**Software Requirements Specification**

**for**

Homework Tracker

Version 3.2 approved

Prepared by Alex Sweeney

Eman Khan

Donovan Byler

William Oeffner

# Table of Contents

[**Introduction**](#_1fob9te) **4**

[**Purpose**](#_3znysh7) **4**

[**Document Conventions**](#_fgefs34d7rwt) **4**

[**Intended Audience and Reading Suggestions**](#_areh8w5ukcwc) **4**

[**Product Scope**](#_ix9ceah6ff7) **4**

[**References**](#_83pzdon3s23x) **4**

[**Overall Description**](#_cq8my2vinsxa) **5**

[**Product Perspective**](#_2s8eyo1) **5**

[**Product Functions**](#_ckaj9ru54p5f) **5**

[**User Classes and Characteristics**](#_wv7khuytr8jh) **5**

[**Operating Environment**](#_ry81mbletndn) **5**

[**Design and Implementation Constraints**](#_zaw9c0klef) **5**

[**User Documentation**](#_ht3kpfim1b91) **6**

[**Assumptions and Dependencies**](#_dsm9br3gkhs5) **6**

[**External Interface Requirements**](#_ygagb3n4u3fp) **6**

[**User Interfaces**](#_2jxsxqh) **6**

[**Hardware Interfaces**](#_erzil5nmu4wl) **6**

[**Software Interfaces**](#_jwqmjmam9yzg) **6**

[**Communications Interfaces**](#_70oqosmp2vvx) **6**

[**System Features**](#_e0s7ix1h7ita) **7**

[**Record Assignments**](#_bljeo3npxr8) **7**

[**Schedule/Prioritize Assignment Completion**](#_adzf7jy1d66t) **7**

[**Automated Class Information Importation**](#_n5bu4chkv1tc) **8**

[**Free Time**](#_1mgo6itvzd6x) **8**

[**Other Nonfunctional Requirements**](#_3whwml4) **9**

[**Performance Requirements**](#_2bn6wsx) **9**

[**Security Requirements**](#_fxkswjha6eif) **9**

[**Software Quality Attributes**](#_bxr35hhbs3u0) **9**

[**Appendix A: Glossary**](#_g8aik7txdvjn) **9**

[**Appendix B: Analysis Models**](#_pypm4zpclacy) **11**

[**Appendix C: To Be Determined List**](#_fcm6lj1bvp7y) **12**

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Whole Team | 2/11/21 | Initial Outline | 1.0 |
| Donovan Byler | 2/15/21 | System Features | 1.1 |
| William Oeffner | 2/15/21 | Other Nonfunctional Requirements | 1.2 |
| Alex Sweeney | 2/16/21 | Overall Product Description | 1.3 |
| Eman Khan | 2/16/21 | Introduction and External Interfaces | 1.4 |
| Alex Sweeney | 2/16/21 | Clean up formatting | 1.5 |
| Whole Team | 2/18/21 | Update Use Case Diagram | 2.0 |
| Donovan Byler and William Oeffner | 2/22/21 | Added Activity Diagram | 2.1 |
| Donovan Byler | 2/22/21 | Updated System Features | 2.2 |
| Alex Sweeney | 2/23/21 | Added Database Diagram | 2.3 |
| Alex Sweeney | 3/23/21 | Updated Database Diagram | 3.0 |
| Donovan Byler | 4/7/21 | Changed Course Definition | 3.1 |
| Alex Sweeney | 5/4/21 | Updated Database Diagram | 3.2 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **Introduction**

## **Purpose**

The purpose of this software is to build an online mobile and web application to manage student assignments and tasks with regards to a timeline to ease homework management.

## **Document Conventions**

Defined terms are highlighted with bolding.

Requirements will come with priority to indicate in which order they will be implemented.

Different versions might be released with only some of the total list of requirements implemented.

## **Intended Audience and Reading Suggestions**

This project is a prototype for the Homework Tracker system and it is currently focused to be used by Kent State students. It is intended for all students who are attending school and are enrolled in courses.

Below is a list of collaborators to this document. The draft will be sent to this email list for feedback and comments.

|  |  |
| --- | --- |
| Donovan Byler | dbyler8@kent.edu |
| Eman Khan | ekhan2@kent.edu |
| William Oeffner | woeffner@kent.edu |
| Alex Sweeney | asween12@kent.edu |

## **Product Scope**

The purpose of the Homework Tracker is to ease homework management and to create a convenient and easy-to-use application for users, to keep on top of assignments. The system is based on a relational database with its homework management and organizational functions. We will have a database server collecting information entered by the user as well as a clean and practical user interface to help the user navigate through the tasks and allowing them to checkmark the completed assignments.

## **References**

Used SRS template

<https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc>

Firebase Authentication documentation referenced the security section

<https://firebase.google.com/docs/auth>

# **Overall Description**

## **Product Perspective**

This product will be an entirely new offering - fully self-contained.

Everything will be handled internally within the application.

There will be a function that will scrape course information from Kent State, however, this will remain within the app and only run on an administrative level.

## **Product Functions**

The main functions of this application is to allow users to 1) Create a list of their assignments, 2) schedule and prioritize the completion of these assignments, and 3) offer a list of classes to tag the assignments to (automatically imported from Kent).

## **User Classes and Characteristics**

There will only be two major user classes in this application: Admins (us), and Students (users).

The admin role is to manage and maintain the application, including the importation of course information into the app.

The primary user class will be the students, who will input their assignment information and perform the necessary functions on them.[[1]](#footnote-0)

## **Operating Environmen**t

Our application will run entirely within the Google Firebase Platform.

The database that stores data will be hosted within Cloud Firestore.  
The front end website will be hosted by Firebase Hosting, with the dynamic content to be generated and served by Firebase Cloud Functions.

Authentication will be handled through Firebase Authentication.

The users will access the application within any modern web browser, including mobile devices.

## **Design and Implementation Constraint**s

The main constraints we are limited by are the time we have available to implement this application, and the lack of in-person meetings to plan and work on things together.

We are addressing these by implementing a strict timeline for development, and through regular online meetings where content can be shared with each other and keeping open communication throughout the project.

We are expecting limited usage of the app up front, where there is no concern to cost within the Google Firebase platform, as it is extremely unlikely that we will exceed their free tier.

## **User Documentation**

The website will include a built-in user manual to allow users to obtain guides on how to use the application.

## **Assumptions and Dependencies**

As previously mentioned, this application will run entirely on the Google Firebase platform. As of such, it is highly dependent on Firebase being available and performing as expected.

# **External Interface Requirements**

## **User Interfaces**

Tags are an essential part of the presentation of data. They will be color coded regarding the course and assignments will be found under each tag.

A calendar will display assignments with respect to their due dates so the user can plan and work accordingly.

A checkmarking option will be accessible to the user to keep track of completed tasks.

An editing option will be available to modify/update/delete an existing assignment or tag.

The program will be linked to the school’s database to gather information regarding the student’s courses.

## **Hardware Interfaces**

User’s Screen

Input Device

## **Software Interface**s

HTML

CSS

Javascript

Node.js backend

React js frontend

Python

Google Firebase

## **Communications Interfaces**

Web browser

HTTPS

# **System Features**

## Record Assignments

**4.1.1 Description and Priority**

The baseline feature of our product is the ability to record assignments and tasks with our Homework Tracker application. Unregistered users must first create an account to store all of their user information and to add any tasks or tags. Every task consists of a title, text description, and due date. For each task, the user may associate one or more tags to help organize their tasks. Tags can be made for courses, work, personal, or any other organization that the user wishes to create. To further help the user organize their tasks each tag has a color associated with it which helps the user identify which tag the task is under quickly. When a task is completed, the user may mark the task as finished Because this feature is essential to the system, the priority is high. (Add in description)

**4.1.2 Stimulus/Response Sequences**

The user records tasks, creates tags, and assigns an association between a task and tag The system will record the tasks and tags on the user’s account and display each task on the application. The system will retain all tasks, but not display completed tasks in the same section as the uncompleted tasks.

**4.1.3 Functional Requirements**

REQ-1: Creation of tags and tasks are only valid if they consist of Alphanumeric characters

REQ-2: Each task may have one or more tag associated with it

REQ-3: Tag names must be unique though task names may be repeated

## Schedule/Prioritize Assignment Completion

**4.2.1 Description and Priority**

To facilitate easy assignment completion, the user may create their own personal priority level for each task. The system sorts and displays tasks according to priority, deadline, or tag. By sorting for each of these factors, the user is better able to understand the importance and proper prioritization of each task. The first additional aspect of the prioritization feature is the ability to add in an estimated time to completion for each task. Secondly, the system also allows users to add into the system which days and times the user plans on completing the task. This feature adds extensive functionality to the product and has the assigned priority level of medium.

**4.2.2 Stimulus/Response Sequences**

The user adds in the priority, deadline, tag, and estimated time to completion for each task. This information is stored in the user’s account and may be used by the system to sort each task according to each aspect of prioritization.

**4.2.3 Functional Requirements**

REQ-1: Priority level must be an integer

REQ-2: When sorting by deadline, the earlier tasks will appear first

REQ-3: Estimated time to complete is an optional field

## Automated Class Information Importation

**4.3.1 Description and Priority**

A Python script will scraper the Kent State website for the complete catalogue of classes. Each class will then be added as a tag into the database automatically, so the user will not be required to enter any specific class tags from the beginning. The tags will then be subdivided into sections based on the department that the class is from, and it will be displayed as the class code on the schedule. When viewing a detailed version of the tag on the class page it will show the full name of the class. It is a medium priority task since it is for convenience of the users and not core to the functionality of the product, but it is still an important feature that we would like to implement.

**4.3.2 Stimulus/Response Sequences**

A Python script is run when the database is initialized or when a new semester starts, and it inputs all class data into the database as a new tag with a new identifier for the year and semester of the class.

**4.3.3 Functional Requirements**

REQ-1: class name and class number must be a string

REQ-2: All classes must be associated with a specified department at Kent State

## Free Time

**4.4.1 Description and Priority**

This feature allows the user to mark the general times that they would be available to work on their tasks. The user may enter their free times for a week and later use these times as a reference to know how much time in a week they have to accomplish their tasks. Most users would find this useful if they had consistent, weekly availability. Free time is a low priority feature.

**4.4.2 Stimulus/Response Sequences**

The user defines which time periods in the week are available for work on their tasks. The system tracks this available time and displays it to the user.

**4.4.3 Functional Requirements**

REQ-1: Free time must be given in a weekly format

# **Other Nonfunctional Requirements**

## **Performance Requirements**

Actions within the software should have a response time that is not noticed to the average user, and they should be easy to use without special instructions. This includes users being able to navigate web pages and know within a few seconds the basic way to operate the webpage. It should also not consume so much bandwidth that it becomes impractical to run the application on mobile devices.

## **Security Requirements**

The security of this application will be concerned with usernames and passwords that could potentially identify users.

In order to mitigate risk we have assigned the task to be done through Google Firebase’s authentication services. This allows us to make use of many of Google’s security features, such as a secure login.

## **Software Quality Attributes**

This application has a few key qualities that are very important to us as both creators and potentially future users. We wish for the application to be available to users at all times through many different methods of interaction, such as mobile devices and personal computers.

We also expect the application to reliably provide users with the account information they provide to best keep track of classes and not require students to potentially lose important information.

In order to help with this the application must also be intuitive to use and make sense to an end user from the moment they first try to use the application.

# Appendix A: Glossary

**Assignment.** A task that the user inputs that will be added to the schedule for completion at a

specified date.

**Account.** A collection of data within the database associated with a specific user.

**Class.** Interchangeable with Course

**Course.**  An instructional class taught by an educational institution

**End User.** See User

**Priority.** A user’s approximation of how important a task is

**Schedule.** A unique list of assignments that are linked to a user account and sorted chronologically as well as with special prioritization parameters from the user.

**Tag.** A categorization of tasks. A tag may represent class courses, work, personal, or any other organization that the user wishes to create

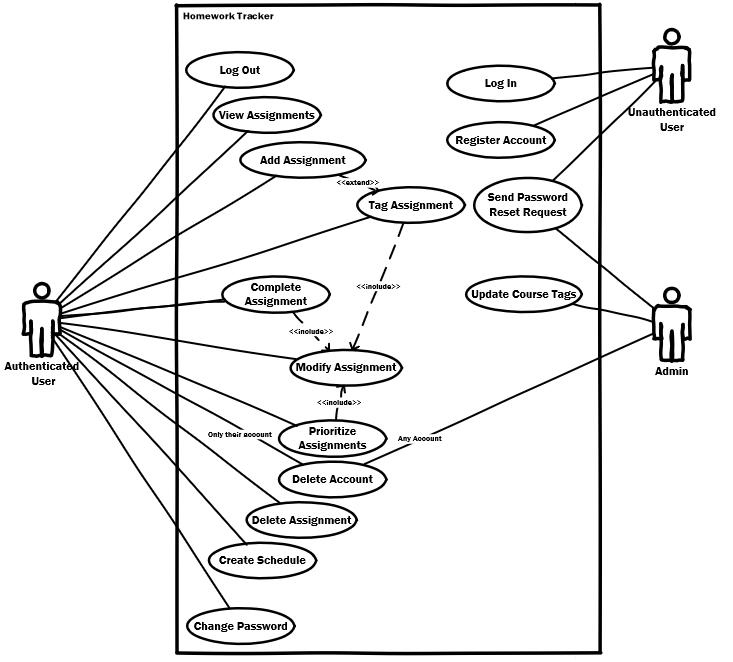
**Task.** Something which must be accomplished that the user records in the system

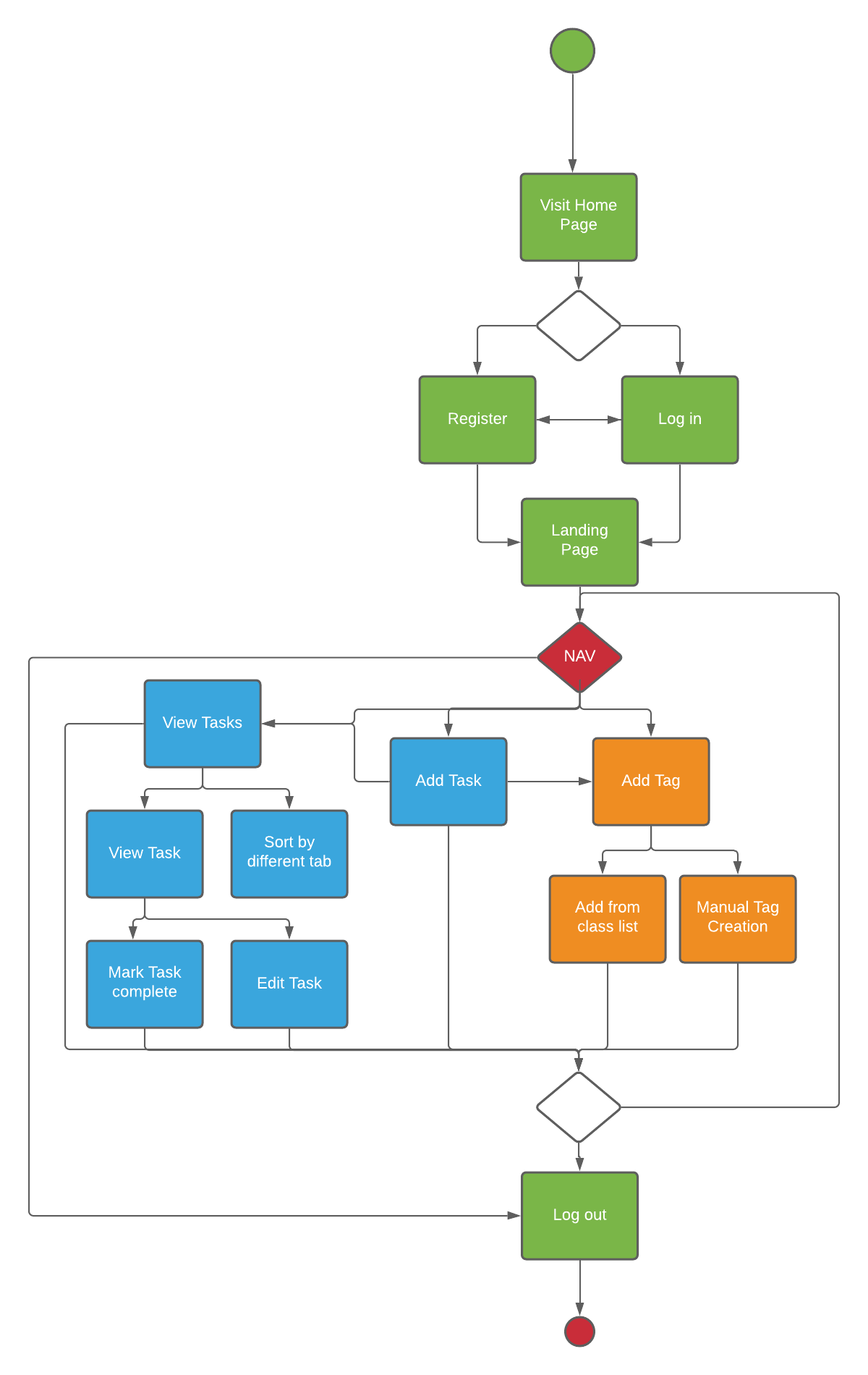
**User.** A person who is using the Homework Tracker application.

# 

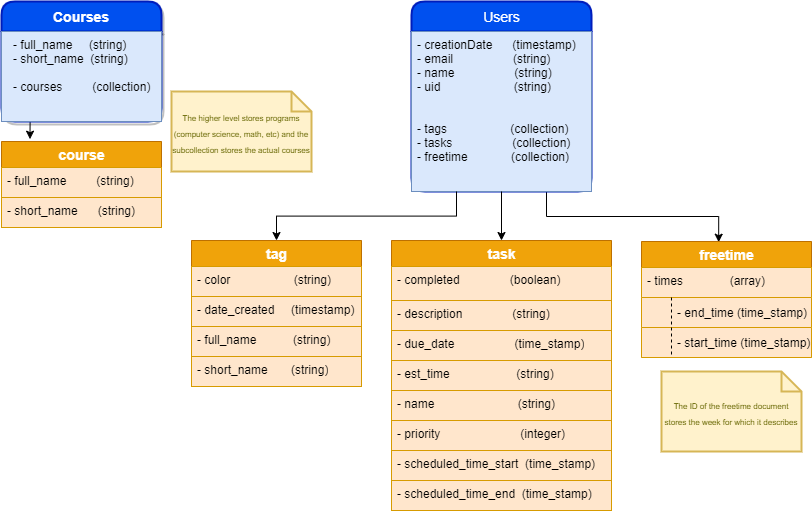
# Appendix B: Analysis Models

**Fig 1. Use Case Diagram**

*Use case diagram for the End User*

**Fig 2. Activity Diagram**

**Fig 3. Database Design**

****

# Appendix C: To Be Determined List

* Automated Schedule Suggestion

1. For more information [see Fig 1. in Appendix B](#_hbpfsz1mqwmy) [↑](#footnote-ref-0)