Software Requirements Specification (SRS)

"HAIL-TOO"

Team Members:

Christopher T.

Christina T.

Jameka H.

Wesley W.

Table of Contents

Contents

1.0 - Introduction

The following document is the HAILToo SRS. This documents intended use is for customer and stakeholder buy-in and approval. This document will explain how HAILToo is intended to be used and how it is supposed to interact with all external counterparts.

1.1 - Purpose

The purpose of this document is to collect information that embodies the requirements of our system. This document will contain information about how the system should be developed by laying out functional and non-functional requirements. This document will give detailed descriptions of how HAIL-Too will interact with the users. It will also illustrate the purpose of the development of this system. It will contain the constraints of the system and its use of external applications. The intended audience is to include the customer, stakeholders and HAILToo team members.

1.2 - Scope

The HAILToo team is developing a “Clue-Like” game application to work on various browsers and accessible from anywhere in the world. The application should be free to use from either a mobile device or desktop platform. HAILToo customers can connect to the application from their updated web browser via internet connection. As long as the customer has internet connectivity, they will be able to reach our application.

The application will be managed by HAILToo administrator via a web-portal. The administrator will use the web-portal to administer the system and keep the game application available to the user and up-to-date. All application information that is saved by the user will be stored on a server.

1.3 - Glossary

Term	**Definition**
User	Someone who interacts with the application
Admin/Administrator	System administrator who is given specific permission for managing and controlling the system
Web-Portal	A web application which present special facilities for restaurant owner
Application	Short name for HAILToo’s Clue game application.
Stakeholder	Any person who has interaction with the system who is not a developer.
HCI	Human-Computer Interaction

2.0 - Software Architecture

The HailToo Clue-less game is made up of two subsystems – the in-browser client, and the web service. Each subsystem is quintessential for the ability of the system to provide a multiplayer interactive board game. The subsystems strongly rely on network communication mechanism to properly interact (across the internet), and hold a one-to-many relationship from server to client.

2.1 - Game Client

The client application will be accessible through the user's web browser and provide the essential *clue*-like interactions. This subsystem represents the user interface of the game to multiple players concurrently. Here users will be able to view the game board, select the piece which represents them on the board, and make direct moves in the game (moving from room to hallway, attempting to solve the mystery, etc).

2.1.1 - Information Domain

The client predominantly handles data which the user inputs into the browser. The user will provide inputs via keyboard strokes, mouse clicks, or touch-events to convey their desire to make moves within the clue game. The client will transmit move-related information to the server and periodically receive game state information from the server. The game state data will be transposed onto the virtual game board for each user in the game.

2.1.2 - Functional Domain

The client will provide functionality to accept user input for making moves within the game, make asynchronous calls to the server to validate said moves, and display the results on screen. To accurately display the current game state, the client will frequently poll the server and render the changes. Other functional pieces the client should provide are game creation and joining capabilities.

2.1.3 - Interface

The game client will interact with the server in an effort to maintain game-state through numerous serialized messages. The exchanged messages will fall under one of the following categories:

- PlayerMove – This type of message contains the information specific to a player's move within the game. When a player makes a move, a PlayerMove message will be transmitted to the server, and upon validation, all clients within the game will receive the PlayerMove message indicating a need to update the gameboard within the client system.
- GameStatus – This type of message incorporates information about an instance of the Clue game, whether it is active or inactive, what players have joined, and the solution information (if it has been solved). The GameStatus will also contain metadata regarding the instance such as when it was started, the number of moves made, and other relevant details.

2.2 – Server

The server application will be made available on an internet-facing machine to be interacted with by clients on internet-connected devices. This subsystem is the orchestrator of numerous concurrent, ongoing game instances. As the orchestrator, the server is responsible for managing each game in [some variation of] memory – creating the solution to the game's mystery, persisting where each player is on the board, validating player's moves as they're being made, and other activities.

2.2.1 – Information Domain

A majority of the data transmitted throughout the game originates from the server. The types of information handled pertain to the game, its players, metadata about each game instance. Players have no direct access to the server, all data flowing through the server should be initiated by input/request from the client system, with the exception of the administrative portal. Administrative portal data will mostly consist of statics/metrics about the games being served and the players connected.

2.2.2 – Functional Domain

As previously mentioned, the server performs a number of tasks but its primary focus is on game state persistence. Utilizing memory/cache, each instance of the clue game will be persisted detailing each player's position, the solution, timestamps and other metadata.

2.2.3 – Interface

The message/object types described by the client are shared with the server. These objects will be serialized and transmitted between the client and server through each game to synchronize state amongst all players.

3.0 - Non-functional Requirements

NFR	Requirement
1	The system shall be easily accessible.
1.1	The system shall be accessed by a web browser.
1.2	The system shall be available on the internet.
2	The system shall provide a graphical user interface (GUI).
2.1	The GUI shall accept user inputs (e.g. keyboard/mouse/touch).
2.2	The GUI shall provide navigation menu (e.g. start/join game).
2.3	The GUI shall display the game board.
2.4	The GUI shall be intuitive to use.
3	The system shall provide a Clue-like game experience.
3.1	The system shall enable Clue gameplay.
3.2	The system shall enforce the rules of Clue.
4	The system shall implement security best-practices.
4.1	The system shall maintain the integrity of game data.

4.2		The system shall maintain the confidentiality of user data.
4.3		The system shall maintain high availability.

3.1 - Requirements Breakdown (Non-Functional)

Non- Functional Requirement 1

Title: The system shall be easily accessible.
Description: The means by which any player accesses the Clue game is simple, intuitive and is least-resistant as possible.
Reason: Being accessible through a web browser maximizes the potential user base. Most internet-connected devices with a graphical user interface offer a web browser, so rather than requiring users to install software or go through setup routines we're able to *lower the bar of entry* to simply having an internet-connected device!

Non- Functional Requirement 2

Title: The system shall provide a graphical user interface (GUI).
Description: The system shall present an intuitive, easy-to-navigate gameplay interface for users.
Reason: The need for an intuitive user interface is essential for successful gameplay and a positive HCI user experience. Basic widgets for making moves and navigating through game play and content, including but not limited to: buttons, tweet messages (for both prompting and displaying game state notifications to user), appropriate color variation, and sound effects will make for a more pleasant user experience with the intended play-through quality.

Non- Functional Requirement 3

Title: The system shall provide a Clue-like game experience.
Description: The system shall create, persist, transmit and process data to facilitate a multiplayer game resembling Clue.
Reason: The desire of the customer is to implement the board game Clue.

Non- Functional Requirement 4

Title: The system shall implement security best-practices.
Description: Employ best practices to ensure that our systems are secure.
Reason: Internet users have been betrayed by entertainment providers many times in the past and now have immediate skepticisms towards new and hip games. In order to welcome more users (and maximize the potential user base) HailToo will focus on security implementations so that our users can establish trust with us.

4.0 - Functional Requirements

FR		
1		The client shall provide basic widgets to begin game and view game content.
1.1		The client shall provide button to join game.
1.2		The client shall provide button to create game.
1.3		The client shall display the game board and all game features.
2		The client shall maintain game state and communicate to player as appropriate.
2.1		The client shall periodically poll for GameStatus.
2.2		The client shall update game board display.
2.3		The client shall display notifications to the screen.
3		The client shall allow players to make game “moves”.
3.1		The client shall accept a player’s attempt to move their game piece.
3.2		The client shall allow player to solve the mystery.
4		Client-server communication shall be maintained at all times.
4.1		The client shall transmit PlayerMover data to the server.
4.2		The server shall inform client of game play options.
4.3		The server shall notify client of its turn.
4.4		The server shall notify all clients when a player exits the game.
5		The server shall operate on designated frameworks.
5.1		The client shall operate on a JavaScript framework.
5.2		The server shall operate in a UNIX-based environment.
6		The server shall demonstrate high availability and connectivity.
6.1		The server shall provide availability to “three 9s” (99.9%).
6.2		The server shall maintain consistent internet connectivity.

7		The server shall manage game state and records.
7.1		The server shall accommodate between 2 and 6 players per game instance.
7.2		The server shall accommodate at least 20 concurrent game instances.
7.3		The server shall persist game state.
7.4		The server shall persist record of users.
7.5		The server shall validate each player's move.
7.6		The server shall validate player's attempt to solve.
7.7		The server shall initialize game instance upon request.
7.8		The server shall update game instance to accommodate new players.

5.0 - Implementation Constraints

Constraints may vary as the project continues through development and deployment. Below are some constraints that we foresee possibly being an issue.

5.1 - Constraint List

5.1.1 - Time:

Time is always a factor when developing any kind of project. For HailToo, we have a limited amount of time (the semester length) to develop and deploy the solution. Considering that the HailToo team is working on this solution as a side project, HailToo development time is limited.

5.1.2 - Personnel:

The HailToo team is limited in personnel and capacity. We all have varying backgrounds and aren't dedicated to developing software full-time.

5.1.3 - Budget:

Money may be a factor in keeping our personnel around for a long period of time. Considering that HailToo is a not-for-profit development effort, we may have issues keeping our team together throughout the semester. Budget is essential in development and deployment of software projects.

5.1.4 - Other Resources:

TBD – Will vary as project development progresses.