Quark App

System Requirements Specification

Version 1.0 approved

Prepared by Jazer Barclay + Fin Watling ( a bit ) :P

University of Brighton

22nd March 2022

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Jazer Barclay | 22/03/22 | Initial Version | 1.0 |
|  |  |  |  |

# Introduction

## Purpose

The purpose for this document is to explain the process by which the preliminary requirements from our project planning were analysed and refined.

## Intended Audience and Reading Suggestions

This product is primarily targeted at knowledge workers in the academic field such as students in both foundational and further education and researchers.

This app, however, can be used for simple time tracking over extended periods of time allowing a more general usage.

## Product Scope

This product is a software application which is accessed from a desktop application and is used to track work units in increments of 30 minutes with a split of 25 minutes of work and 5 minutes of rest.

The application extends to a website that can be accessed for historic tracking data for an individual to view and to compare against other users of the application in a ranked leader board.

## References

* IEEE 830
* Business Analysis, Requirements, and Project Management (Chapter 4) [K. Cox (2022)]
* CI536 Requirements Documentation (Week 3) by Karl Cox (2022)

## Overview

This document is laid out as per the IEEE-830 standard.

# Overall Description

This system is designed to be a facility for tracking a user’s time during extended work sessions and has many applications such as recording study sessions, research on topics that require great depth and complex writing or development tasks.

## Product Perspective

Diagram

Description automatically generatedThe time tracking application is broken down into a back-end, website and desktop application. The back-end serves as the data storage system where custom user implementations can access the infrastructure for tracked session storage. The website is where the user may view their tracked information in a standard format and view the stats of other users of the system. The desktop application serves as a simple, minimal method to track their work sessions and upload the data to the back-end system.

## Product Functions

The functionality is split between the website where insight is generated and the desktop app which serves as a method of time tracking via a fixed timer. The functions are labelled for where they will need implementing to meet the user requirements.

* Signup (Website & Application)
* Login (Website & Application)
* Start timer (Application)
* Stop timer (Application)
* Log 30 minute session (Application)
* View personal historic stats (Website)
* View other user’s historic stats (Website)
* View leader board of all users (Website)

## User Classes and Characteristics

There are three types of users in this system. The first two are authorised users and the third is non-authorised users. The first is the administrator. They have direct access to the server and thus the data, infrastructure and source code. They will be responsible for hosting and managing the services exposed to the second type of user.

The second type of user is the end user. They are the authorised users, customers, who have signed up for the service and operate the desktop application. Their main interaction is via the use of the timer and accessing their profile via the website or API.

The third type of user is the unauthorised user. They have not signed up for the service and as a result have no profile data of their own on the system. They may wish to access other end user’s data on the system which they are not allowed access to until they have signed up for the service, becoming an end user.

All these users must have basic computer skills which include working with a web browser such as Google Chrome or Mozilla Firefox. Since all interactions with the UI of the desktop and website, the system cannot be used without access and knowledge of how desktop applications and specifically web browser functionality as part of their operating system.

## Operating Environment

The desktop application will be supported on any desktop or laptop device running an up-to-date version of the Windows, MacOS or Linux operating systems.

The website will be accessible by any modern web browser from any device with a minimum screen resolution of 280x360 and JavaScript support.

## Design and Implementation Constraints

* Data must be stored in a relational database for fast queries and optimal storage
* Passwords must be stored encrypted via a salt and hash function
* Unauthorised users must not be allowed to interact with the data via the website or associated API
* Secure communication between server and client via HTTPS connections
* Robust enough to handle server error responses and notify the end user

## User Documentation

Along side the source code for the application and website, a PDF manual for the server-side API will be available for the developer audience.

# External Interface Requirements

## User Interfaces

On the website, there will be a navigation panel that allows easy access to traverse between any pages. This will have a wildcard link that, depending on login status and active screen, will show different links.

The pages must conform to a standard layout and scheme being mobile responsive and mutable for desktop use.

The desktop application must remain small in size and be unobtrusive to the user’s view. To ensure its focus is available, the window will remain top most compared to other apps on the screen.

## Hardware Interfaces

The websites will be accessible via mobile and desktop devices with varying screen sizes. This must be taken into consideration for layout and design factors.

## Software Interfaces

The website and application will both share a connection to a server-side database via an API. This API will be public facing allowing direct interaction with it. All data stored in the database will only be visible via the API reducing the attack surface.

These communications must be over an encrypted connection via HTTPS.

## Communications Interfaces

Communications via the API and website must be over an encrypted connection via HTTPS.

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## Website Signup

4.1.1 Description and Priority

A new or existing user can create a new account using their given email address, username and password. This is a high priority feature that is required for any user to become an active member.

4.1.2 Stimulus/Response Sequences

The user will first access the website via search engine or url. They will click on the signup button on the home page and fill in the required form. Upon completion and successful account creation, the user will be forwarded to the login screen. Here they can use their new account details to perform a login. Should the validation of user details fail upon signup, the form will display an appropriate error.

4.1.3 Functional Requirements

Link from homepage to the signup page

Form needed on signup page

Submit form interacts with API

API checks if user exists already, if not, saves data to database and redirects user to login

## Website Login

4.1.1 Description and Priority

An existing user can login to the web application and visit the profile and leader board pages, given that the correct user details are supplied in the login form.

This is a high priority feature that is required for any user to log into their account.

4.1.2 Stimulus/Response Sequences

The user will first access the website via search engine or url. They will click on the login button on the home page, from here they will supply their login details and if correct they will be redirected to their profile page.

4.1.3 Functional Requirements

Link from homepage to the login page

Form needed on login page

Submit form interacts with API

API checks if user exists and if details are correct, if so the user is redirected to profile page.

## Application Timer Start

4.1.1 Description and Priority

An existing user can login to the desktop application and click the start button to begin tracking a unit with Quark.

This is a high priority feature that is required for any user to log units with Quark.

4.1.2 Stimulus/Response Sequences

The user will open the java file downloaded from our webpage, log in or sign up and log in with their details through the application and click the start button to begin tracking a unit.

4.1.3 Functional Requirements

Open application jar file

Log in or sign up and log in with their details

Click the start button to begin tracking units

## Website Leaderboard Display

4.1.1 Description and Priority

An existing user can login to the web application and visit the leaderboard page which displays the top 100 users.

This is a medium priority feature that we would like to implement to add extra functionality to Quark.

4.1.2 Stimulus/Response Sequences

The user will first access the website via search engine or URL. They will click on the login button on the home page, from here they will supply their login details and if correct they will be redirected to their profile page. From here they can click the leaderboard link, and this will take them to the leaderboard which shows the top 100 ranking users.

4.1.3 Functional Requirements

Link from homepage to the login page

Form needed on login page

Submit form interacts with API

API checks if user exists and if details are correct, if so, the user is redirected to profile page.

Link from profile page to leaderboard page

Leaderboard page displays top 100 users

## Website Leaderboard Filters

4.1.1 Description and Priority

An existing user can sort the leaderboard by top users this day, week, month, and year.

This feature is of low priority. We would like to add it, however it is not required.

4.1.2 Stimulus/Response Sequences

Once the user is logged in, they navigate to the leaderboard page. When viewing the leaderboard, the users will be able to press tabs at the top of the page to filter by day, week, month, and year.

4.1.3 Functional Requirements

Link from homepage to the login page

Form needed on login page

Submit form interacts with API

API checks if user exists and if details are correct, if so, the user is redirected to profile page.

From here the user navigates to the leaderboard page

Once on the leaderboard page the user can filter the view of the leaderboard

## Website View Profile Graph

4.1.1 Description and Priority

An existing user can login to the web application and visit the profile page, on this page the user can see the graph of their commits over the last two weeks.

This feature is required for our profile page to be functional.

4.1.2 Stimulus/Response Sequences

The user will first access the website via search engine or url. They will click on the login button on the home page, from here they will supply their login details and if correct they will be redirected to their profile page. On this page they can scroll down and will be presented with their units from the last 2 weeks plotted on the graph.

4.1.3 Functional Requirements

Link from homepage to the login page

Form needed on login page

Submit form interacts with API

API checks if user exists and if details are correct, if so, the user is redirected to profile page.

On profile page loading, the JS interacts with the API to collect the user data for the last two weeks, this then gets plotted on the graph for the user to see.

## Application Pausing

4.1.1 Description and Priority

In the case of some unintended event occurring, the user might want to temporarily pause the event so that they could continue to study later in time. Although helpful, it doesn’t necessarily prevent the main purpose of what the pomodoro technique does.

This can be an extra feature and isn’t required for the application

to fully function.

4.1.2 Stimulus/Response Sequences

The user will open the jar file and log in. They will click start to begin the timer; the user can click pause to temporarily stop the timer where it is at.

4.1.3 Functional Requirements

Open the jar file.

Login with details.

Click start timer to begin timer.

Click on pause.

## Application Stop and Restart

4.1.1 Description and Priority

Should the user encounter any errors, the user can stop the timer to restart the timer. This shouldn’t prevent what the main timer does so it is not as important as the timer counting down for the technique to be effective.

This isn’t required but can be a useful extra feature.

4.1.2 Stimulus/Response Sequences

The user will click on the jar file then log into the system. Click on the start timer to begin the timer, then click on stop to stop the timer and a 25-minute timer will be shown waiting to begin counting down.

4.1.3 Functional Requirements

Open Jar file.

Login with details.

Click start button to begin timer.

Click on stop.

## Application Break Timer

4.1.1 Description and Priority

When the 25 minutes study time has finished, the app should then start a 5-minute timer to let the user know that they should take a quick break. This helps maintain the users mental state by allowing them to stretch and feel refreshed.

This is a required to allow users to properly rest.

4.1.2 Stimulus/Response Sequences

The user will log in to the app. User will click on start to begin the timer. The timer will then start a 25-minute timer then a 5-minute timer.

4.1.3 Functional Requirements

Open application jar file.

Login with details.

Click start button to begin timer.

25 Minute timer ends.

5 Minute break time starts.

## Application Restarting Timer

4.1.1 Description and Priority

When the 5 minutes break time ends, it should restart the 25-minute timer again to restart the pomodoro unit of time.

This is a must have feature so that we can quickly restart the

timer.

4.1.2 Stimulus/Response Sequences

The user will log in to the app. User will click on start to begin the timer. The timer will then start a 25-minute timer then a 5-minute timer. Afterwards, it will send a unit and a 25-minute timer begins again.

4.1.3 Functional Requirements

Open application jar file.

Login with details.

Click start button to begin timer.

25 Minute timer ends.

5 Minute break timer ends.

25 Minute timer starts again.

## Application Sending Units to Database

4.1.1 Description and Priority

After each complete pomodoro time of 25 minutes study and 5 minutes break, the system needs to be able to send the unit to the API so that it can be added to the database connected to the users account to keep track of their total units. This is very important as it is the main function of our app.

This feature is required so that data can be tracked

4.1.2 Stimulus/Response Sequences

When beginning the countdown, it will countdown from 25 minutes, then 5 minutes. After the 5 minute break is over, a unit will be sent to an API which will add a unit to an existing database.

4.1.3 Functional Requirements

Open application jar file.

Login with details.

Click start button to begin timer.

25 Minute timer ends.

5 Minute break timer ends.

Unit gets sent to the API.

API will add unit to database.

## Application Access

4.1.1 Description and Priority

For the user to use the app, the app needs to be downloaded so that the user can join other members to share stats and for users to manage their time effectively. The download button needs to be able to download a jar file when clicked.

This feature is required so that users can use the app.

4.1.2 Stimulus/Response Sequences

The user will first access the website via search engine or URL. They will click on the download button on the top right. Since an account is required, it will direct you to the login page where you can signup if you don’t have an account made. Once you log in, clicking on the download button will direct you to a page showing downloads for 3 different versions. Users can then click on the windows button to download the jar file.

4.1.3 Functional Requirements

Link from homepage to downloads.

Redirects to login page if not logged in.

If logged in, will go to downloads page.

3 Clickable links are shown.