Homework Tracker SRS

Introduction

Homework Tracker is a software solution that helps students manage homework across the courses that they are taking. It aims to reduce the time and stress that comes from homework management and to simplify the process of prioritizing homework. Since deadlines can be difficult to track and prioritize, Homework Tracker notifies users of upcoming deadlines and allows them to order assignments based on various parameters including grade percentage and estimated completion time.

Current System

N/A

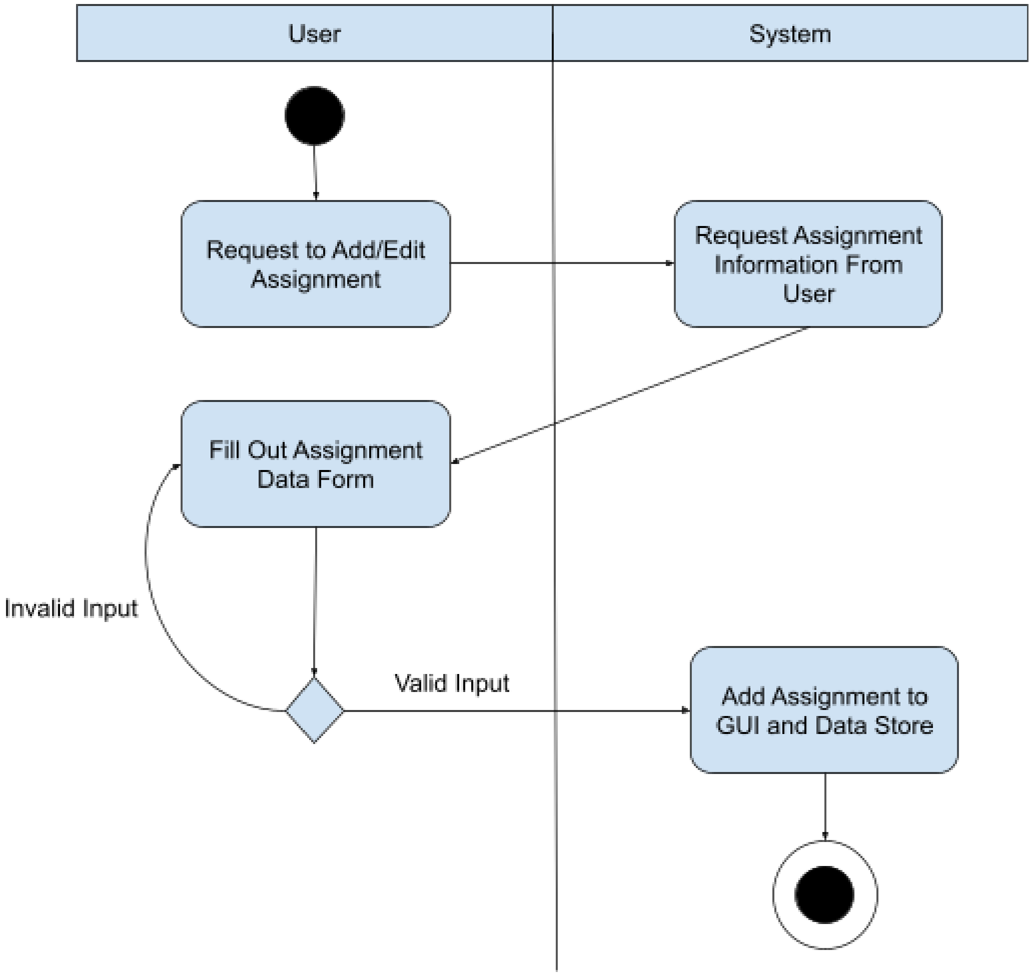
Proposed System

To-be Solution:

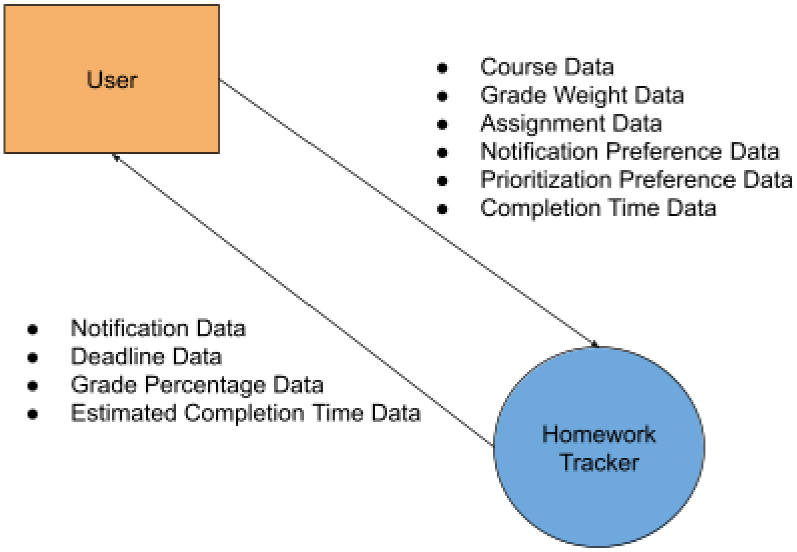
- **Summary:**

  Homework tracker helps the user by getting course and assignment data which it uses to assist the user with homework management. Specifically, homework tracker keeps track of when assignments are due and notifies the user when they have upcoming deadlines. In addition to notifications, homework tracker estimates how long it will take the user to complete an assignment based on previous completion times.To help the user figure out which assignment to focus on, Homework Tracker allows the user to prioritise assignments based on the due date, expected completion time, or grade percentage.

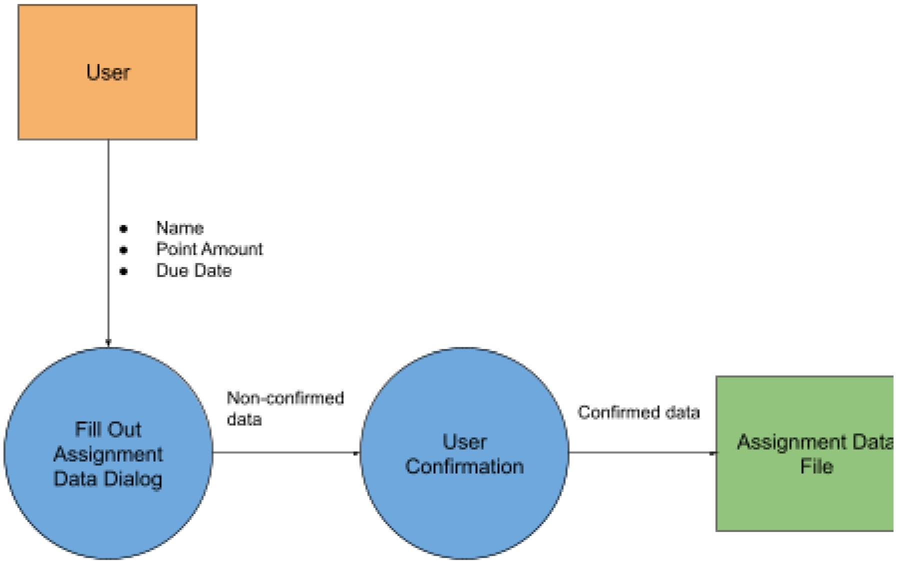
- **Activity diagrams:**
    - Adding/Editing an Assignment:


User
System
Request to Add/Edit Assignment
Request Assignment Information From User
Fill Out Assignment Data Form
Invalid Input
Valid Input
Add Assignment to GUI and Data Store

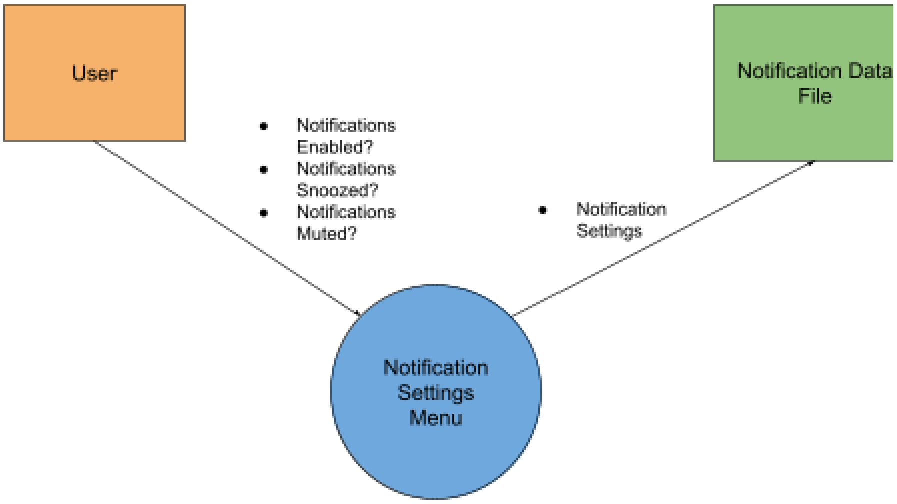
- **DFD[1]s:**
    - Level 0 DFD:



- Level 1 DFD, Assignment Data:



- Level 1 DFD, Notification Data:



Requirements:

Functional Requirements:

- **Use Cases for Courses:**
  - **(1) Add Course (Figure 1)**:
    - Actors:
      - User
    - Entry Condition:
      - User has application open and wants to add a new course.
    - Flow of Events:
      - (1) User selects “Add Course.”
      - (2) User fills out course data form provided by the system.
      - (3) User confirms the information is correct.
    - Exit Condition:
      - System registers new class and displays it to the user.
    - Exceptions:
      - (2a) User enters invalid input, system prompts user to re-enter input
      - (3a) User does not confirm information is correct, system discards information
  - **(2) Set Course Credits**
    - Actors:
      - User
    - Entry Conditions:
      - User has application open and wants to set number of credits for a course.
    - Flow of Events:
      - (1) User selects “Course Credits”
      - (2) User inputs how many credit hours the course is worth
      - (3) User confirms new information for the course
    - Exit Condition:
      - System displays new information to the user
    - Exceptions:
      - (2a) User enters an invalid input, prompts user to re-enter value
      - (3a) User does not confirm new information, system reverts course to previous state
  - **(3) Edit Course**:
    - Actors:
      - User
    - Entry Condition:
      - User has application open and wants to edit a course.

- Flow of Events:
  - (1) User selects a course
  - (2) User selects “Edit Course”
  - (3) System allows user to edit course name
  - (4) User confirms new information is correct for the course
- Exit Condition:
  - System displays the course with the updated information
- Exceptions:
  - (2a) User fails to select a course, system does not allow user to select “Edit Course”.
  - (3a) User enters invalid input, system prompts user to re-enter input
  - (4a) User does not confirm information is correct, system reverts course to pre-editing state.

- **(4) Remove Course**:
  - Actors:
    - User
  - Entry Condition:
    - User has application open and wants to remove a course.
  - Flow of Events:
    - (1) User selects a course
    - (2) User selects “Remove Course”
    - (3) System asks the user for confirmation that it is okay to remove the course and its associated weight categories and assignments
    - (4) User confirms removal.
  - Exit Condition:
    - System remove the course and its associated weight categories and assignments.
  - Exceptions:
    - (2a) User fails to select a course, system does not allow user to select “Remove Course”.
    - (4a) User does not confirm removal, system cancels removal process

- **Use Cases for Grade Weight Categories:**
  - **(5) Add Grade Weight Category**:
    - Actors:
      - User
    - Entry Condition:
      - User has application open and wants to add a new grade weight category.
    - Flow of Events:
      - (1) System gets category name, and weight amount

    - (2) User confirms information is correct for the course to be added
  - Exit Condition:
    - System registers the newly added grade weight category in the course and displays it to the user.
  - Exceptions:
    - (1a) User enters invalid input, system prompts user to re-enter input
    - (2a) User does not confirm information is correct, system discards information
- **(6) Edit Grade Weight Category**:
  - Actors:
    - User
  - Entry Condition:
    - User has application open and wants to edit a grade weight category.
  - Flow of Events:
    - (1) User selects a grade weight category
    - (2) User selects “Edit Grade Weight Category”
    - (3) System allows user to edit grade weight category name, and weight amount
    - (4) User confirms new information is correct for the grade weight category
  - Exit Condition:
    - System displays the grade weight category with the updated information
  - Exceptions:
    - (2a) User fails to select a grade weight category, system does not allow user to select “Edit Grade Weight Category”.
    - (4a)   User enters invalid input, system prompts user to re-enter input
    - (4a) User does not confirm information is correct, system reverts grade weight category to pre-editing state.
- **(7) Remove Grade Weight Category**:
  - Actors:
    - User
  - Entry Condition:
    - User has application open and wants to remove a weight category.
  - Flow of Events:
    - (1) User selects a grade weight category
    - (2) User selects “Edit Grade Weight Category”
    - (3) System asks the user for confirmation that it is okay to remove the grade weight category

        - Exit Condition:
            - System removes the grade weight category
        - Exceptions:
            - (3a) User does not confirm removal, system cancels removal process

- **Use Cases for Assignments:**
    - **(8) Add Assignment (Figure 2)**:
        - Actors:
            - User
        - Entry Condition:
            - User has application open and wants to add a new assignment.
        - Flow of Events:
            - (1) User selects “Add Assignment”
            - (2) System gets assignment name, due date, and point amount from the user
            - (3) User confirms information is correct for the assignment to be added
        - Exit Condition:
            - System displays the newly added assignment in the assignment list
        - Exceptions:
            - (2a) User enters invalid input, system prompts user to re-enter input
            - (3a) User does not confirm information is correct, system discards information
    - **(9) Edit Assignment**:
        - Actors:
            - User
        - Entry Condition:
            - User has application open and wants to edit an assignment.
        - Flow of Events:
            - (1) User selects an assignment
            - (2) User selects “Edit Assignment”
            - (3) System allows user to edit assignment name, due date, and point amount
            - (4) User confirms new information is correct for the assignment
        - Exit Condition:
            - System displays the assignment with the updated information
        - Exceptions:
            - (2a) User fails to select an assignment, system does not allow user to select “Edit Assignment”.

        - (3a)   User enters invalid input, system prompts user to re-enter input
        - (4a) User does not confirm information is correct, system reverts assignment to pre-editing state.
- **(10) Remove Assignment (Figure 3)**:
    - Actors:
        - User
    - Entry Condition:
        - User has application open and wants to remove an assignment.
    - Flow of Events:
        - (1) User selects an assignment
        - (2) User selects “Remove Assignment”
        - (3) System asks the user for confirmation that it is okay to remove the assignment
        - (4) User confirms removal.
    - Exit Condition:
        - System removes the assignment
    - Exceptions:
        - (2a) User fails to select an assignment, system does not allow user to select “Remove Assignment”.
        - (4a) User does not confirm removal, system cancels removal process
- **(11) Change an assignment’s Grade Weight Category**:
    - Actors:
        - User
    - Entry Condition:
        - User has application open and wants to change an assignment’s grade weight category.
    - Flow of Events:
        - (1) User selects a course
        - (2) User selects a grade weight category
        - (3) User selects an assignment
        - (4) User selects “Change Grade Weight Category”
        - (5) User selects another grade weight category
        - (6) System confirms that the user wants to move the assignment to the new grade weight category
    - Exit Condition:
        - System moves the assignment to the new grade weight category
    - Exceptions:
        - (4a) User fails to select a course, system does not allow user to select “Change Grade Weight Category”.
        - (4b) User fails to select a grade weight category, system does not allow user to select “Change Grade Weight Category”.

        - (4c) User fails to select an assignment, system does not allow user to select “Change Grade Weight Category”.
        - (5a) User fails to select another grade weight category, system prompts user to select another grade weight category
        - (6a) User does not confirm change, system reverts the assignment to its original grade weight category
- **(12) Enter assignment start time**
    - Actors:
        - User
    - Entry Condition:
        - User has application open and wants to enter the time that they started working on an assignment
    - Flow of Events:
        - (1) User selects an assignment
        - (1) User selects “Enter Start Time”
        - (3) User enters time they started the assignment
        - (4) System confirms the information with the user
    - Exit Condition:
        - Assignment is updated with when the user started working on it
    - Exceptions:
        - (4a) User does not confirm new settings, system reverts changes
- **(13) Enter assignment end time**
    - Actors:
        - User
    - Entry Condition:
        - User has application open and wants to enter the time that they finished an assignment
    - Flow of Events:
        - (1) User selects an assignment
        - (1) User selects “Enter Completion Time”
        - (3) User enters the time they finished the assignment
        - (4) System confirms the information with the user
    - Exit Condition:
        - Assignment is updated with how long it took to complete
    - Exceptions:
        - (4a) User does not confirm new settings, system reverts changes
- **(14) Check Estimated assignment completion time**
    - Actors:
        - User
    - Entry Condition:
        - User has application open and wants to check the estimated completion time of an assignment
    - Flow of Events:

            - (1) User selects an assignment
        - Exit Condition:
            - Assignment menu displays estimated completion time
        - Exceptions:
            - N/A

- **Use Cases for Notifications**
    - **(15) Mute notifications**
        - Actors:
            - User
        - Entry Condition:
            - User has application open and wants to mute notifications
        - Flow of Events:
            - (1) User selects “notification settings, course settings”
            - (2) User chooses mute option for *course or app*
            - (3) System confirms that the user wants to keep the new settings
        - Exit Condition:
            - System saves and applies changes to notifications settings for *course or app*
        - Exceptions:
            - (3a) User does not confirm new settings, system reverts changes
    - **(16) Enable notifications**
        - Actors:
            - User
        - Entry Condition:
            - User has application open and wants to enable notifications
        - Flow of Events:
            - (1) User selects “notification settings, course settings”
            - (2) User chooses enables notifications for *course or app*
            - (3) System confirms that the user wants to keep the new settings
        - Exit Condition:
            - System saves and applies changes to notifications settings for *course or app*
        - Exceptions:
            - (3a) User does not confirm new settings, system reverts changes
    - **(17) Disable Notifications**
        - Actors:
            - User
        - Entry Condition:
            - User has application open and wants to disable notifications
        - Flow of Events:
            - (1) User selects “notification settings, course settings”
            - (2) User chooses disable notifications option for *course or app*

      - (3) System confirms that the user wants to keep the new settings
    - Exit Condition:
      - System saves and applies changes to notifications settings for *course or app*
    - Exceptions:
      - (3a) User does not confirm new settings, system reverts changes
  - **(18) Snooze Notifications**
    - Actors:
      - User
    - Entry Condition:
      - User indicates to the system they want to snooze notifications
    - Flow of Events:
      - (1) User selects “notification settings, course settings”
      - (2) User chooses snooze option for *course or app*
      - (3) System confirms that the user wants to keep the new settings
    - Exit Condition:
      - System saves and applies changes to notifications settings for *course or app*
    - Exceptions:
      - (3a) User does not confirm new settings, system reverts changes
  - **(19) Change Notification Frequency**
    - Actors:
      - User
    - Entry Condition:
      - User has application open and wants to change notification settings
    - Flow of Events:
      - (1) User selects “notification settings”
      - (2) User selects “notification frequency”
      - (3) User changes notification frequency
      - (4) System confirms changes with the user
    - Exit Condition:
      - System changes notification settings
    - Exceptions:
      - (4a) User does not confirm changes, notification frequency is reverted to its previous state

- **Use Cases for Prioritization:**
  - **(20) Change Prioritization Settings for Assignments**
    - Actors:
      - User
    - Entry Condition:

            - User has application open and wants to change how assignments are prioritized.
        - Flow of Events:
            - (1) User selects “Set Prioritization Settings”
            - (2) User selects the parameter that should be used for assignment prioritization
            - (3) System confirms that the user wants to keep the new settings
        - Exit Condition:
            - System prioritizes assignments based on the new settings
        - Exceptions:
            - (3a) User does not confirm new settings, system reverts changes

- **Other use cases**
    - **(21) Close Program**
        - Actors:
            - User
        - Entry Condition:
            - User has application open and wants to close the program
        - Flow of Events:
            - (1) User selects “Close Program”
            - (2) System prompts for user confirmation to exit program
            - (3) User selects yes to confirm close of the program
        - Exit Condition:
            - System closes the program
        - Exceptions:
            - (3a) User does not confirm exit or backs out of confirmation

    - **(22) Modify GUI[2] Settings**
        - Actors:
            - User
        - Entry Condition:
            - User has application open and wants to modify the GUI settings
        - Flow of Events:
            - (1) User selects “GUI Settings”
            - (2) Application displays all GUI options available to the user
            - (3) User modifies GUI settings
            - (4) System asks the user to confirm changes
        - Exit Condition:
            - System displays courses with new changes to GUI
        - Exceptions:
            - (3a) User does not confirm exit or backs out of confirmation
    - **(23) Change order of classes**
        - Actors:

      - User
    - Entry Condition:
      - User has application open, has added courses, and wants to change the order of classes
    - Flow of Events:
      - (1) User presses and holds a course
      - (2) The course is selected and the user can move the course up or down in the list
      - (3) User releases hold of the course
    - Exit Condition:
      - Course is now saved in the order the user arranged
    - Exceptions:
      - (3a) User clicks or drags outside of the application
  - **(24) Customize Notifications**
    - Actors:
      - User
    - Entry Condition:
      - User has application open and wants to customize notifications
    - Flow of Events:
      - (1) User selects “Notification Settings”
      - (2) Application displays all Notifications options available to the user
      - (3) User modifies GUI settings
      - (4) System prompts user to confirm changes
    - Exit Condition:
      - System saves changes and returns to main view
    - Exceptions:
      - (3a) User does not confirm exit or backs out of confirmation

Non-Functional Requirements:

- Must be a simple and easy to use interface so the user can make modifications quickly and easily.
- Must be running in the background and rarely crash.
- Must not lose data if it crashes.
- Must consume very limited system resources while running in the background.

Constraints:

The application is constrained by the information the user is enters into the program. The application is not capable of getting information about user's classes without them entering it, so if they forget to enter a class, or don't specify crucial information, the application will not be valuable to them.

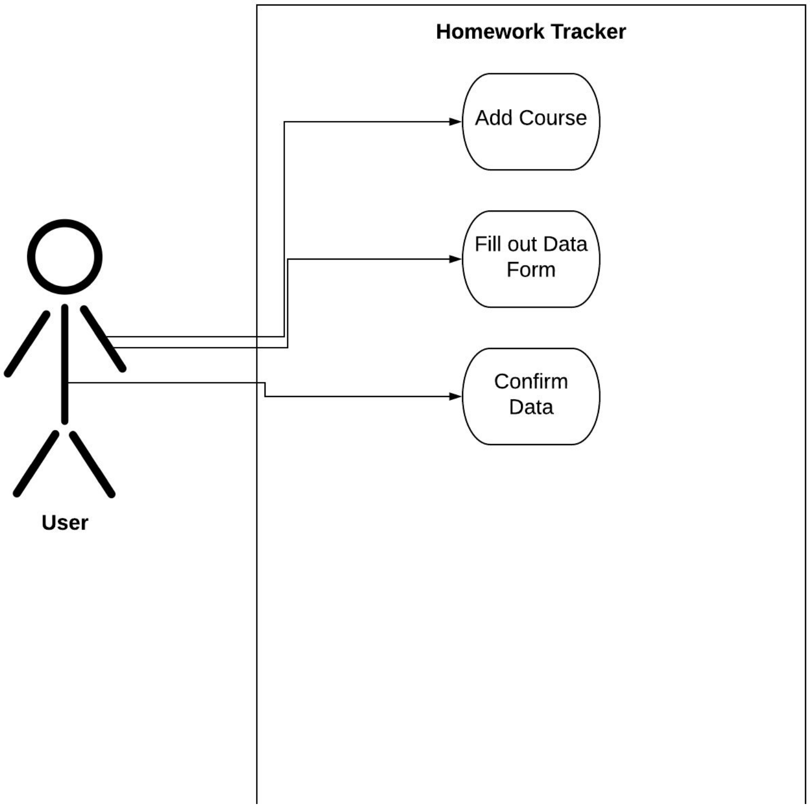
The application is constrained by operating system from which they are running it. Linux and MacOS are not supported.

The application is constrained such that it must be running in the background to send the user notifications.

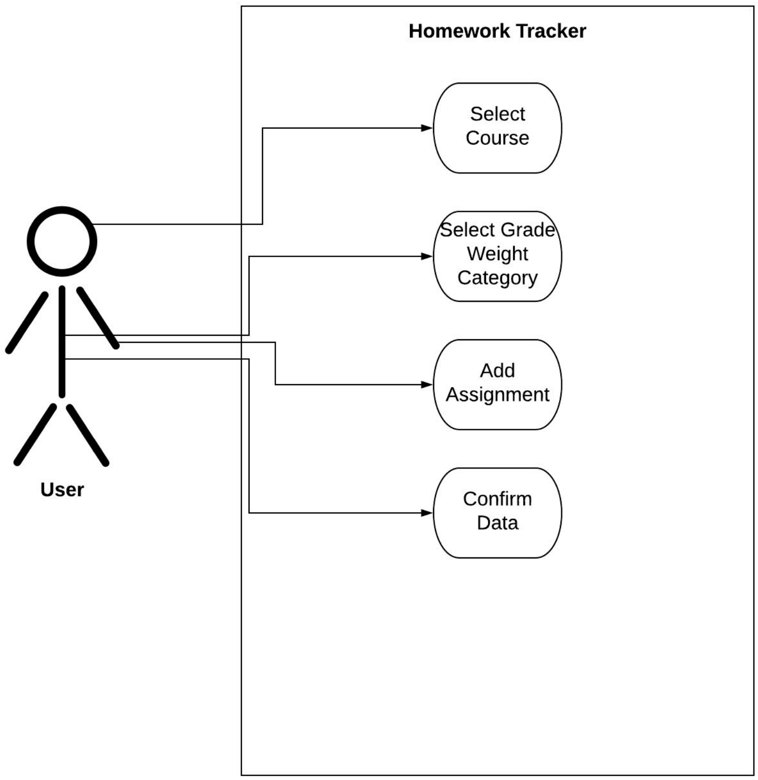
System Models

Use Case Diagrams:

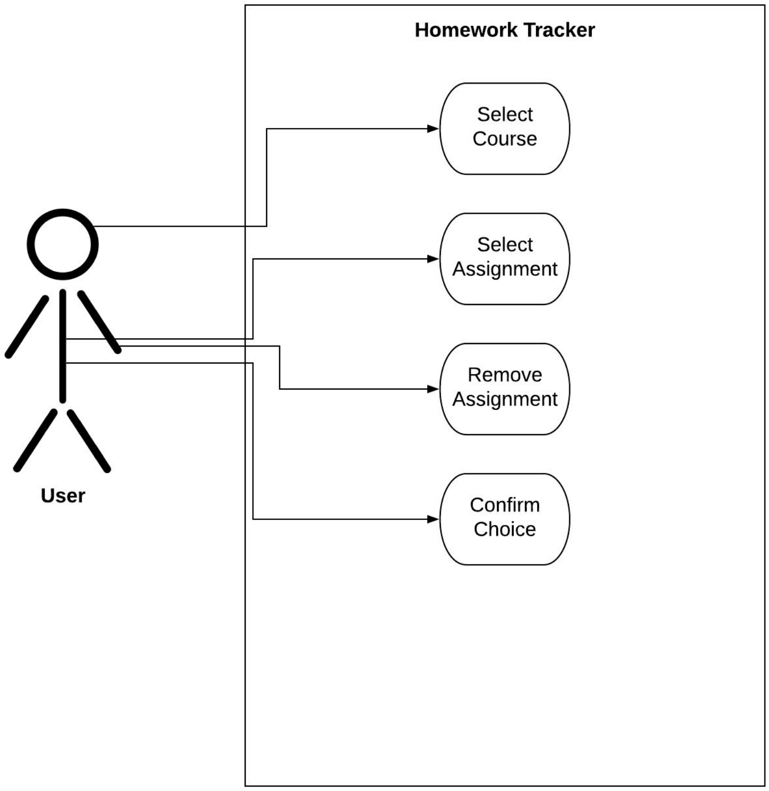
- **Adding a Course (Figure 1):**



- **Adding an Assignment (Figure 2):**



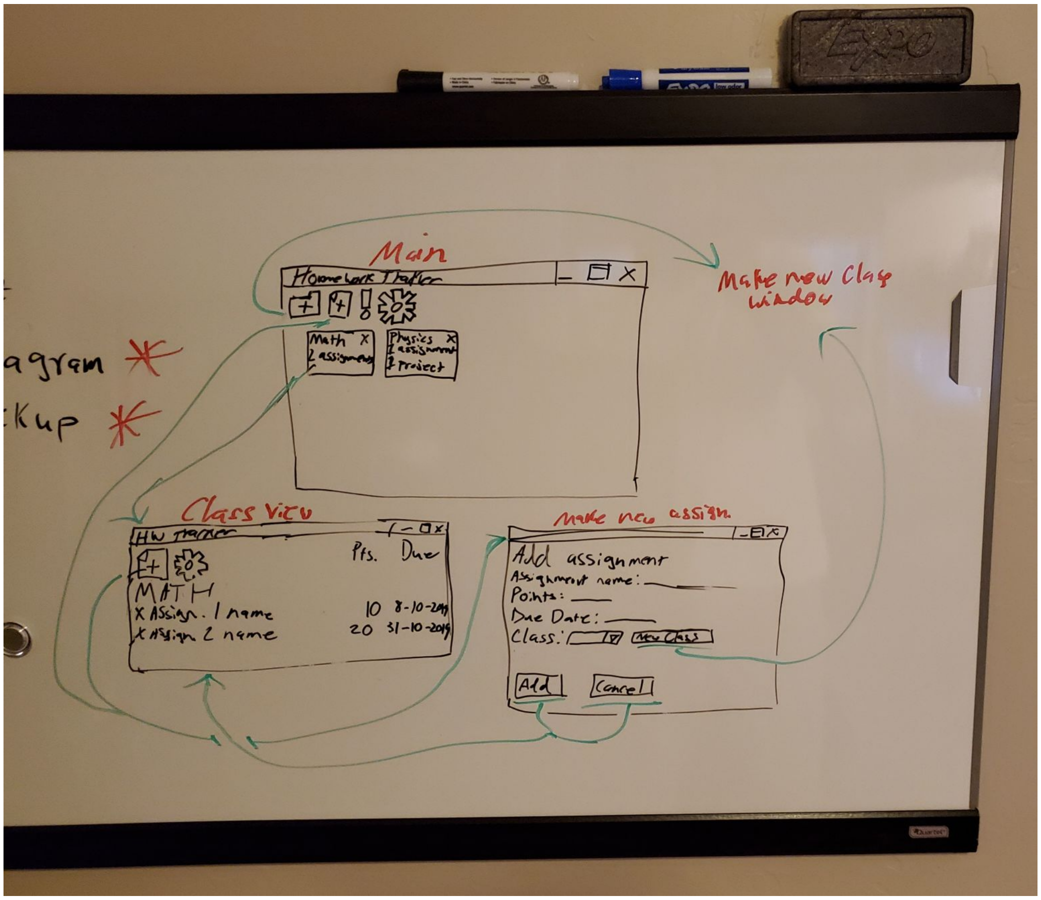
- **Removing an Assignment (Figure 3):**

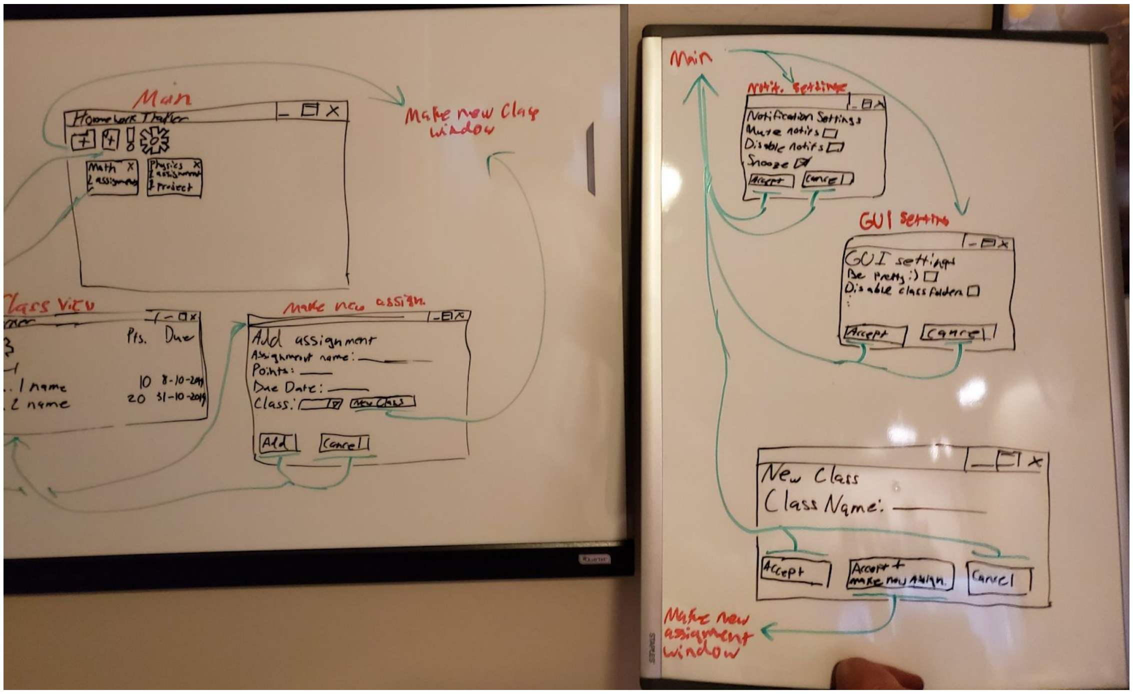


Data Dictionary:

Field Name	Data Type	Data Format	Field Size	Description	Example
Course Name	String	.{1-100}	100	The name of a course the user is taking	CS 4720: Design and Analysis of Algorithms
Course Point Amount	Integer	#{1-20}	20	The total amount of points a course has	1500
Course Credits	Integer	#	1	The amount of credits a course is worth	3
Grade Weight Name	String	.{1-100}	100	The name of a grade weight category in a course	Projects
Grade Weight Amount	Float	0\.#{1-4}	5	How much a weight category contributes to the overall course grade	0.25
Assignment Name	String	.{1-100}	100	The title of an assignment	Project 1: Exploring Greedy Algorithms
Assignment Due Date	Date and Time	MM/DD/YYYY HH:MM {AM | PM}	19	The date an assignment is due	11/01/2019 11:59 PM
Assignment Point Amount	Integer	#{1-10}	10	How many points an assignment is worth	100
Assignment Start Time	Date and Time	MM/DD/YYYY HH:MM {AM | PM}	19	When the user started working on an assignment	10/10/2019 10:35 AM
Assignment Completed Time	Date and Time	MM/DD/YYYY HH:MM {AM | PM}	19	When the user completed an assignment	10/25/2019 11:21 PM
Notifications Muted	Boolean	T | F	1	User has notifications muted?	F
Notifications Enabled	Boolean	T | F	1	User has notifications enabled?	T
Notifications Snoozed	Boolean	T | F	1	User has notifications snoozed?	T
Notification Frequency	Integer	1 | 2 | 3	1	Controls the notification frequency (1 = low, 2 = medium, 3 = high)	2
Prioritization Parameter	Integer	1 | 2 | 3	1	Selects what assignments are sorted by (1 = deadline, 2 = expected completion time, 3 = grade percentage)	3

User Interface Prototypes:





Formulas for Calculations:

- Total Course Points:

$$\sum_{i=1}^{n} \text{assignment}_i * \text{weightOf}(\text{assignment}_i)$$

- Assignment Completion Time:

Assignment End Time - Assignment Start Time

- Estimated Completion Time for new Assignment:

$$\frac{1}{\textit{number of assignments in course}} * \sum_{\textit{assignment} \in \textit{course}} \text{completionTimeOf(assignment)}$$

Glossary/References

Acronym Definitions:

1. DFD: Data Flow Diagram
2. GUI: Graphical User Interface

References:

N/A