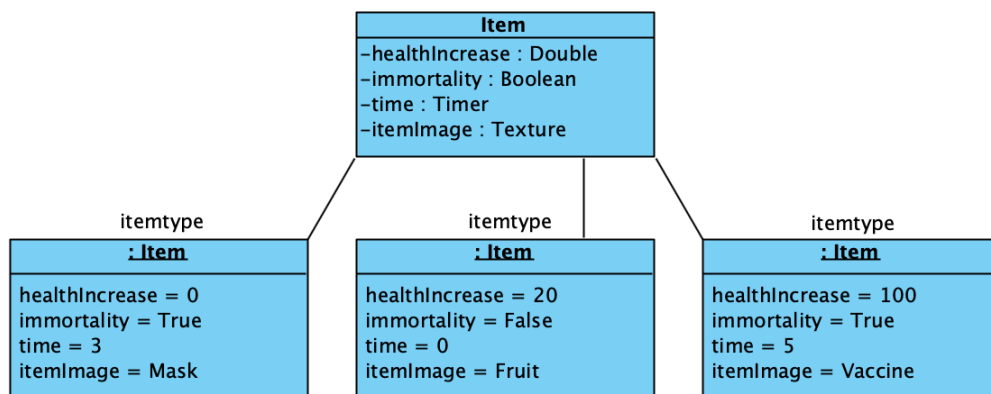
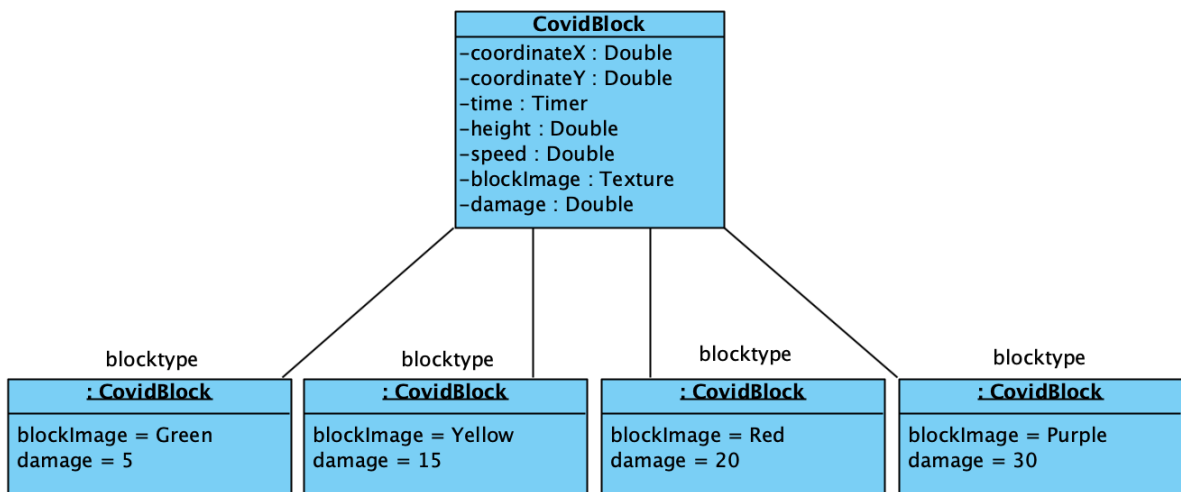
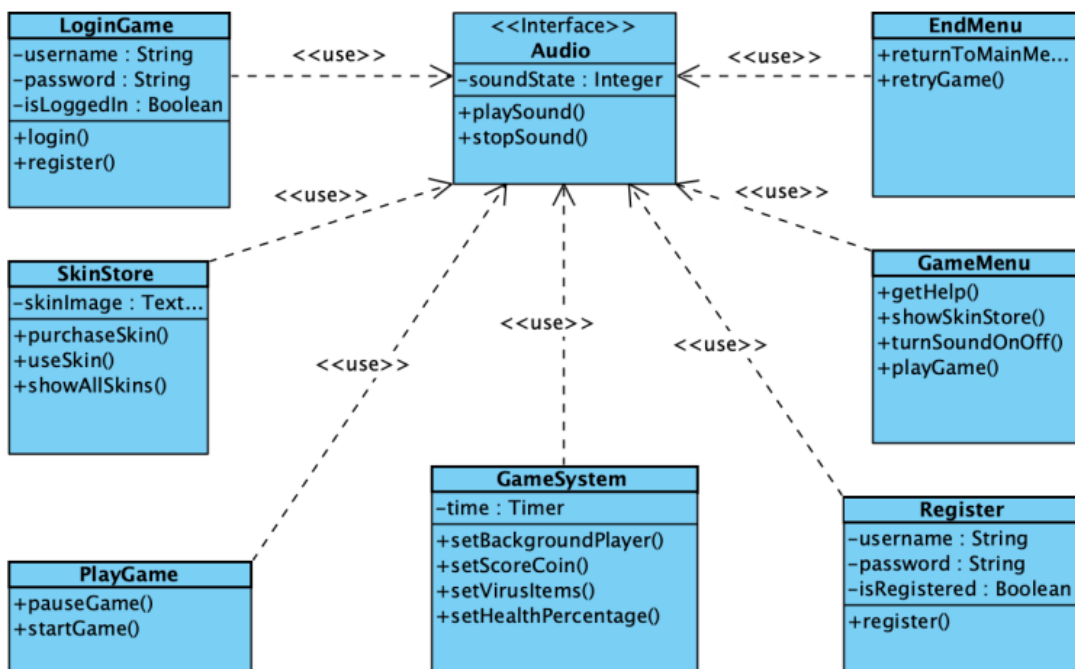
Requirement ID:9	Requirement Type:NFR (Reusability)	Event/Use Case:#
Description: The game shall have an object-oriented architecture and be portable so that it can be played on at least 2 platforms such as mobile and other desktop operating systems if needed.		
Rationale: The game’s architecture should preserve the portability of software to different platforms and programming languages so that it can be integrated into a variety of game engines and platforms. Also this can help in case of the game getting more popular.		
Source: CBD Team		
Fit Criteria: It can be tested with required file extensions on the other platforms thanks to the Unity environment.		
Customer Satisfaction:4	Customer Dissatisfaction:3	
Priority:Desirable	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

Requirement ID:10	Requirement Type:NFR (Maintainability)	Event/Use Case:#
Description: The game structure shall be easily modifiable to add new features such as new skins and game modes.		
Rationale: The game must be developable to appeal to more people.		
Source: CBD Team		
Fit Criteria: We can check whether the game is still working as expected or not after the new features are added.		
Customer Satisfaction:4	Customer Dissatisfaction:3	
Priority:Desirable	Conflicts: None	
Supporting Material: None		Volere
History: Created March 24, 2021		

4. SYSTEM MODELS

4.1. Object and Class Model







4.2. User Interface

CovidBird


Covid Bird
LOGIN






CovidBird


Covid Bird
LOGIN

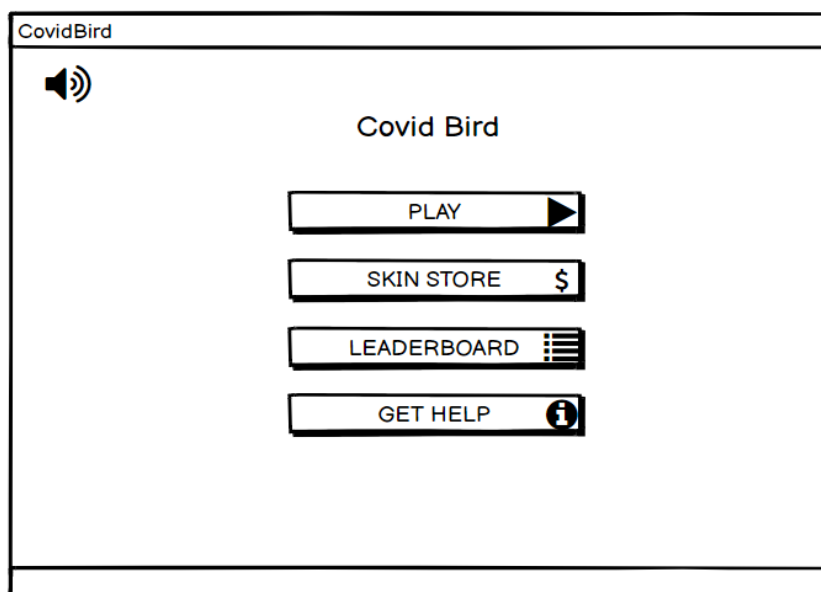
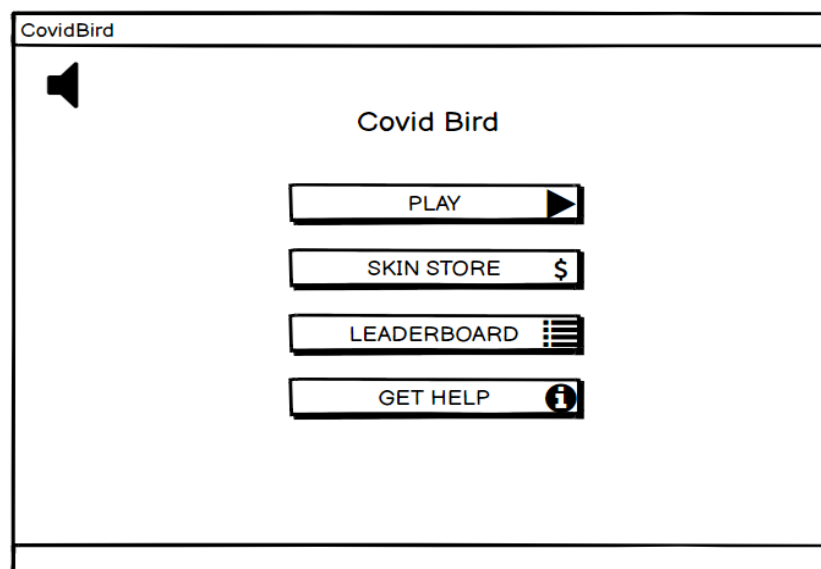
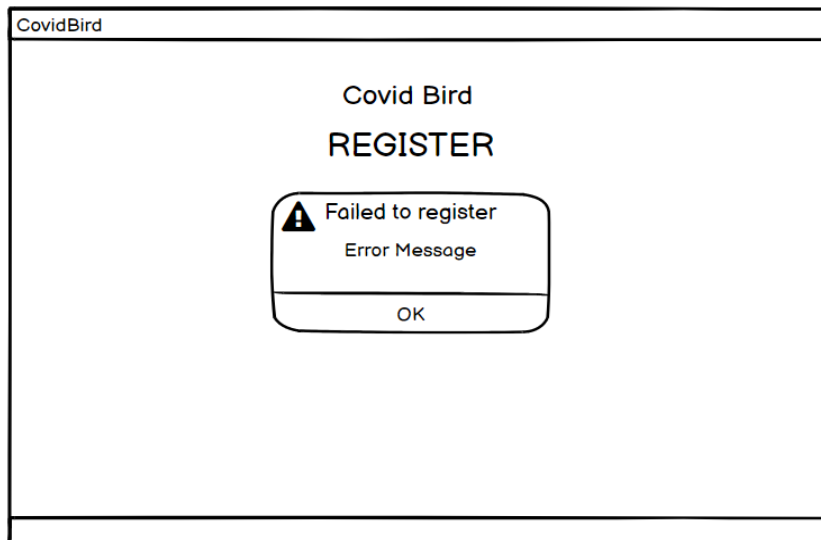
 Failed to login
Error Message

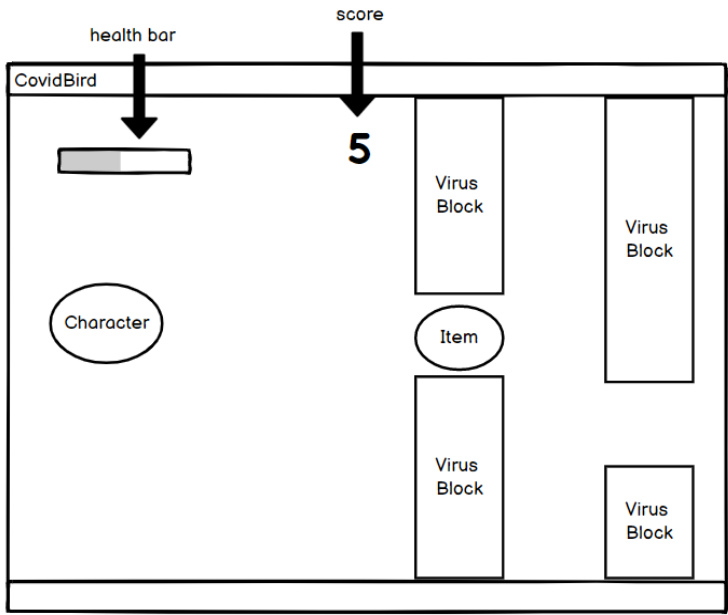
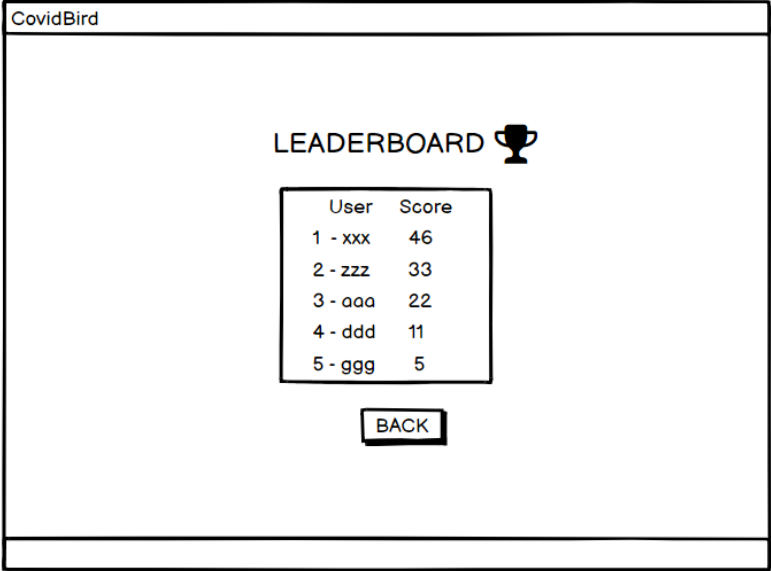
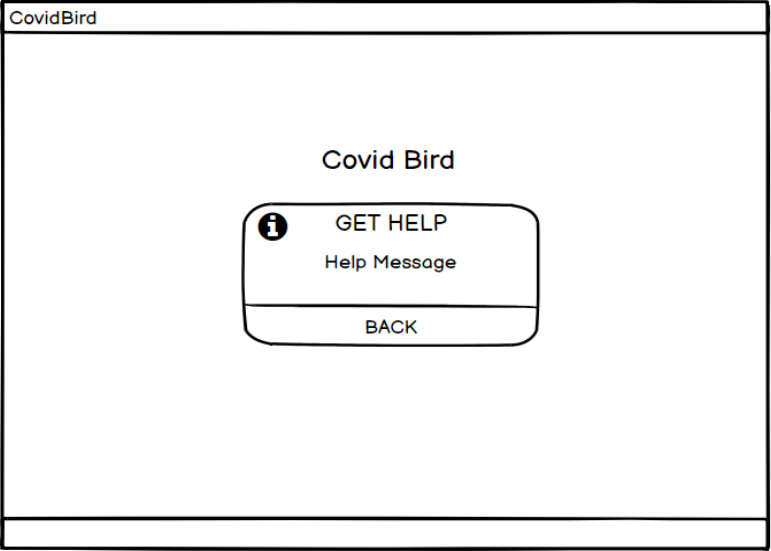
CovidBird

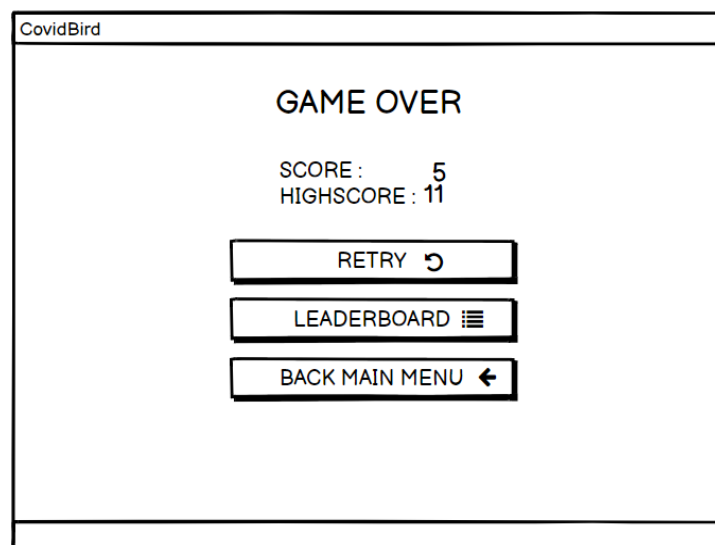
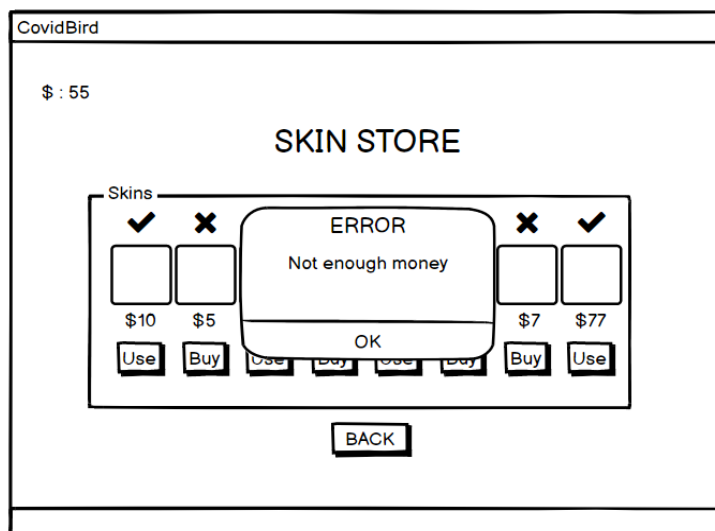
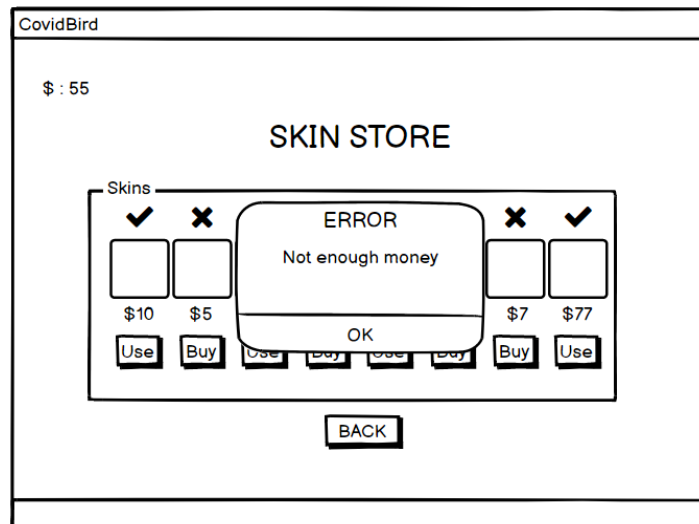
Covid Bird
REGISTER











5. DEFINITIONS, ACRONYMS AND ABBREVIATIONS

NFR - Non-functional Requirements

CPU - Central Processing Unit

CBD - Covid Bird Developers

GUI - Graphical User Interface

RAM - Random Access Memory

OS - Operating System

IOS - iPhone operating system

MB - Megabyte

6. GLOSSARY & REFERENCES

6.1. Glossary

Non-functional Requirements : In systems engineering and requirements engineering, a non-functional requirement (NFR) is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors.

Database : A database is an organized collection of data, generally stored and accessed electronically from a computer system.

Unity : Unity is a cross-platform game engine developed by Unity Technologies, first announced and released in June 2005 at Apple Inc.'s Worldwide Developers Conference as a Mac OS X-exclusive game engine.

Android : Android is a mobile operating system based on a modified version of the Linux kernel and other open source software, designed primarily for touchscreen mobile devices such as smartphones and tablets.

iOS : iOS (formerly iPhone OS) is a mobile operating system created and developed by Apple Inc. exclusively for its hardware.

Windows Task Manager : Task Manager, previously known as Windows Task Manager, is a task manager, system monitor, and startup manager included with Microsoft Windows systems.

Response Time : The time lag between an electronic input and the output signal which depends upon the value of passive components used.

6.2. References

- [1]<https://www.visual-paradigm.com/guide/use-case/what-is-use-case-specification/>
- [2]<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>
- [3]https://www.visual-paradigm.com/support/documents/vpuserguide/94/2584/7191_drawingobject.html
- [4]<https://balsamiq.cloud/>
- [5]Timothy C. Lethbridge and Robert Laganière. (2004). *Object-Oriented Software Engineering: Practical Software Development using UML and Java*. (2nd edition). McGraw Hill Higher Education.
- [6]<https://www.site.uottawa.ca/~tcl/seg2105/coursenotes/ClassDiagramChecklist.html>
- [7]<https://www.scribd.com/document/350193974/Use-Case-Specifications>
- [8]<https://en.wikipedia.org>