Functional Specification

UFCDLE

Ruadhan McCloskey

19444786

Daniel McCartney

19395186

09/11/2022

** This document contains the general outline that you should adopt for your Functional Specification **

Functional Specification Contents

0. Table of contents

A table of contents with pages numbers indicated for all sections / headings should be included.

1. Introduction

1.1 Overview

Provides a brief overview of the system / product to be developed. It should include a description of the need for the system, briefly describe its functions and explain how it will work with other systems (if appropriate).

We are creating a WORDLE type guessing game for UFC fighters. The first thing we need for this application is a database of all of the fighters on the UFC roster. Luckily for us this is available on the UFC website. We will need to assign attributes to the fighters also such as their height, age, record, weight class, fighting style etc. This information can all be found on the UFC website also. We will need a randomize function to choose a random fighter in the database for the user to guess. We will also need a function for the board generation for when you guess a fighter, we need their attributes to show up and we need to color code the attributes which are the same or similar to the fighter that they're trying to guess. If they have the same attribute such as "Height" then this will show up green or if their height is within 2 inches below or above the height of the fighter you're trying to guess it will show up yellow with an arrow pointing up or down depending if the fighter is taller or shorter than the fighter that you're trying to guess. We also want to implement a versus mode and a timed mode that you can play online against other users or your friends. For this we will need a time function and a server to allow 2 users to play against one another.

1.2 Business Context

Provides an overview of the business organization sponsoring the development of this system / product or in which the system / product will / could be deployed. *Note - may not be applicable to all projects*

Our application will not be used for profit more so to promote some of the lesser known fighters in the promotion and to create a fun game for UFC fans to play against each other.

1.3 Glossary

Define and technical terms used in this document. *Only include those with which the reader may not be familiar.*

2. General Description

2.1 Product / System Functions

Describes the general functionality of the system / product.

The first thing we need to do is get the actual game working so we need to be able to query from the database and have the base of the game working before we move onto multiplayer and the different modes that we want to implement. For this we will need the randomize function to pick a random fighter for the user to guess, we will need a search box for the user to search the fighter that they want to guess, we will need our board generation working and we will also need a “How to play” option which outlines the rules of the game. We will also need a set of attributes for each fighter eg. Weight class, height, reach, country, fighting style, age, record.

2.2 User Characteristics and Objectives

Describes the features of the user community, including their expected expertise with software systems and the application domain. Explain the objectives and requirements for the system from the user's perspective. It may include a "wish list" of desirable characteristics, along with more feasible solutions that are in line with the business objectives.

All we really need from the user is somebody who is familiar with the UFC and knows some of the fighters. Our target audience will be people who watch the UFC. As far as using the application it will be quite straightforward and we will have the rules as well for users who aren’t familiar with such games. We want to provide a fun game for the users to play where they can guess their favorite fighters and compete with their friends and find out who has the better knowledge on the sport.

2.3 Operational Scenarios

This section should describe a set of scenarios that illustrate, from the user's perspective, what will be experienced when utilizing the system under various situations.

In the article Inquiry-Based Requirements Analysis (IEEE Software, March 1994), scenarios are defined as follows:

In the broad sense, a scenario is simply a proposed specific use of the system. More specifically, a scenario is a description of one or more end-to-end transactions involving the required system and its environment. Scenarios can be documented in different ways, depending up on the level of detail needed. The

simplest form is a use case, which consists merely of a short description with a number attached. More detailed forms are called scripts.

Scenario 1:

The user logs into the game and chooses to play the “Daily Challenge”. They guess the fighter “Leon Edwards”. The only similar attribute that they guessed was the weight class which will then show up green.

Scenario 2:

The user is on their final guess and they have some known attributes from their previous guesses. They know the fighter is a featherweight, with a 72 inch reach and that he is 34 years of age. Based on this the user guesses Calvin Kattar which is the correct answer. The system will show a picture of the fighter and their name will show up with a congratulatory message attached also with how many guesses it took them to get the answer.

Scenario 3:

2 users log onto the game to play timed mode against one another.(This is where 2 players compete to guess as many fighters as possible within a certain timeframe.) User 1 guesses 3 fighters in 5 minutes and user 2 guesses 4 fighters in 5 minutes. User 2 wins and receives a message indicating that they have won the game.

2.4 Constraints

Lists general constraints placed upon the design team, including speed requirements, industry protocols, hardware platforms, and so forth.

I believe one of the main constraints that we will have is getting a server to use and for multiplayer modes to work also. Another difficulty we may have is the board generation, getting the attributes of the fighter that you guess to show up in the correct order with the correct color coding for the attributes that are the same as or similar to the fighter that you’re trying to guess.

3. Functional Requirements

This section lists the functional requirements in ranked order. Functional requirements describes the possible effects of a software system, in other words, *what* the system must accomplish. Other kinds of requirements (such as interface requirements, performance requirements, or reliability requirements) describe *how* the system accomplishes its functional requirements.

As an example, each functional requirement could be specified in a format similar to the following:

- **Description -** A full description of the requirement.
- **Criticality -** Describes how essential this requirement is to the overall system.
- **Technical issues -** Describes any design or implementation issues involved in satisfying this requirement.
- **Dependencies with other requirements -** Describes interactions with other requirements.
- **Others as appropriate**

Fighter Database:

- **Description -** A full database of fighters in the UFC organization.
- **Criticality -** Will be critical to the success of our project. We need this in order for our project to be feasible.
- **Technical issues -** The UFC has a database of their fighters on their website with all of the necessary attributes that we need. Potentially we could have difficulty with taking out some of the attributes that we don't need.
- **Dependencies with other requirements -** Everything is dependent on this database, without it the application won't work.
- **Others as appropriate**

UI:

- **Description -** We will need to design an easy to use simple UI that has a nice aesthetic to it.
- **Criticality -** This will be important for the game to look good and run smoothly. UI is important for any game, if a game doesn't look good people won't want to play it.

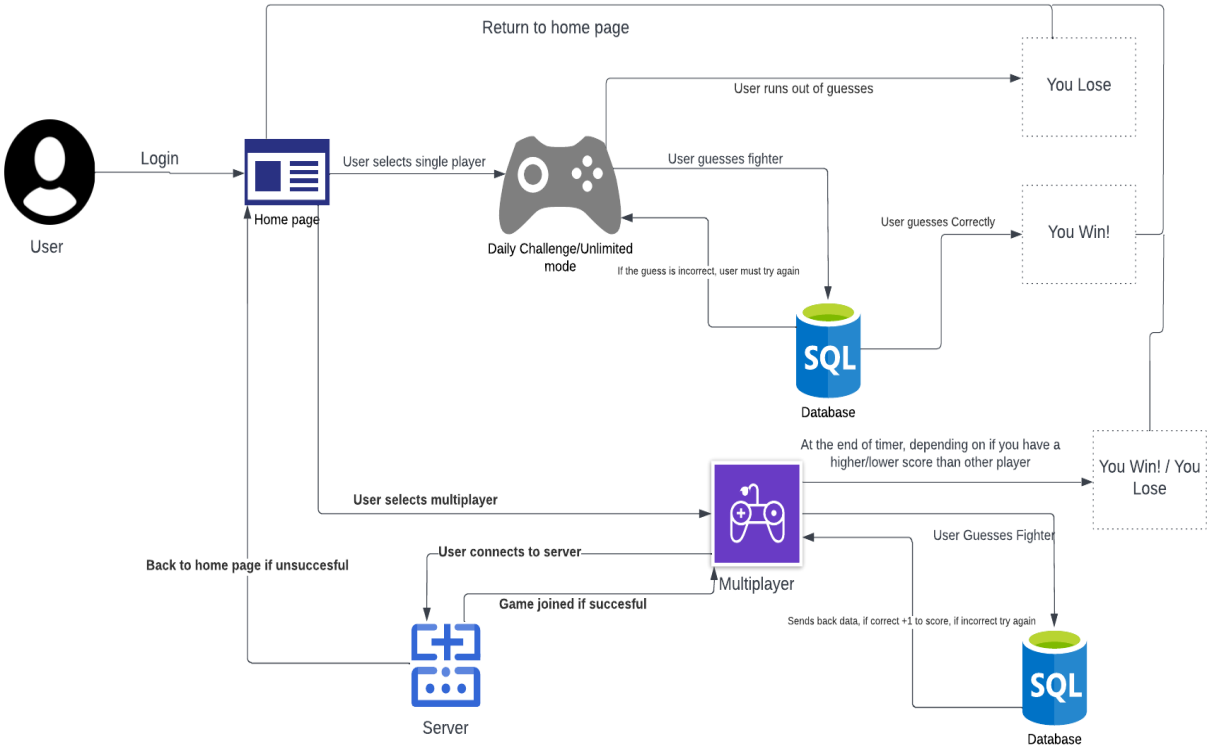
- **Technical issues -** It’s one thing to design a nice UI for your system, it's another thing to have that UI work smoothly with the front and backend.
- **Dependencies with other requirements -** The UI will be essential for the overall system to work, if we design a nice UI that works properly then we can build other modes and games off of it.
- **Others as appropriate**

Board Generation:

- **Description -** We will need to be able to generate a board of the attributes of fighters that have been guessed. We will also need the attributes to be colour coded in relation to the attributes of the fighter that the user is trying to guess eg. green for common attributes and yellow for attributes that are close to one another.
- **Criticality -** This will be essential to our application as this feature will be used in every game mode we make off of the original game.
- **Technical issues -** This could prove to be easier said than done as we will need to query the database every time a fighter is guessed.
- **Dependencies with other requirements -** This feature is also tied into the previous 2 features as we will be querying the fighter database and it will be a part of the UI also.
- **Others as appropriate**

4. System Architecture

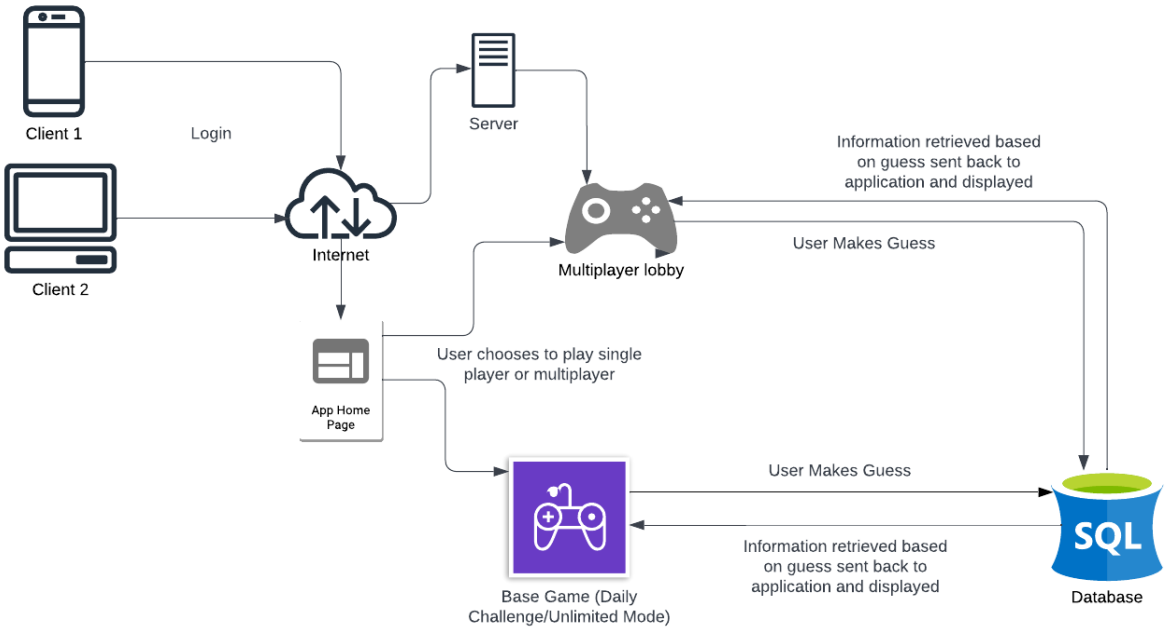
This section describes a high-level overview of the anticipated system architecture showing the distribution functions across (potential) system modules. Architectural components that are reused or 3rd party should be highlighted.



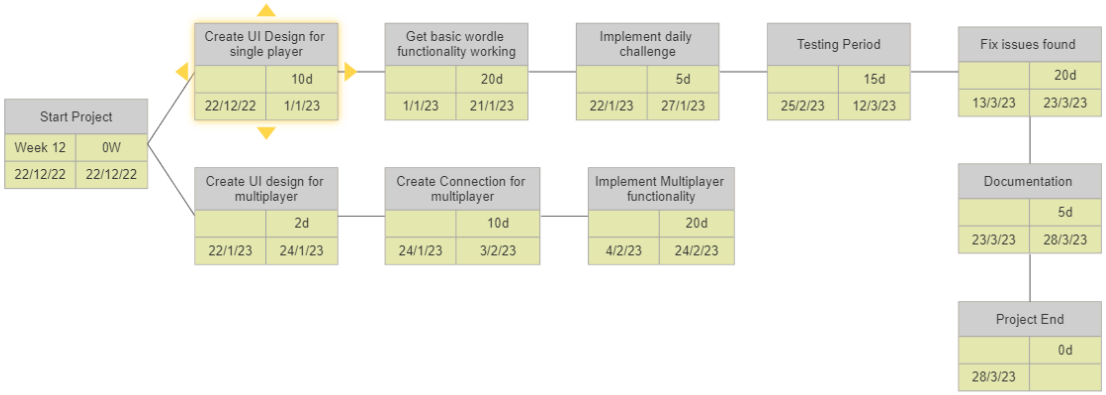
5. High-Level Design

This section should set out the high-level design of the system. It should include one or more system models showing the relationship between system components and the systems and its environment. These might be object-models, DFD, etc.

High Level Design



6. Preliminary Schedule



7. Appendices

Specifies other useful information for understanding the requirements.