Software Requirements Specification

for

Sierra Store

Version 1.0 approved

Prepared by Sierra Software Team

Sierra Software

May 10, 2019

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Pre-Release	4/4/19	First Commit	0.0
Web Server Functionality	4/10/19	Web Server Framework Created	0.1
User Functionality	4/12/19	User Functionality Created	0.2
Web Functionality	4/16/19	HTML Functionality	0.3
Sierra Store	5/9/2019	First Release	1.0

1. Introduction

1.1 Purpose

The product we have is Sierra Store, version 1.0.

1.1.1 Vision Statement

The aim of this project is to create a flexible and easy-to-use repository where users can quickly find video games on different platforms and in different genres. Many people, young and old, will be able to use this software to easily find new games.

1.1.2 Scope

The scope of this project includes the development of a web application framework, searching and sorting algorithms, and maintenance of a database containing information about games. The searching algorithm works with the database, while the sorting and live searching work with the table of information provided to the user.

1.2 Document Conventions

This document follows MLA Format. Bold text has been used to emphasize the main sections and their sub-headings. Italicized text is used to label and recognize diagrams provided.

1.3 Intended Audience and Reading Suggestions

This document is intended to be read by the Sierra Software Development team, the project managers, the professors, and the Teaching Assistants. Our stakeholders who clone or download the project from the Github Repository may review this document to learn more about the project and understand the requirements. The SRS has been organized approximately in order of increasing specificity. The developers and project managers need to be very familiar with this SRS.

Others involved need to review the document as such:

Overall Description - Future marketing staff would have to become accustomed to the various features of the Sierra Software Store in order to advertise the project effectively.

System Features - Testers need an understanding of the system features to develop meaningful test cases and give useful feedback to the developers

External Interface Requirements - The hardware developers need to know the requirements of the device they need to build. The marketing staff also needs to understand the external interface requirements to sell the product by describing the user-friendly features of Sierra Software Store.

Nonfunctional and Functional Requirements - The hardware Developers need to be familiar with the specific requirements to effectively develop the product.

1.4 Product Scope

Sierra Software Store has features that enable users to search, sort, and filter through a database of games from various platforms. Users are also able to sign in and submit a game application to have a game added to the database.

1.5 References

A use case diagram has been attached to accompany sections 2.1: Product Perspective and 2.2: Product Features. Product mock-ups and diagrams have been attached for reference.

2. Overall Description

2.1 Product Perspective

The software product being developed is for a Linux-hosted web server which holds a database of Games and enables sorting and filtering through video games from different platforms. The product works with front-end HTML and Javascript, and back-end gems utilized by Ruby on Rails to perform the functionality. The product uses hardware common to any desktop computer running a Linux variant. The product is accessible by IP address and port number, which is determined when the server is launched. Currently, there are many game databases existing, but there are none that allow for multi-platform searching. Our product allows users to find games for their Xbox while on their Windows 10 computer. Our product combines the wide functionality of

search engines like Google and specific Game Stores, like Steam. Refer to the attached use case diagram for more information.

2.2 Product Functions

The Sierra Store contains the following key features:

1. The software product can carry on the user registration, the user authentication information.
2. Show all available games in the store.
3. Allowed user to sign up for a new account.
4. Accept user search for games by keywords, sort games by name, description, price.
5. Provide game information, such as game I sample, game name, game description, game price
6. Allows users to purchase games
7. Allows administrators to add games and delete games.
8. Allows administrators to review game requests and approve games.
9. Allows Administrator reject game with details about the operation.
10. Record administrator action records.
11. Allows administrators to edit users, such as rename, edit email, and change user permissions.

2.3 User Classes and Characteristics

2.3.1 Customer:

Users can browse for games that exist in the store.Users can search games by entering keywords in the search box.Users can also browse the detailed description of each game, such as game introduction, game platform.Users can buy games.

2.3.2 DBA:

The DBA is expected to have a field appropriate college degree and relevant coursework completed in Databases. He/She will approve and reject user applications and manage user privileges. The DBA can also remove games from the database and change any information about games.

2.3.3 Data Entry Level Personnel:

The Data Entry Level Personnel must have a High School diploma or equivalent certification. They do not have privileges to directly edit games or users in the database, but will interact with the Sierra Software interface by adding new games in.

2.4 Operating Environment

The software will operate with the following software components and applications:

The software will be running within a Linux distribution, running Ruby version 2 or later, and Rails version 5.0 or later, using the specified Gems in the Gemfile. The hardware

running this project can be achieved by installing Linux on any existing computer. The build script is written for Linux operating systems and will not run on Windows.

2.5 Design and Implementation Constraints

1. Synchronization: uses Bash, can only execute in a Unix-based Operating System.
2. Memory: needs 5GB available space to download source code and provide sufficient temporary storage for log files and temporary internet files.
3. Internet: needs sufficient connection to the internet to allow clients to reach the web server.

2.6 User Documentation

For user documentation and information, consult section 4: External Interface Requirements.

2.7 Assumptions and Dependencies

It is assumed that the hardware on the host server meets the minimum standards to run a Unix-based operating system and effectively run a web server. Each Linux Distribution ships with different software packages. If you encounter errors during initialization, please visit https://rubygems.org to resolve the dependency.

3. External Interface Requirements

3.1 User Interfaces

The Sierra Software Store user interface has been designed for those people who like video games, giving them an extensible framework to search for new, lesser-known games.

Our home page has a live search bar with the that allows the user to immediately begin searching, sorting, and filtering all the available games in the database. In addition, if user wants to search the game he want and have persistent results that he can then sort and filter, they can use the search bar at top of the home page to first generate results then sort and filter through them.

Our login page has a simple, elegant framework, where the login is centered in the screen with the sierra logo above. There is a hyperlink below the login button for users who do not have an account yet, which takes them to a sign-up page. Once signed in, the users have an easy-to-use drop-down menu to choose actions from.

3.2 Hardware Interfaces

Sierra Software is hosted on a Linux server. The server can have multiple internal hard drives to store large databases, leaving plenty of room for the database to grow. There is no limit to the size of internal storage available for these Linux servers.

3.3 Software Interfaces

The Sierra Software Server runs on Linux, a free, open-source operating system. Linux can be installed on many old machines, making them efficient web servers. The server is run through the terminal, where the puma web server displays output from the operating of the server. Data is stored using SQLite3, a gem used by Ruby on Rails to easily interface databases to code. The server uses the gem ‘paperclip’ to allow file uploading and file attachment. These files are stored in the database in a hash format, and will be deleted with the game.

3.4 Communications Interfaces

The server uses HTTP GET requests to extend services over the Internet to clients. These GET requests are responded by sending certain HTML and Javascript files, as well as encrypted data, to the client’s browser. These GET requests are displayed in the server terminal to aid in troubleshooting errors. Users have no interaction directly with the GET requests, the browser generates them based on the URL, and the server responds, where the browser interprets and displays the data.

4. System Features

4.1 Use case name and identifier - View Games (U1)

4.1.1 Description and Priority

Users will be able to view all the games in the database in a clean, easily-understood interface without logging in. Games should be neatly and efficiently organized. This requirement is high priority, as this is the backbone of the software.

4.1.2 Stimulus/Response Sequences

4.1.2.1 Basic Flow

1) User navigates to website URL

2) Server displays table of all games

3) User clicks ‘View’ button on game

4) Server displays all information about a game in an engaging format

4.1.2.2 Alternative Flow - After searching through multiple pages, user decides to return to home

1) User presses ‘Sierra’ image in the upper left corner

2) Server displays table of games unchanged

4.1.2.3 Exception Flow - At step 2, the database does not respond

1) Server returns internal server error page

4.1.3 Functional Requirements

REQ-1: The system shall display the table of all games when the website is navigated to by URL

REQ-2: The system shall display the table of all games when the Sierra image is clicked

REQ-3: The system shall display all information about a game when the ‘View’ button is clicked

REQ-4: The system shall display the Internal Server Error page if the database is unreachable or if there is a fault in the Game

4.2 Use case name and identifier - Sort Games (U2)

4.2.1 Description and Priority

Users will be able to sort all the games in the database in a clean, easily-understood interface without logging in. The sort function will be clearly visible and intuitive to the user. This requirement is high priority, as this is a very important functions for users who want to see the cheapest games. Sorting is a very important component to making a database of games useful to a user.

4.2.2 Stimulus/Response Sequences

4.2.2.1 Basic Flow

1) User clicks on the heading of one of the table columns

2) The browser sorts all games in the current table according to the column the user selected

3) If a column is already the sorting query, the browser will reverse the order of all games in the table

4.2.2.2 Alternative Flow - After sorting, user decides to click the Sierra logo

1) User presses ‘Sierra’ image in the upper left corner

2) Server displays table of all games in their unsorted order

4.2.2.3 Exception Flow - At step 2, the browser does not interpret the JavaScript

1) The server will not return any response, the browser will not take any action

4.2.3 Functional Requirements

REQ-1: The system shall display the table of all games

REQ-2: The system shall sort the table of all games when a column header is clicked

REQ-3: The system shall reverse the order of the sorting if a column header is clicked twice

REQ-4: The system shall take no action if the browser does not enable the Javascript sorting function

4.3 Use case name and identifier - Filter Games (U3)

4.3.1 Description and Priority

Users will be able to filter all the games in the database in an easily-understood interface without logging in. Games should be filtered by platform and price,

allowing the user to view only games that pertain to them. Games should be neatly and efficiently organized. This requirement is high priority, as it will improve the user's experience with the software.

4.3.2 Stimulus/Response Sequences

4.3.2.1 Basic Flow

1) User types the desired platform name in the search bar near the table

2) Browser finds and displays all games in the current table with a platform that matches

3) User clicks 'View' button on game

4) Server displays all information about a game in an engaging format

4.3.2.2 Alternative Flow - After applying a filter, the user decides to return to home

1) User presses 'Sierra' image in the upper left corner

2) Server displays table of games of all games in the database, unfiltered

4.3.2.3 Exception Flow - At step 2, the browser does not accept the Javascript

1) Server takes no action, Browser takes no action. Games remain unfiltered

4.3.3 Functional Requirements

REQ-1: The system shall display the table of all games

REQ-2: The system shall filter the table of all games when a query is typed into the Live Search box

REQ-3: The system shall display all games unfiltered if the Sierra logo is clicked

REQ-4: The system shall take no action if the browser does not enable the Javascript sorting function

4.4 Use case name and identifier - Log In/Sign Up (U4)

4.4.1 Description and Priority

Users will be able to log in, or sign up if they do not have an account. Logging in will introduce User functionality like adding games, commenting on games, or favoriting games (not yet implemented). This requirement is medium priority, as this is somewhat important, but the website can operate without it.

4.4.2 Stimulus/Response Sequences

4.4.2.1 Basic Flow

1) User clicks Log In/Signup button on home page

2) Server displays user login table with a link to sign up beneath

3) User enters credentials and clicks enter

4) Server displays homepage with the login button updated to show username

4.4.2.2 Alternative Flow - User does not have account, so they click sign-up instead of logging in

1) User clicks ‘Sign Up’

2) Server displays form for User to enter their information

3) User clicks ‘Submit’

4) Server saves user account data and logs the User in

4.4.2.3 Exception Flow - At step 3, the server fails to log the user in because it thinks a session is already in progress

1) Server returns internal server error page

4.4.3 Functional Requirements

REQ-1: The system shall display a login page, prompting the user for username and password

REQ-2: The system shall display a sign up page when the Sign Up button is clicked, prompting the user for username, email, and password

REQ-3: The system shall save user information when the ‘Sign Up’ button is clicked

REQ-4: The system shall display the Internal Server Error page if the server enters a fault stage while creating a session or user.

4.5 Use case name and identifier - Submit Game Application (U5)

4.5.1 Description and Priority

Users will be able to submit an application to have a game added to the database. The user must be logged in to perform this function, to prevent spam applications. Applications will be stored in a temporary database while waiting for approval. This requirement is medium priority, as it is a convenience to users to add games

that they enjoy that are not in the database, and it will improve the database and website.

4.5.2 Stimulus/Response Sequences

4.5.2.1 Basic Flow

1) User (after signing in) clicks ‘Add Game’ in the drop down menu

2) Server displays form to get information for new application

3) User clicks ‘Submit’ button

4) Server displays “Your Application has been received”

4.5.2.2 Alternative Flow - After clicking ‘Add Game’, the user decides not to and clicks back

1) User presses ‘Back’ button

2) Server displays home page

4.5.2.3 Exception Flow - At step 3, the input requirements are invalid (missing field, no file attached, etc)

1) Server does not submit application and displays error message so user can correct their mistake

4.5.3 Functional Requirements

REQ-1: The system shall display the form for users to submit a new game application.

REQ-2: The system shall validate the information being submitted to ensure all fields are filled out before saving game

REQ-3: The system shall save the game application and store it in the Application database until it is approved

REQ-4: The system shall display the Internal Server Error page if the database is unreachable or if there is a fault in the Game

REQ-5: The system shall display the error message if there are errors validating Application data

4.6 Use case name and identifier - Approve Game Application (U6)

4.6.1 Description and Priority

Administrators should be able to approve game applications to automatically add them to the full Game database. Upon clicking ‘Approve’, a game is added to the game database that is a clone of the Application, and the approved application is hid from the Pending Applications screen, but remains in the Application database. This is medium priority, because it depends on the user being able to submit an application, so it is the same priority level.

4.6.2 Stimulus/Response Sequences

4.6.2.1 Basic Flow

1) Admin clicks on ‘Manage Applications’ from the drop-down menu

2) Server displays table of all un-approved applications - either waiting for approval or rejected applications

3) Admin clicks ‘Approve’ on an application

4) Server displays the updated table of un-approved applications

4.6.2.2 Alternative Flow - Admin decides not to approve a game at this time, and returns to the home page

1) Admin presses ‘Sierra’ image in the upper left corner

2) Server displays home page

4.6.2.3 Exception Flow - At step 3, the database says that the arguments from Application are not valid for creating a game

1) Server returns detailed error message to admin

4.6.3 Functional Requirements

REQ-1: The system shall display the table of all un-approved applications

REQ-2: The system shall create a new game from the application fields when the approve button is clicked

REQ-3: The system shall retain the application in the database even after approval for record-keeping

REQ-4: The system shall display the Internal Server Error page if the database is unreachable or if there is a fault in the Game creation process

4.7 Use case name and identifier - Reject Game Application with comments (U7)

4.7.1 Description and Priority

Administrators should be able to reject game applications with comments. Upon clicking ‘Reject’, an application display page should be rendered with a comment box below it. The admin should be able to enter comments, then click ‘Reject Application’. This is medium priority, because it depends on the user being able to submit an application, so it is the same priority level.

4.7.2 Stimulus/Response Sequences

4.7.2.1 Basic Flow

1) Admin clicks on ‘Manage Applications’ from the drop-down menu

2) Server displays table of all un-approved applications - either waiting for approval or rejected applications

3) Admin clicks ‘Reject’ on an application

4) Server displays the application and a comments box

5) Admin enters comments and clicks the ‘Reject’ button

6) Server displays the updated table of un-approved applications

4.7.2.2 Alternative Flow - After clicking ‘Reject’ initially, the admin decides not to reject the game after all

1) Admin presses ‘Back’

2) Server displays table of un-approved applications unchanged

4.7.2.3 Exception Flow - At step 5, the database fails to update the application

1) Server returns the exact error to the admin

4.7.3 Functional Requirements

REQ-1: The system shall display the table of all un-approved applications

REQ-2: The system shall allow the admin to submit comments when rejecting a game

REQ-3: The system shall retain the application in the database even after rejection for record-keeping

REQ-4: The system shall display the Internal Server Error page if the database is unreachable or if there is a fault in the application rejection process

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The server must be able to operate in near-real-time to provide the client with a fast, efficient browsing process. Slow communication and processing can lead a user to stop using Sierra Software. The server should be connected with at least a 100Mbps connection to the Internet, with suitable bandwidth to allow many clients to connect at once without seeing any performance reductions. The server should also be able to

search and order the database quickly, returning the result back to the user when a search query is entered.

5.2 Safety Requirements

1. Application validates every field before creating application.
2. The submission of new games need to be approved by the administrator to avoid fake games, viruses and other unsafe factors.
3. Stored user password in an encrypted hash code in the database
4. Need to verify the user name, password and confirm the password before creating a user account to combat fake or incomplete accounts

5.3 Security Requirements

1. The user shall not be permitted to login if he has a invalid password and accountID.
2. System will validate username,email,and password and password_digest.
3. The password is represented as dots at login.

5.4 Software Quality Attributes

Our website is run on Ruby on Rails,which has a special test suite via a gem called Minitest. It can test our models through the controller to ensure the initializers and parameter checking methods are performing correctly. It also tests to ensure that validation is working correctly.

Additionally, our Ruby on Rails application has a framework that can be applied to many different types of media, including movies, books, software packages, among

other things. Our application also allows users to decide how many elements per page they wish to view, allowing for greater client customizability. The user interface is also intuitive for users to search, sort, and filter through the Game database.

Sierra Software Store is designed to be intuitive for users, with a sortable table, live search bar, and clearly labeled actions in the user drop-down menu.

5.5 Business Rules

Normal users with and without accounts can browse and search the Sierra Store site for games. These users can also view the game page, description, and be linked off to an external site to purchase the game.

Once users log in, they are able to submit applications for a game to be added into the database. In future releases, they will also be able to comment on and favorite games.

Admins have full privileges over the database contents for users, games, and applications. Project Developers have the administrator level of permissions. Admins can also approve and reject applications submitted to add a game into the Sierra Store database.

6. Other Requirements

There are dependencies for many of the gems utilized by the Ruby on Rails project. These dependencies will present as errors installing gems when running the initialization script. All dependencies can be found at https://rubygems.org. These

dependencies can be installed on Linux with a command similar to ‘sudo <*package_manager*> install <*package_name*>’. This varies with distributions, so check how to install packages in your distribution before continuing.

This project can be reused in many different applications as referenced above by changing the database schema slightly. The current database schema can be viewed under the ‘schema.rb’ file found at ‘sierra/db/schema.rb’ on the server.

Before running the server, please run the ‘initialize.sh’ script to install gems and migrate the database properly.

Appendix A: Glossary

Github- a web based hosting service used for version control for computer code

Linux-hosted web server- A computer that runs a Linux Operating System and accepts and responds to client connections

HTML- Hypertext Markup Language. A language used for creating web pages and applications

Javascript- A language that is integrated with HTML. Can be used to expand the functionality of HTML.

Ruby on Rails- A server side web application framework that uses the language Ruby. Ruby alone is a object-oriented programming language. Rails is the integration of Ruby for web development

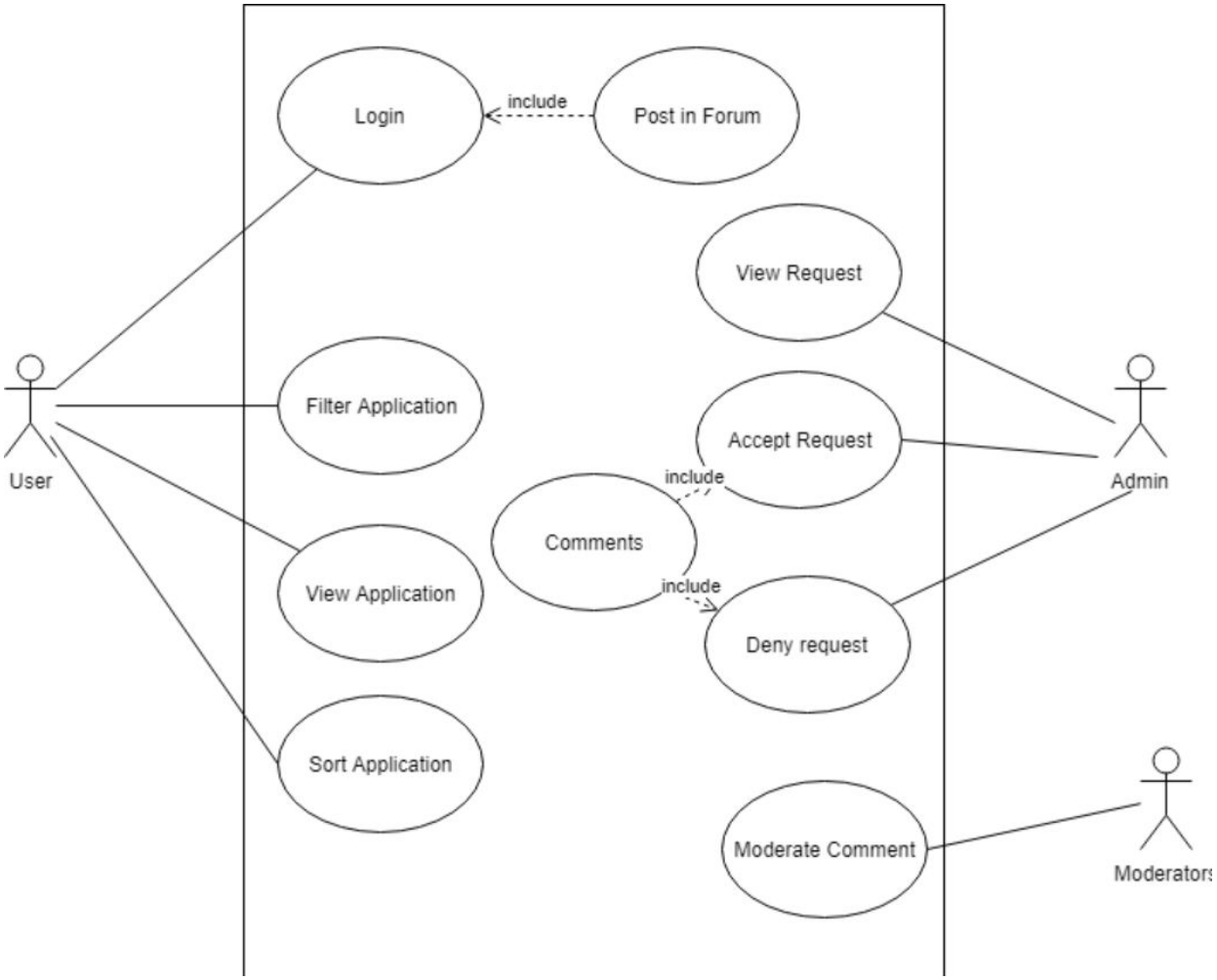
Gems- Software packages used by Ruby on Rails to perform specialized functions. Ex: SQLite3 is designed to create and manage a SQL-based database

Bash- A scripting language used in Unix-based systems

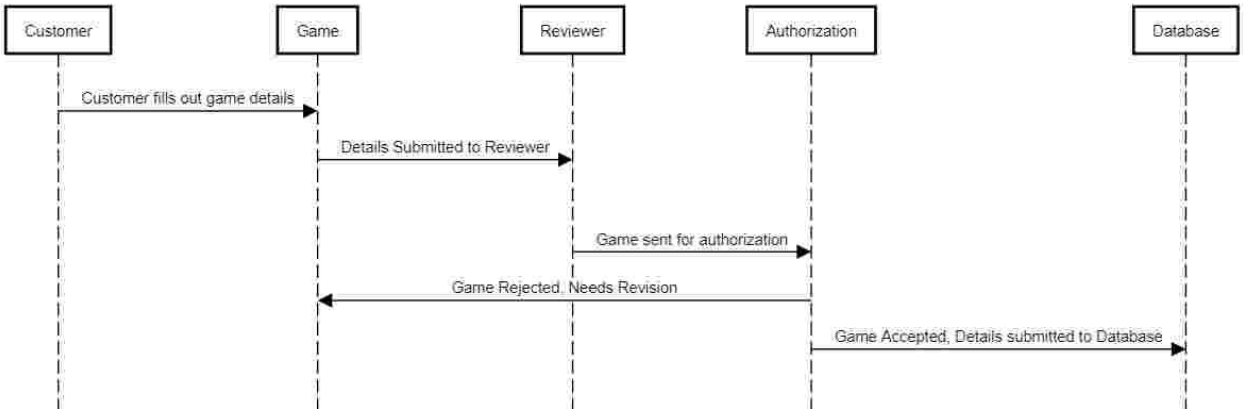
SQLite3- A Ruby on Rails gem that allows for the creation and management of a SQL-based database

HTTP GET- HTTP (Hypertext Transfer Protocol) uses different methods to transfer information. A GET request requests a file (or collection of files) from the server, and responds by sending the file to the client.

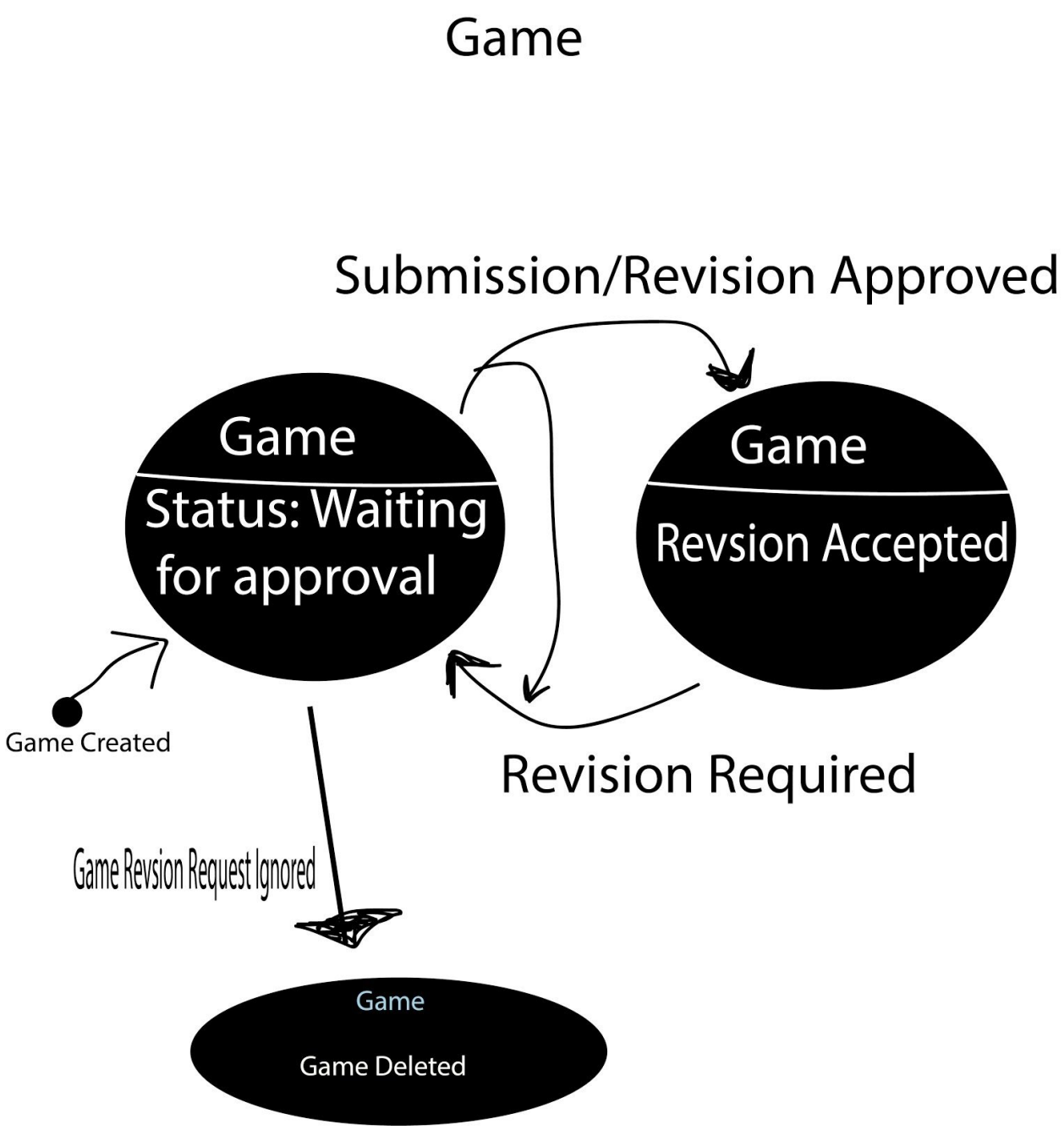
Appendix B: Analysis Models



F-1 Use Case



F-2 Sequence Diagram



F-3 Statechart

Appendix C: To Be Determined List

1) Enable a comments section of games
2) Allow users to view applications they have submitted in the past
3) Allow users to favorite games
4) Enable Sierra Software on mobile devices
5) Allow users to save search lists
6) Allow users to make lists of games and add games to them