Software Requirements Specification

for

CowClicker

Version 1.0 approved

Prepared by Tolga Olcay (T00666715) and Sadman Mugdho (T00674177)

Thompson Rivers University Comp 2920

10/25/2021

Table of Contents

1. Introduction

1.1 Purpose

The Purpose of this SRS document is to identify and highlight the requirements for a simulation idle game. CowClicker is a simulation idle game that runs on an offline webpage. The purpose of the game is to exponentially increase the rate of production of in-game currency and milk units by researching and purchasing upgrades, similar to other popular idle games such as Cookie Clicker[1] and Antimatter. The goal of CowClicker is to create an engaging simulation game that will reach a vast audience of gamers.

1.2 Document Conventions

This SRS document will contain the following conventions.

- Abbreviated words will be written in **bold** and their definition can be found in the Glossary
- Functional requirements will be written as FR###, the # representing the digit used to identify each requirement. The leftmost digit represents the system feature that the functional requirement is related to.
- Non-functional requirements will be written as NFR###, the # representing the digit used to identify each requirement.
- Words that are in *italics* will have a detailed definition in the glossary

The purpose of these conventions is to provide a clear and consistent structure for the reader of this document.

1.3 Intended Audience and Reading Suggestion

This SRS document is intended to provide insight of the features and requirements of CowClicker to developers, and project managers, who intend to develop and maintain this application. The following document shall be organized to highlight the purpose and scope of **CC**, an overall description of the systems of **CC**, outline the user interfaces of **CC**, and define the system features along with their function and non-functional requirements. For maximum clarity it is suggested to read the document in order, and to visit the Glossary to learn the abbreviated terms and terms that are specialized to our scope .

1.4 Product Scope

CowClicker is a single-player simulation idle game that runs on a static web browser. Any user with a device that can open this static web application will be capable of playing CowClicker.

1.5 References

1. *Orteil, “Cookie Clicker,” Cookie clicker. [Online]. Available: https://ozh.github.io/cookieclicker/. [Accessed: 02-Nov-2021].*

2. *W3schools, “Window localstorage Property,” Window localstorage property. [Online]. Available: https://www.w3schools.com/jsref/prop_win_localstorage.asp. [Accessed: 02-Nov-2021].*

3. *U. I. tricks Tricks, “Disadvantages or limitations of JavaScript,” UI Tricks, 09-Dec-2015. [Online]. Available: https://uitrickz.wordpress.com/2015/12/10/disadvantages-or-limitations-of-javascript/. [Accessed: 02-Nov-2021].*

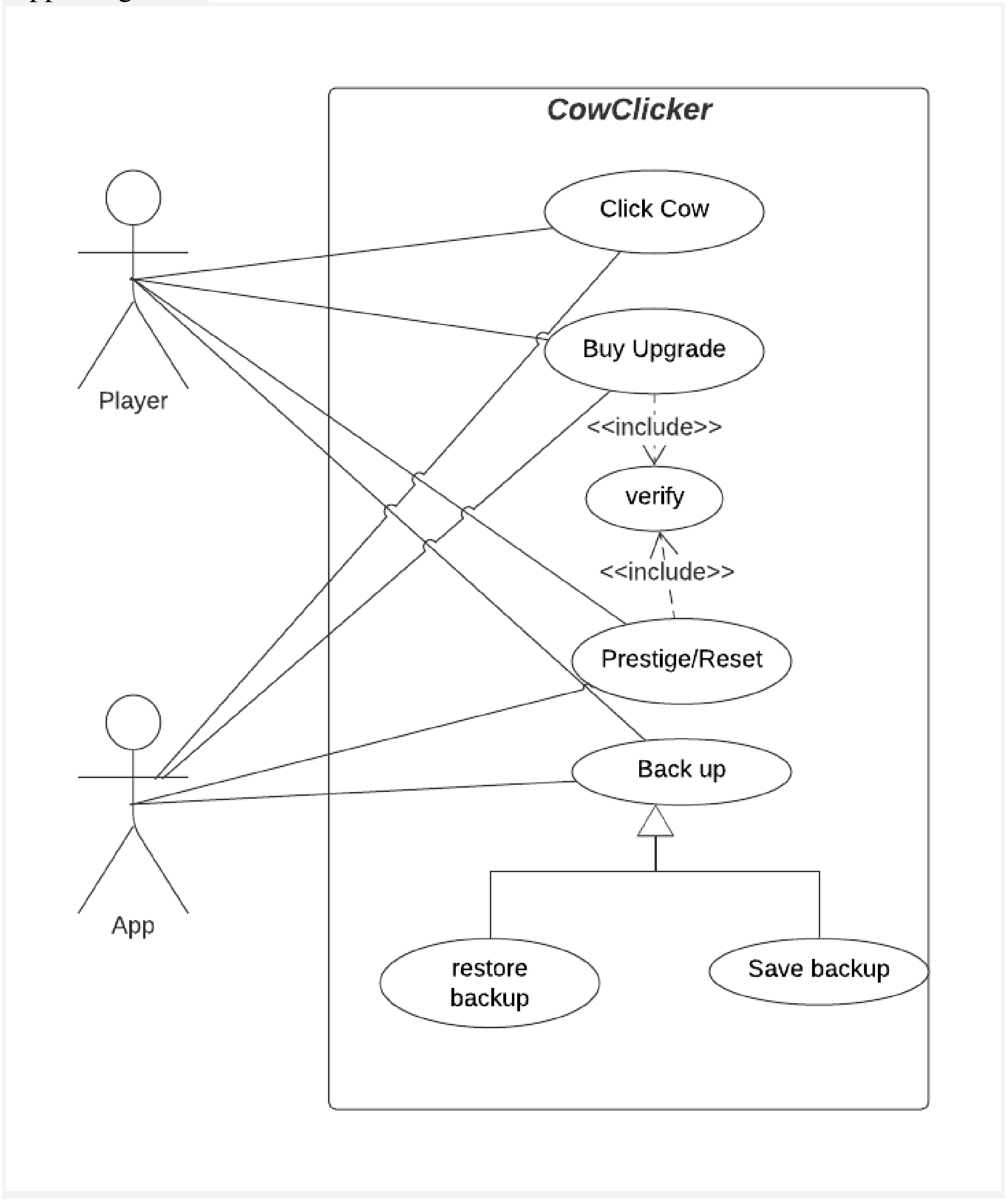
2. Overall Description

2.1 Product Perspective

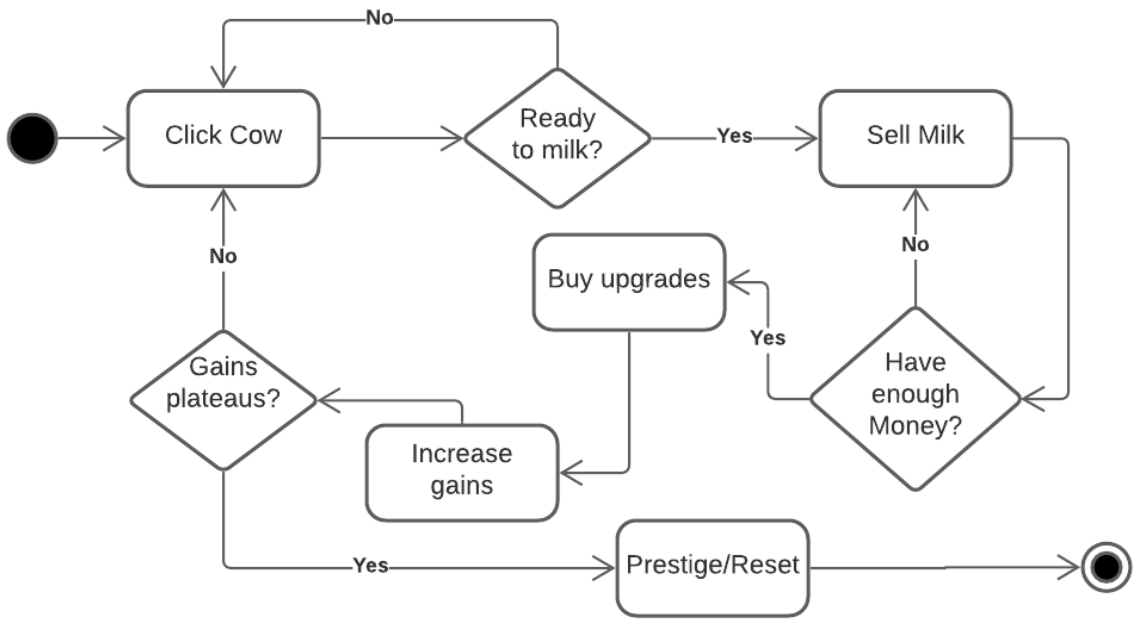
CowClicker is a new self contained software, and is not a component of a larger software.

2.1.1 Use Case Diagram

In this Use Case Diagram, the player is the primary actor while the app (CowClicker) is the supporting actor.



2.1.2 Main Activity Diagram



2.2 CowClicker Functions

The main functionality of CowClicker is to provide a simulation of earning milk units and in-game currency. The user can purchase upgrades and increase the rate of earned milk and currency. The simulation will simulate the amount of milk and currency earned while the application was closed, and will have the functionality to save progress onto a text file, and to load progress from a text file. The user will earn trophies for reaching certain milestones, and provide the user with an endgame/reset, allowing the user to replay the simulation.

2.3 User Classes, Use Cases and Characteristics

There are many anticipated classes that will be used to create CowClicker, such as the Cow Class, in which many subclasses (such as the LeakyCow, the SuperCow, the GoldenCow etc) will have inherited functionality. There will also be a MilkUnits Class, a Currency Class, a Trophies Class, and an Upgrades Class. These classes will have relationships with each other in order to increase the **MPR** and the **CPR**

2.4 Operating Environment

The app will be designed with the chromium browser in mind. Applicable on both desktop and mobile. It is recommended to get the latest chromium version and have javascript enabled to run the app.

2.5 Design and Implementation Constraints

CowClicker is a Javascript based static web browser application. Thus it's design limitations are dependent on the tools used to create the application. Some common Javascript limitations are:

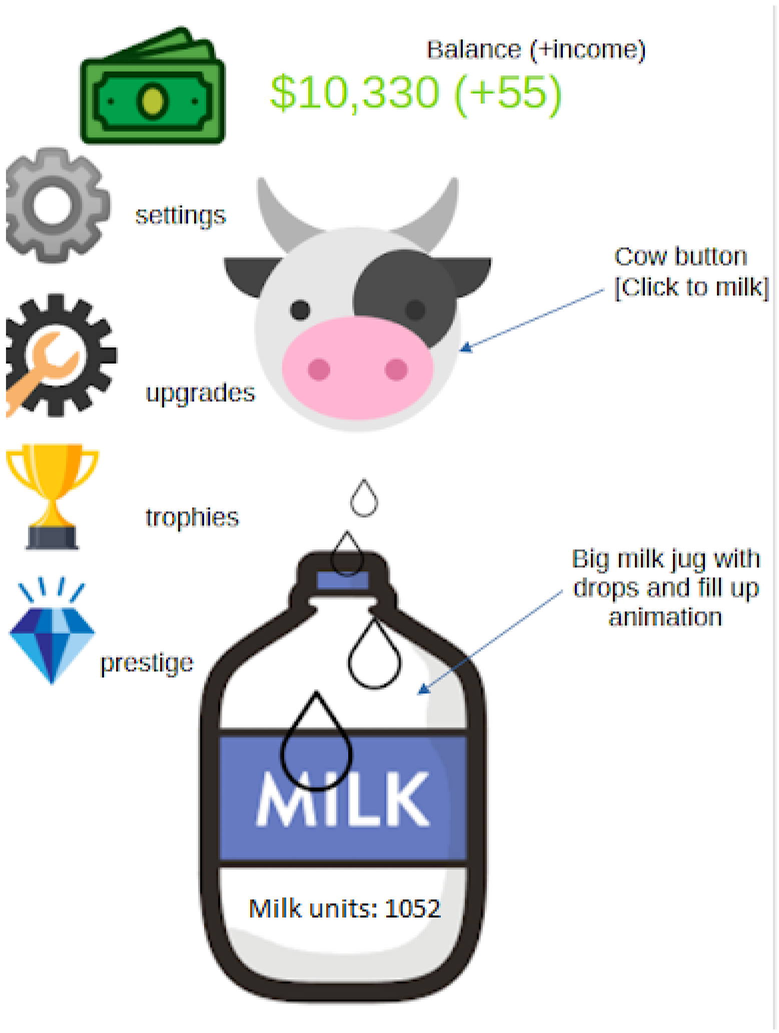
- Browser compatibility- JavaScript mostly depends on the web browser. As different browsers interpret JavaScript differently using their own JavaScript Engines. You have to write cross browser codes to support all browsers. Sometimes your application may behave differently in different browser.[3]
- JavaScript can be disabled – All the browsers provides option to disable JavaScript to run by the user due to specific security reasons. If your website is highly dependent on JavaScript codes it is sometimes difficult to support to run your web app.[3]

These limitations of the tools used to develop CC have been taken into account when designing the application.

2.6 Assumptions and Dependencies

CowClicker is not a component of a larger software system therefore it is free from many software dependencies. The main software dependency of CowClicker is that the user is running CowClicker on a compatible device and browser. It is assumed that the user is using a browser and device that is capable of running javascript. It is assumed that the user is capable of accessing CowClicker. It is assumed that the user can read english.

2.7 Interface Design



2.8 Software Architecture to follow

Model-view-controller will be the software architecture for CowClicker.Since this is a game and visual in nature, the Model, which contains the core code needs to be abstracted into the Visual. Since the game requires user input, a controller, to progress through the game, by purchasing upgrades and clicking the cow, model-view-controller will be necessary.

3. External Requirements

3.1 Hardware Interfaces Requirements

On the desktop, no keyboard action is required, as everything will be click based.
On mobile, only single tap actions will be allowed on touch screens.

3.2 Software Interfaces Requirements

Browser localstorage will be used to save, restore and back up game progress data. Otherwise, the app is made in vanilla javascript with no other dependencies or databases. It is a self contained simulation game.

4. System Features

The following system features outline the functional requirements of CowClicker. These system features shall be organized by functional hierarchy, logically from beginning to end. Each system feature will be assigned a number that will correspond to their Function requirement.

4.1 Simulation Summary (1)

4.1.1 Description and Priority

Upon loading **CC**, the user will be greeted with the simulation summary, the main webpage where the simulation of milk units and currency takes place. Here the user can view the current totals of their **MP**, **Credits** and trophies. While the simulation summary webpage is active, the user will receive a constant stream of **MP** and **Credits** at a rate depending on their *milk production rate* and their *currency production rate*, respectively. The Simulation Summary will keep track of relevant data, such as the current time, the **MPR**, the **CPR**, and the current total of the users **MP** and **Credits**. The Simulation summary shall have a button with a cow on it, clicking the cow shall add 1 milk unit. As this system feature is the main content of CowClicker, it has High priority.

4.1.2 Stimulus/Response Sequences

When a *returning user* loads **CC**, the simulation summary will simulate the amount of earned **MP** and **Credits** while CowClicker was closed.

4.1.3 Functional Requirements

FR101: The simulation summary shall be loaded upon opening CowClicker
FR102: The user shall receive a constant stream of milk units depending on their MPR, if the MPR is 0, they will earn no milk units.
FR103: The user shall receive a constant stream of in-game currency depending on their CPR, if the CPR is 0, they will earn no currency.
FR104: The number of total **MP** shall be a variable in the **CPR**
FR105: The simulation summary shall display the total **MP** earned
FR106: The simulation summary shall display the total **Credits** earned
FR107: The simulation summary shall display all earned trophies.
FR108: The simulation summary must allow the user to visit the settings menu and research and shop menu
FR109: The simulation summary shall save the current system time, **MPR**, **CPR**, and the total **MP** and **Credits** to JS local storage on every update cycle.
FR110: For a returning user who has reopened **CC** on the same browser and device, the simulation summary shall use the *game data* from local storage, and the current system time to simulate the amount of earned **MP** and **Credits**, while **CC** was closed.
FR111: The Simulation summary shall have a clickable button, with each click earning one milk unit.

4.2 Settings Menu (2)

4.2.1 Description and Priority

The settings menu is a tab that displays the following options:

- backup to file: save the current *game data* to a file
- restore from file: restore a game using a save file
- wipe save: delete current *game data*, and start from beginning

This feature has medium priority

4.2.2 Stimulus/Response Sequences

Clicking the “backup to file” option will notify the user where the gamedata text file, a file containing *game data*, has been saved.

Clicking the “restore from file” option will prompt the user to select a valid text file containing *game data*. If a text file with invalid *game data* is selected, the user will be notified accordingly and the simulation will not be changed.

Clicking the “wipe save” option will delete all *game data* from JS local storage, causing the simulation to restart from the beginning.

4.2.3 Functional Requirements

FR201: Settings Menu must be able to open and close
FR202: Settings Menu must display the options “backup to file”, “restore from file” , and “wipe save”
FR203: Clicking “backup to file must notify the user where the gamedata text file has been saved.
FR204: Clicking “restore from file” will prompt the user to select a text file that contains valid *game data*.
FR205:If the user selects an invalid file, the user shall be notified accordingly and the simulation will not be affected.

4.3 Backup and Restore (3)

4.3.1 Description and Priority

By backing up to file, a text file named gamedata will be created in a TBD directory. This text file will contain the values of the time, **MP, Credits, MPR, and CPR** that is saved in the browser's local storage. This is the process of backing up.

By restoring from a file, CowClicker will read *game data* from a text file and alter the simulation to fit the specifications based on the *game data* from the text file. This process allows the user to restore a previous game, on to any device or browser capable of running **CC**. The priority for backing up and restoring is high.

4.3.2 Stimulus/Response Sequences

When backing up to file, a text file named gamedata will be created in a TBD directory.
When restoring from a file, CowClicker will read *game data* from a text file and alter the simulation to fit the specifications based on the *game data* from the text file.

4.3.3 Functional Requirements

FR301: When the user backs up to a file, a text file named gamedata, which contains the current *game data* saved in local storage, shall be created in a TBD directory.
FR302: When the user restores from a file, **CC** shall read the *game data* from the text file, and **CC** shall run the simulation based on the attributes of the text file.

4.4 Simulating (4)

4.4.1 Description and Priority

When a *returning user* reopens CowClicker from the same device and browser, **CC** will simulate the amount of earned **MP** and **Credits** while the web application was closed. This feature has high priority.

4.4.2 Stimulus/Response Sequences

The user must have reopened CowClicker on the same device and browser. **CC** will calculate and simulate the amount of earned milk units and currency while **CC** was closed, based on the **MPR**, **CPR**, current time, and the time saved in local storage.

4.4.3 Functional Requirements

FR401: When a user reopens **CC** on the same browser and device, the *game data* saved in *local storage* shall be used to calculate and simulate the amount of earned **MP** and **Credits**.

4.5 Research & Shop (5)

4.5.1 Description and Priority

The Research and Shop menu is a tab that displays available upgrades that the user could purchase with in-game currency. These upgrades can increase the **CPR** and **MPR**. Upgrades can be bought multiple times allowing for their bonuses to stack. Each time a specific upgrade is purchased, their price will be increased. The priority for this feature is medium.

4.5.2 Stimulus/Response Sequences

The following are shop choices the user can purchase, the price for each is TBD:

- Regular Cow - adds 1**MP** per 5 seconds **MPR**
- Leaky Cow - adds 10 **MP** per 5 seconds **MPR**
- Super Cow - adds 15 **MP** per 1 second **MPR**
- Golden Cow - adds 100 **MP** per 5 seconds **MPR**
- Rainbow Cow - adds 100 **MP** per 1 second **MPR**
- Invisible Cow - adds 1000 **MP** per 1 second **MPR**
- Improve milk amount - increase the amount of **MP** earned for all cows by 2%, thus increasing the **MPR**
- Efficient Regular Cows - increase the amount of **MP** earned from regular cows by 20%, thus increasing the **MPR**
-
- Efficient Leaky Cows - increase the amount of **MP** earned from regular cows by 20%, thus increasing the **MPR**
-

- Efficient Super Cows - increase the amount of **MP** earned from regular cows by 20%, thus increasing the **MPR**
-
- Efficient Golden Cows - increase the amount of **MP** earned from regular cows by 20%, thus increasing the **MPR**
-
- Efficient Rainbow Cows - increase the amount of **MP** earned from regular cows by 20%, thus increasing the **MPR**
-
- Efficient Invisible Cows - increase the amount of **MP** earned from regular cows by 20%, thus increasing the **MPR**
- Speedy Cows - the time it takes to produce milk for all cows is reduced by 1%, thus increasing the **MPR**
-
- Improve Milk Taste - The total amount of **MP** increases the **CPR** by 1%
- Improve milk Quality - the total amount of **MP** increases the **CPR** by 5%
- Secret Formula - Increase **MPR** by 200%
- Cloning Machine - the number of owned cows will be doubled

4.5.3 Functional Requirements

FR501: The Research and shop menu must be able to open and close
FR502: The user must be able to spend in-game currency to purchase upgrades
FR503: Upgrades can be purchased multiple times, allowing for their bonuses to stack
FR504: When an upgrade is purchased, it's price will increase by 20% for the next time it is purchased.
FR505:The shop menu shall display the available upgrades to purchase
FR506: The shop menu shall display the number of times an upgrade has been purchased
FR507: Purchasing a cow upgrade will add its milk production rate to the Total **MPR**
FR508: The regular cow shall have a default **MPR** of 1**MP** per 5 second(s)
FR509: The leaky cow shall have a default **MPR** of 10**MP** per 5 second(s)
FR510: The super cow shall have a default **MPR** of 15**MP** per 1 second(s)
FR512: The golden cow shall have a default **MPR** of 100**MP** per 5 second(s)
FR513: The rainbow cow shall have a default **MPR** of 100**MP** per 1 second(s)
FR514: The regular cow shall have a default **MPR** of 1000**MP** per 1 second(s)
FR515: Efficiency Upgrades shall upgrade a specific cow type's **MPR** by 20% when purchased
FR516: Speedy cow upgrade shall have the time it takes to produce milk for all cows is reduced by 1%, thus increasing the **MPR** for all cows

FR517: The total amount of **MP** increases the **CPR** by 1% when purchasing Improve Milk Taste

FR518: The total amount of **MP** increases the **CPR** by 5% when purchasing Improve quality
FR519: Purchasing secret formula increases the total **MPR** by 200%
FR520: Purchasing cloning machine doubles the amount of owned cows

4.6 Trophies (6)

4.6.1 Description and Priority

Trophies are earned when the user achieves a milestone. Each trophy will have a brief description that describes how it was earned. The user can earn multiple trophies and they will be displayed in the simulation summary. The priority for this system feature is low.

4.6.2 Stimulus/Response Sequences

When a trophy is unlocked the user will be notified with a message.The following trophies will be unlocked after reaching a milestone:

- Best Milk in the Business: Purchase “improve milk taste ” and “improve milk quality” upgrades
- Spend Money to Make Money: Purchase all upgrades
- Mastermind: Acquire 1 million dollars
- Billionaire: Acquire 1 Billion dollars
- Milk Man:Acquire 1 thousand milk units
- Mega Milk:Acquire 1 million milk bottles
- Cows are my friends: Acquire over 1000 cows

4.6.3 Functional Requirements

FR601: The user will earn a trophy when reaching a milestone
FR602: The user will receive a notification after earning a trophy
FR603: earned trophies can be viewed in the simulation summary with a description of how they were earned

4.7 EndGame (7)

4.7.1 Description and Priority

After reaching a TBD condition, the user will be prompted if they would like to start the endgame. The endgame will delete all gamedata, trophies, and restart from the beginning, however, the default total **MPR** will be increased by 100%. This allows the user to potentially reach an even amount of **MP.** The priority for this system feature is low.

4.7.2 Stimulus/Response Sequences

The user will be prompted to start the endgame. If they decline they can continue their current simulation. If they accept, they will restart their simulation with an increased default total **MPR.**

4.7.3 Functional Requirements

FR501: CowClicker must prompt the user if they would like to begin the endgame after reaching a TBD condition
FR502: If the user accepts they will restart the simulation with a 100% increase to their default total **MPR**
FR503: If they decline they will continue their current simulation

5. Nonfunctional Requirements

5.1 Performance Requirements

Here are some non-functional requirements that relate to performance.

NFR101: When a user purchases an upgrade that modifies the current **MPR** and/or **CPR,** the simulation shall simulate based on those changes immediately (in less than one second).
NFR102: When a returning user re opens **CC,** the calculation to simulate the amount of earned milk units and currency while the game was closed, shall be immediate (less that two seconds)
NFR103: Notifications, such as trophies messages, or messages notifying the user where a gamedata text file has been saved, shall notify the user within at least 5 seconds.

5.2 Safety Requirements

Here are some non-functional requirements that relate to safety.

NFR201: The software must not cause the device to overheat

5.3 Security Requirements

Here are some non-functional requirements that relate to security.

NFR301: The software must not be able to access the user's personal data without permission
NFR302: The user must not share any data relating to the user

5.4 Software Quality Attributes

NFR401: The software must be adaptable, allowing for developers to change values of prices and modifiers
NFR402: The software must be testable, allowing for developers to test beta versions of CowClicker.
NFR403: The software must be usable on multiple different types of browsers, such as chrome and Firefox, and devices such as Mac, or Windows.
NFR404: CowClicker must be robust, and have no bugs that cause the software to crash.

6. Training Plan

The team who is responsible for developing Cow Clicker will have 2 (or more) members, who are students at TRU studying computer science. Each member will have a fair and equal portion of workload for the project, and each member will be credited accordingly.

7. Release Plan

Once all high and medium priority system features are implemented, a beta version of CowClicker will be published to test if they meet the standards of the non-functional requirements. Then, after 6months of development, a final product including the low priority system features will be published.

8. Glossary

game data: relevant data such as time, current total of milk units, current total of currency, current CPR and current MPR, and purchased upgrades and earned trophies.
update cycle: the number of milliseconds defined by the function requestanimationframe() which is around 1/60s. Calculating the time passed, we add progress to all updatable data as a function of time passed.
returning user: a user who has already ran CowClicker on their device
static web app : a web application that can be delivered directly to an end user's browser without any server-side alteration of the HTML, CSS, or JavaScript
local storage: The localStorage object stores data with no expiration date. The data will not be deleted when the browser is closed, and will be available the next day, week, or year.[2]
MPR*/Milk production rate:* a mathematical expression used to calculate the rate of earned milk units
CPR*/Currency production rate:* a mathematical expression used to calculate the rate of earned currency
CC *:* CowClicker
MP *:* Milk units - a point system that acts as a variable in the CPR
Credits *:* in-game currency