# To-Do Planner

Iteration #3

**Date:** 05/02/2022
**Course Name**: SFWR ENG I ANALYSIS, DESIGN, TESTING
**Course Number**: CSE 5324-002

**Professor:** Dr. Michael F. Siok, PE, ESEP

**Group Number**: 7
**Group Members**:


Mane, Paridnya Sanjiv (1001863514)
Kalapala, Yogesh (1001879640)
Parshva Urmish Shah (1001838879)
Patel, Parth Bhanuprasad (1001720900)
Abdulbari Syed (1001995871)

TABLE OF CONTENTS:

# User Requirements: To-Do Planner App

Every individual has some tasks to execute as we know how the daily routine tasks in professionals' life matter and the deadlines must be met accordingly as the individual progresses towards successive tasks of the goals to be achieved. The problem that we are trying to solve with the "To-Do planner app" is that of boosting productivity.

The To-Do planner app is a basic and ordered approach to schedule tasks and organizing tasks which will help users to complete the tasks. Task has a section that describes the category of tasks. Users can create tasks by going into the sections or individually. Each new task has its own name and a small description. The user can set a due time for a task and the app will notify the user based on the deadlines and priorities assigned to the task. The due date and time can be added using the date and time picker. Sometimes a user completes a task before due time in that case the user will be provided with the check box to mark the task as completed. A user can set priority to each task which is shown on the dashboard as Critical, high, medium, and low tasks. Alerts are on a recurring basis until the task is dealt with. they are supposed to keep up with so that tasks would be finished within deadlines.

## Functions of the software:

22. **Create tasks:** Users shall create a task with a title and a small
23. description of the task.
24. **Create sections:** The user shall create a section where the section
25. acts as a collection of related tasks.
26. **Schedule a task:** The user shall add the date and time by which
27. the particular task has to be completed.
28. **Prioritize tasks:** Users shall categorize the priority of tasks as
29. Critical, high, medium, and low.
30. **Simple user interface with ease of use:** The user shall be able to
31. navigate between screens with minimum technicality.
32. **Notification, Alert, and reminders to keep track of schedule:**
33. Users will be alerted with notifications before tasks are due.
34. **Dashboard:** Users will be able to see the list view of the top10 tasks
35. according to the priorities. Additionally, the tasks which are due
36. on the present day or within a week are displayed in grid view.
37. **Filtering Tasks:** The user will be able to filter tasks according to
38. the priorities.
39. **Delete Tasks:** Users can delete as Task.

## Resources to be utilized:

1. Database: to store the data of the user.
2. Wireless internet connection: wireless internet connection is required initially to download the application. The application does not require any internet connection.

## Team Members:

1. **Parshva Shah** - I have learned the basics about Android App Development and Java programming language during my under-graduation. I also have some prior experience in mobile engineering with iOS App Development with programming in Swift Language. I have also worked with IDEs like Android Studio and Xcode to develop mobile apps.
2. **Parth Patel** – I have created basic android applications during my undergraduates using android studio and, I have some intermediate-level knowledge of the java programming language.
3. **Paridnya Mane** - I have participated in a few follow-along coding workshops to create Android apps using Android Studio back during my undergraduate studies. I have worked considerably with Java projects and have a fair syntactic understanding of the language.
4. **Yogesh Kalapala** - I have some basic knowledge about how Android Studio works, but I have never applied my skills to a finished project. I have built some clone apps using ReactNative in my free time. In my previous projects, I used Java and JavaScript.
5. **Abdulbari Syed** - I have developed select features for an iOS mobile application such as blood donor user login, blood bank coverage in the area, and blood donation registration using XCode in Swift programming language. I also developed select features of web application to access crop resource information, price forecast, and data analysis for productivity using Visual Studio Code and Python. Still learning basics and advances in Android studio and its documentation.

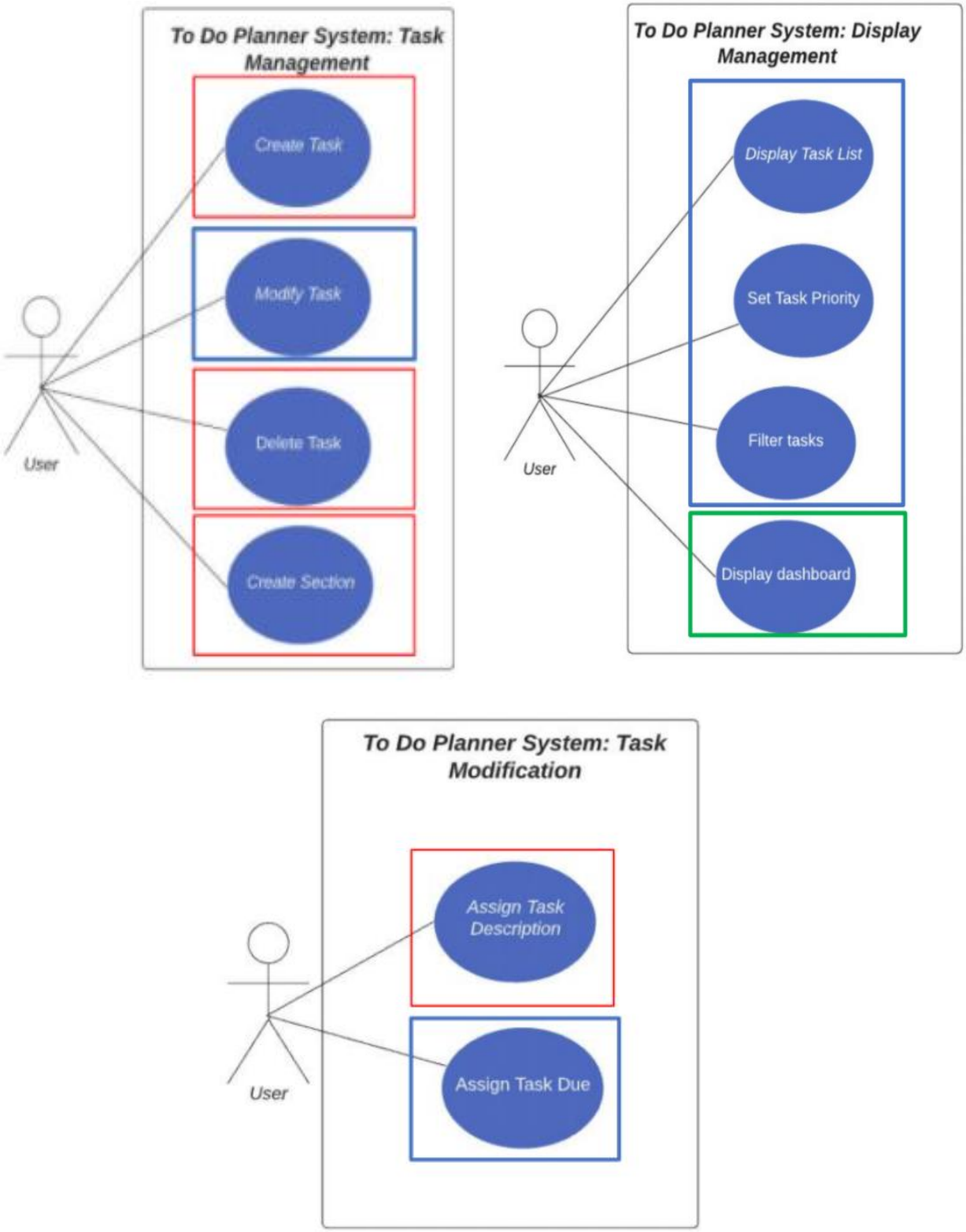
# Requirements:

| Req Id | Req Statement | Line |
|---|---|---|
| R1 | The app shall have section which describe category of task | line 9, line 24-25 |
| R2 | The app shall allow user to create a task | line 10, line 22-23 |
| R2.1 | The app shall allow user to create a task in sections | line 10 |
| R2.2 | The app shall have name assigned to the task | line 11 |
| R2.3 | The app shall have small description of the task | line 11 |
| R3 | The app shall allow user to set due time and date | line 12,line 26-27 |
| R4 | The app shall notify the user | line 13 |
| R4.1 | The app shall notify user based on deadlines | line 13, line 19-21 |
| R4.1.1 | The app shall notify the user 2 hours from the deadline by default | derived |
| R4.2 | The app shall notify user based on priorities | line 13,line 32-33 |
| R4.2.1 | The app shall notify the user before certain amount of time from deadline depending on the priority set by the user | derived |
| R5 | The app shall allow user to change the status of the task | line 16 |
| R5.1 | The app shall provide check box to change the status of task | line 16 |
| R6 | The app shall allow user to set the priority to a task | line 17, line 28 |
| R6.1 | The app shall allow user to set priority as critical or medium or low | line 18, line 29 |
| R6.2 | The app shall allow user to filter tasks based on priorities in dashboard | line 37 - 38 |
| R7 | The app shall display the summary of tasks in list view to user | line 34, line 30-31 |
| R8 | The app shall contain dashboard that displays the tasks which are due in current day | line 34-36 |
| R8.1 | The app shall contain dashboard that displays the tasks which are due in current week | derived |
| R8.2 | The app shall display the summary of tasks according to priorities | line 34,35 |
| R9 | The app shall allow user to delete a task | line 39 |

# Constraints Functionality

| Constraint Id | Constraint Statement | Line |
|---|---|---|
| 1 | The list view of app shall be limited to 10 rows of list. | line 34 |
| 2 | The app shall allow users to assign tasks up to 3 months of due date | derived from R3 |
| 3 | The app shall not keep record of completed tasks. | derived |

## Use Case Diagram:



## High-Level Use Case:

**Use Case1:** Create task
- a. TUCBW the user clicking on "+" symbol
- b. TUCEW the user sees a new task successfully added to the list of tasks

**Use Case2:** Delete Task
- a. TUCBW the user clicks a delete button in task window
- b. TUCEW the user sees a task deleted from the list of tasks

**Use Case3:** Modify Task
- a. TUCBW the users click on the edit button of task window
- b. TUCEW the user changes the required parameter of the task

**Use Case4:** Create Section
- a. TUCBW the user clicking on button
- b. TUCEW the user sees a new section successfully added to the list

**Use Case5:** Assign Task Description
- a. TUCBW the user select text box beside Task Description label
- b. TUCEW the user successfully assigns a description to task

**Use Case6:** Assign Task Due
- a. TUCBW the user select date picker beside Due Date label
- b. TUCEW the user successfully assigns a deadline to a task

**Use Case7:** Set Task Priority
- a. TUCBW the user clicks priority from the drop-down menu
- b. TUCEW the priority is set to the particular task

**Use Case8:** Display Task List
- a. TUCBW the user clicks on button of the section list in application
- b. TUCEW is the application displaying a list view of tasks.

**Use Case9:** Display Dashboard
- a. TUCBW the user clicks on the dashboard icon
- b. TUCEW the application displays summary of tasks and tasks by due date are displayed in grid view.

**Use Case10:** Filter Tasks
- a. TUCBW the user clicks on the filter menu
- b. TUCEW application displays the list view of tasks based on a selected category of priorities.

# Requirements User Case Traceability Matrix

| | Priority weight | UC-1 | UC-2 | UC-3 | UC-4 | UC-5 | UC-6 | UC-7 | UC-8 | UC-9 | UC-10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **R1** | 1 | | | | X | | | | | | |
| **R2** | 1 | X | | | | | | | | | |
| **R2.1** | 2 | X | | | | | | | | | |
| **R2.2** | 2 | | | | X | | | | | | |
| **R2.3** | 2 | | | | | X | | | | | |
| **R3** | 2 | | | | | | X | | | | |
| **R4** | 1 | | | | | | X | | | | |
| **R4.1** | 2 | | | | | | X | | | | |
| **R4.1.1** | 2 | | | | | | X | | | | |
| **R4.2** | 2 | | | | | | X | | | | |
| **R4.2.1** | 2 | | | | | | X | | | | |
| **R5** | 4 | | | X | | | | | | | |
| **R5.1** | 4 | | | X | | | | | | | |
| **R6** | 3 | | | | | | | X | | | |
| **R6.1** | 3 | | | | | | | X | | | |
| **R7** | 1 | | | | | | | | X | | |
| **R8** | 2 | | | | | | | | X | | |
| **R8.1** | 2 | | | | | | | | | X | |
| **R8.2** | 2 | | | | | | | | | | X |
| **R9** | 4 | | X | | | | | | | | |
| | SCORE | 3 | 4 | 8 | 3 | 2 | 11 | 6 | 3 | 2 | 2 |

Priority weights are 1 to 5. 1 being highest and 5 being the lowest

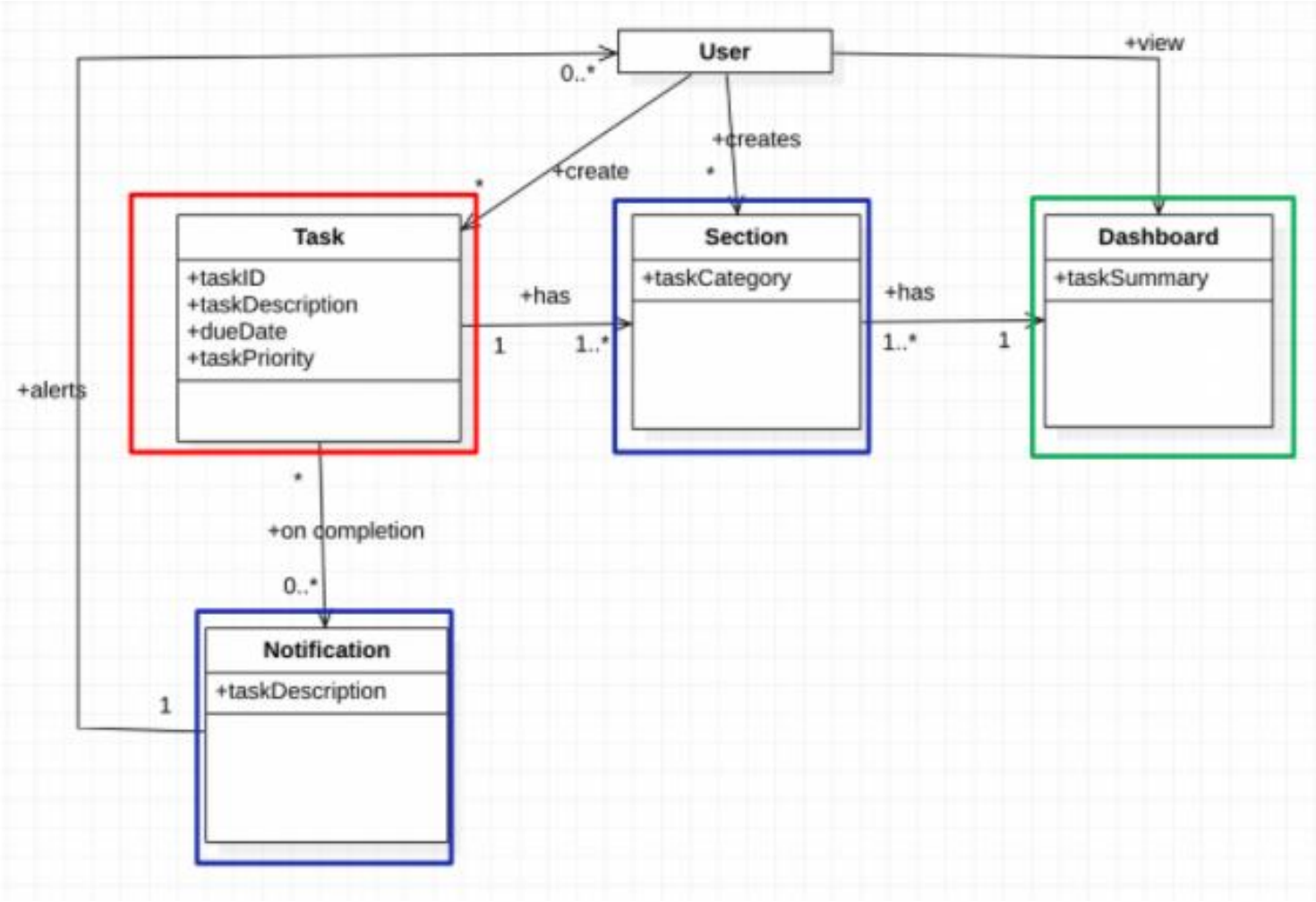
# Increment Matrix

| Use Case | Priority | Efforts(in terms of person) | Depends on | Assigned to | Iteration 1 (3/11/2022) | Iteration 2 (04/08/2022) | Iteration 3 (05/02/2022) |
|---|---|---|---|---|---|---|---|
| UC-1 | 3 | 1 | None | PS | | 1 | |
| UC-2 | 4 | 1 | UC-1 | PS | | 1 | |
| UC-3 | 8 | 3 | UC-1 | AS | | 2 | 1 |
| UC-4 | 3 | 2 | UC-1 | PP | | 2 | |
| UC-5 | 2 | 1 | UC-1 | PM | | 1 | |
| UC-6 | 11 | 4 | UC-1 | PP | | 2 | 2 |
| UC-7 | 6 | 4 | UC-1 | PM | | 1 | 3 |
| UC-8 | 3 | 2 | UC- 1-8 | YK | | 1 | 1 |
| UC-9 | 2 | 4 | UC-1,6,8 | PM | | | 4 |
| UC-10 | 2 | 4 | UC-1,8 | AS | | 1 | 3 |
| Total Efforts | | 26 | | | | 12 | 14 |

PS = Parshva Shah, YK = Yogesh Kalapala, PP = Parth Patel, AS = Abdulbari Syed, PM = Paridnya Mane

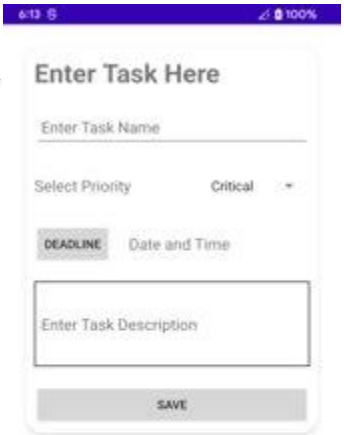
1 person week= 5 hours

## Domain Model

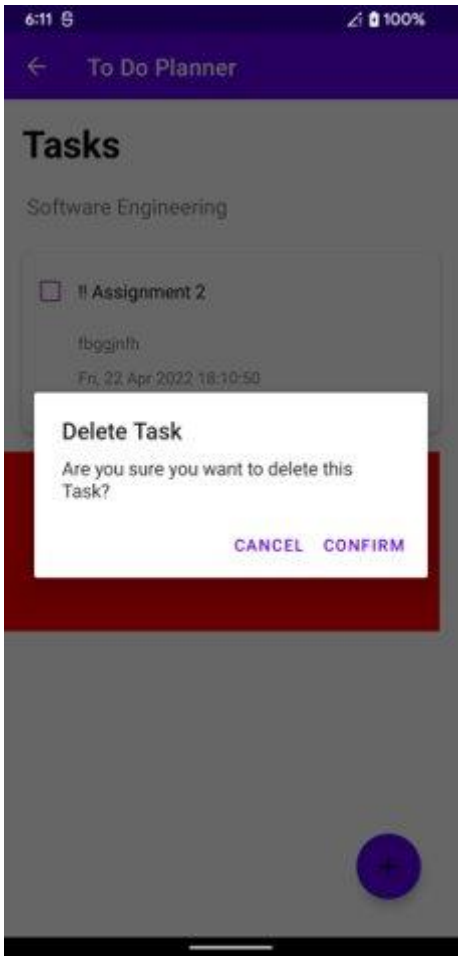


# Expanded Use Case Diagrams & User Interface Prototype


| **EUC 1: Create Task** | |
|---|---|
| **Pre Condition:** This use case assumes that the user is in section part of application. | |
| Actor: User | System: To Do Planner |
| | 0. App displays section screen. |
| 1. **TUCBW** the user clicking on "+" symbol. | 2. App displays Add task screen. |
| 3. The user enters details and clicks save button. | 4.* App creates the task. |
| 5. **TUCEW** the user sees a new task successfully added to the list of tasks. | |
| **Postcondition:** The created task is added to the task list. | |

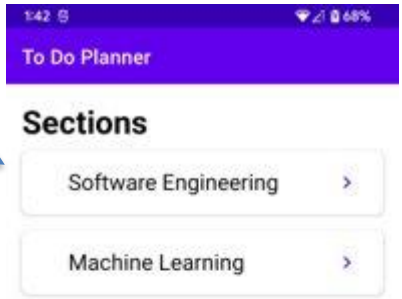


| EUC 2: Delete Task | |
|---|---|
| **Pre Condition:** This use case assumes that the user has created a task. | |
| Actor: User | System: To Do Planner |
| | 0. App displays list of tasks on screen. |
| 1. **TUCBW** the user clicks a delete button in task window. | 2. App displays Delete task screen. |
| 3. The user clicks confirm button. | 4.* App deletes the task. |
| 5. **TUCEW** the user sees task deleted from the list of tasks. | |
| **Postcondition:** The task is deleted from the task list. | |

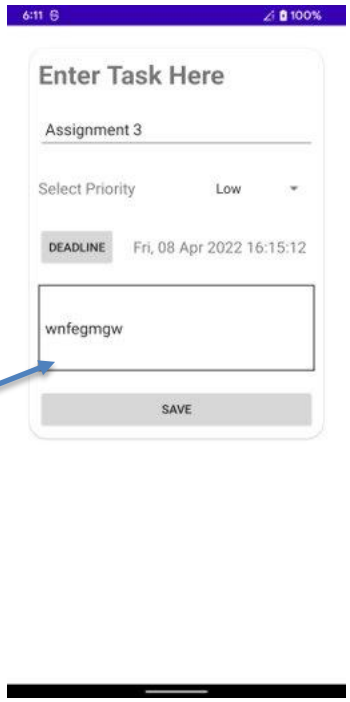



| EUC 4: Create Section | |
|---|---|
| **Pre Condition:** This use case assumes that the user is in launch screen. | |
| Actor: User | System: To Do Planner |
| | 0. App displays launch screen. |
| 1. **TUCBW** the user clicking on button. | 2. App displays Add Section screen. |
| 3. The user enters section name and clicks create button. | 4.* App Creates Section. |
| 5. **TUCEW** the user successfully assigns a name to task. | |
| **Postcondition:** The App creates a new section in list. | |

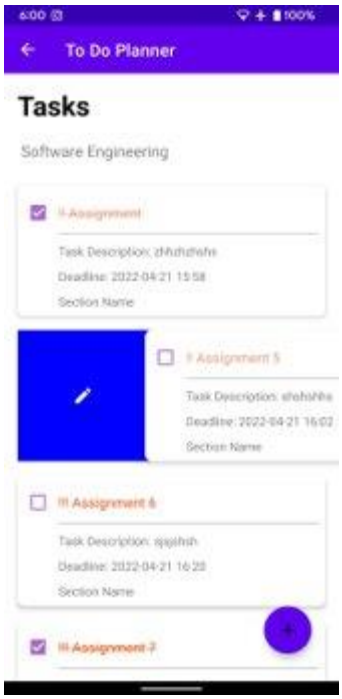





| EUC 5: Assign Task Description(trivial) | |
|---|---|
| **Pre Condition:** This use case assumes that the user is in create task window. | |
| Actor: User | System: To Do Planner |
| | 0. App displays Add task screen. |
| 1. **TUCBW** the user select text box beside Task Description label. | 2. App displays task description parameters window. |
| 3. The user enters task description and clicks create button. | 4. App adds text in the task description. |
| 5. **TUCEW** the user successfully assigns a description to task. | |
| **Postcondition:** The description is added to task. | |



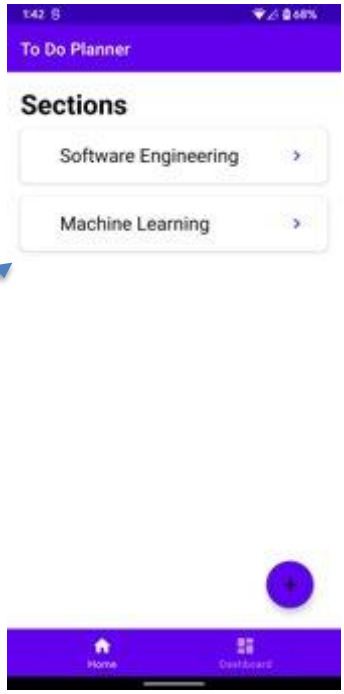
| EUC 3: Modify Task | |
|---|---|
| **Pre-Condition:** This use case assumes that the user has created a task and user is in task window. | |
| Actor: User | System: To Do Planner |
| | 0. App displays list of tasks on screen. |
| 1. **TUCBW** the users click on the edit button of task window. | 2. App displays edit task screen. |
| 3. The user updates desired details and clicks create button. | 4.* App updates the parameters of task. |
| 5. **TUCEW** the user changes the required parameter of the task. | |
| **Postcondition:** The task with changed parameters is updates to task list. | |



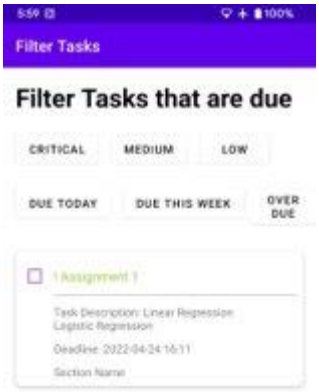

| EUC 6: Assign Task Due(trivial) | |
|---|---|
| **Pre Condition:** This use case assumes that the user is in create task window. | |
| Actor: User | System: To Do Planner |
| | 0. App displays Add task screen. |
| 1. **TUCBW** the user select date picker beside Due Date label. | 2. App displays task parameters window. |
| 3. The sets a date to be reminded and clicks create button. | 4. App assigns user selected date for the task |
| 5. **TUCEW** the user successfully assigns a deadline to task. | |
| **Postcondition:** The due date is assigned to the task. | |


| EUC 7: Set Task Priority(trivial) | |
|---|---|
| **Pre-Condition:** This use case assumes that the user is in create task window. | |
| Actor: User | System: To Do Planner |
| | 0. App displays Add task screen. |
| 1.**TUCBW** the user clicks priority from the drop-down menu . | 2. App displays task parameters window. |
| 3. The user selects priority as either critical, medium, low. | 4. App allocate priority to the task |
| 5. **TUCEW** the priority is set to the particular task. | |
| **Postcondition:** The priority for task is assigned successfully. | |

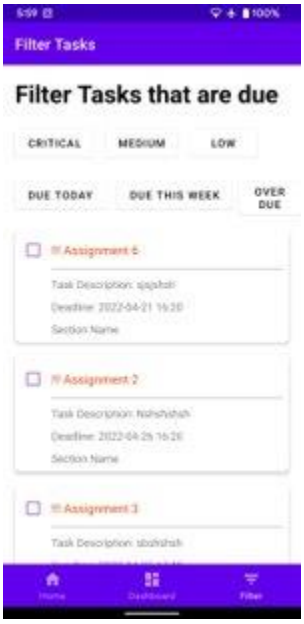
| EUC 8: Display Task List | |
|---|---|
| **Pre-Condition:** This use case assumes that the user has created one or more tasks. | |
| Actor: User | System: To Do Planner |
| | 0. App displays launch screen. |
| 1. **TUCBW** the user clicks on button of the section list in application. | 2. The app calls the required list from database and displays tasks in list view |
| 3. **TUCEW** the application displaying a list view of tasks. | |
| | |
| **Postcondition:** All available tasks are displayed in list view. | |




| EUC 9: Display Dashboard | |
|---|---|
| **Pre Condition:** This use case assumes that the user has created one or more tasks. | |
| Actor: User | System: To Do Planner |
| | 0. App displays launch screen. |
| 1. **TUCBW** the user clicks on the dashboard icon. | 2.* App displays summary of tasks fetching the data from database. |
| 3. **TUCEW** the application displays summary of tasks and tasks by due date are displayed in grid view. | |
| | |
| **Postcondition:** The app displays summary of tasks and tasks by due date are displayed in grid view. | |



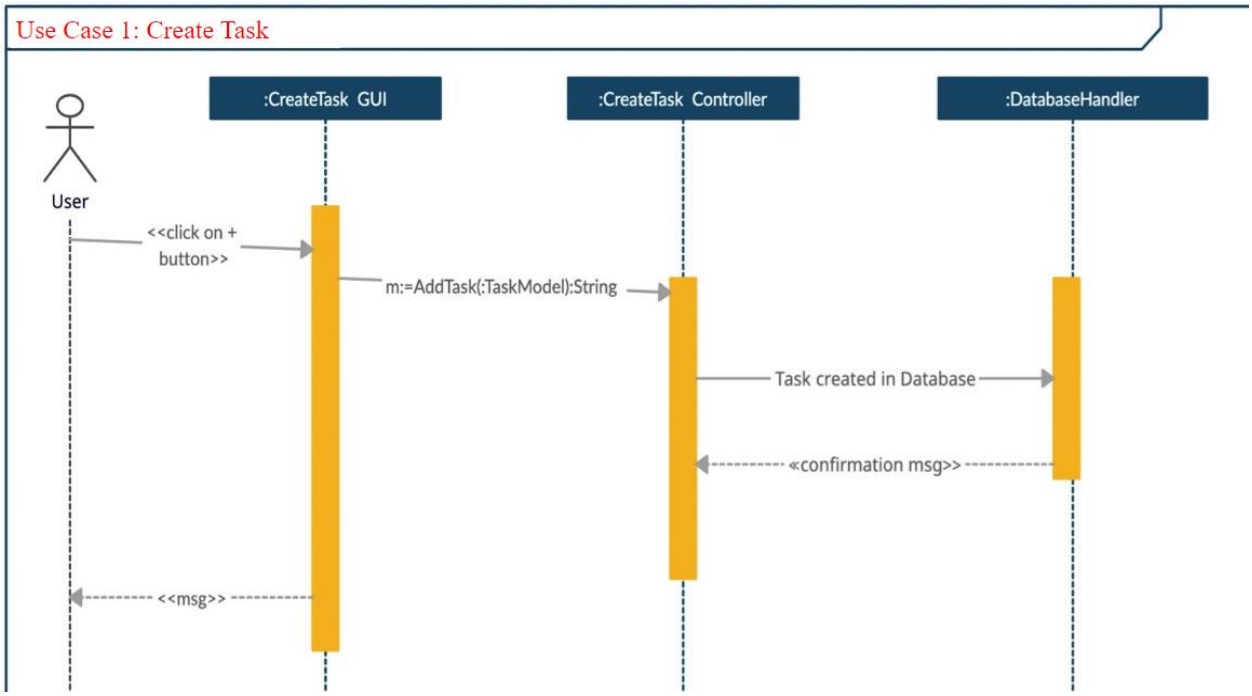

| EUC 10: Filter Tasks | |
|---|---|
| **Pre Condition:** This use case assumes that the user has created one or more tasks. | |
| Actor: User | System: To Do Planner |
| | 0. App displays launch screen. |
| 1. **TUCBW** the user clicks on the filter menu. | 2. App displays filter condition window. |
| 3. The user selects desired sorting condition. | 4.* App displays list of tasks according to the filtering condition. |
| 5. **TUCEW** application displays the list view of tasks based on a selected category of priorities. | |
| **Postcondition:** The created task is added to the task list. | |



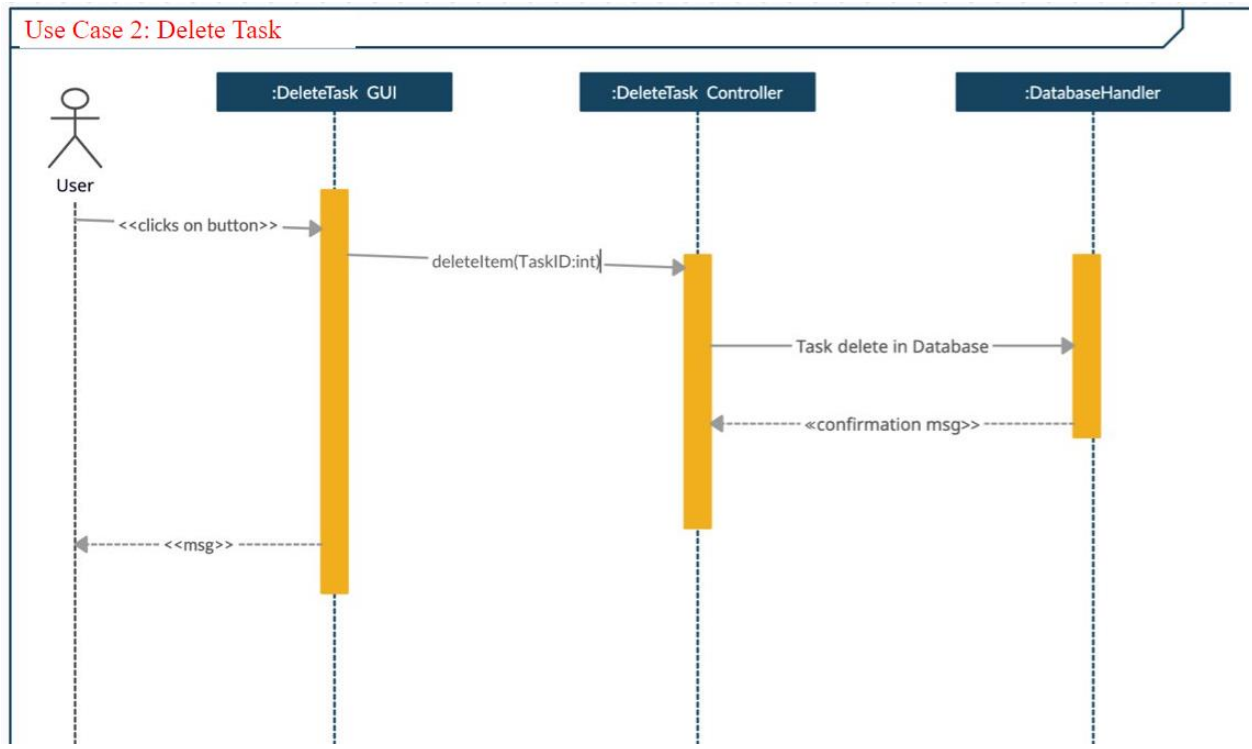
# Sequence Diagrams

The Expanded Use Case Diagrams which are non-trivial has sequence diagrams.

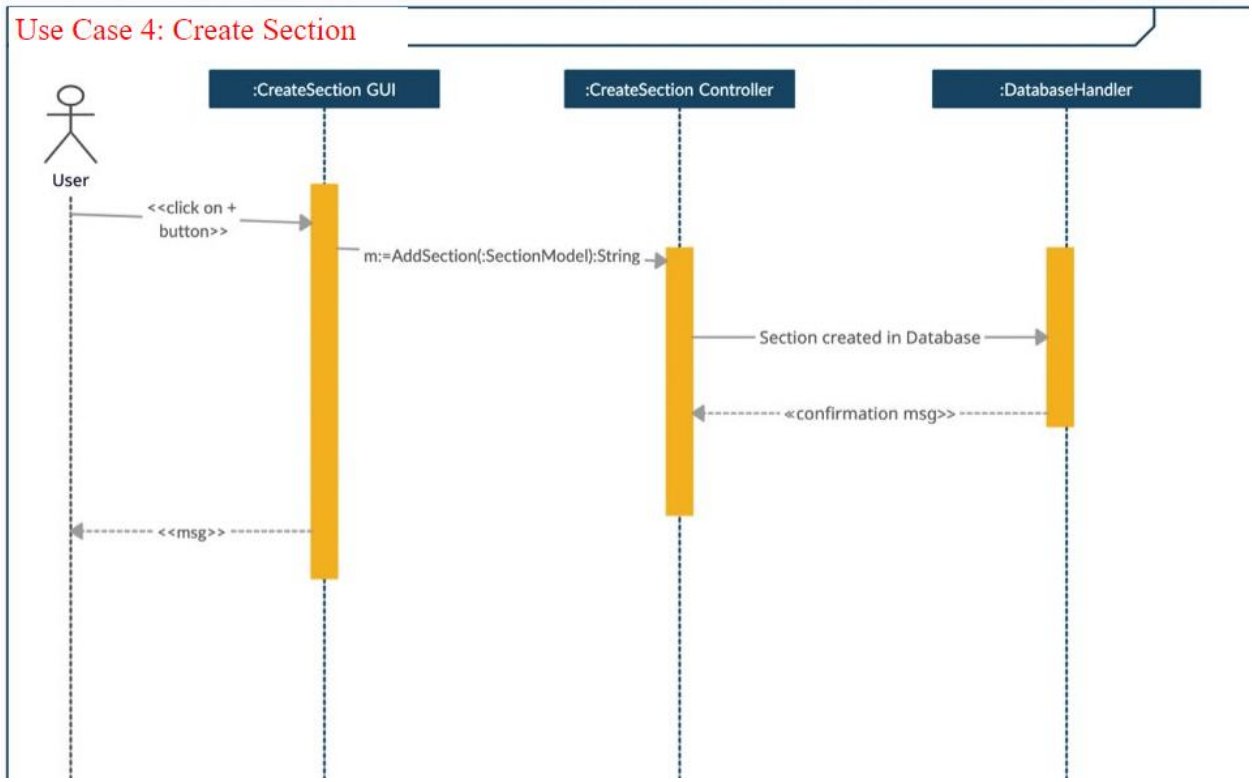
EUC-1 Sequence Diagram



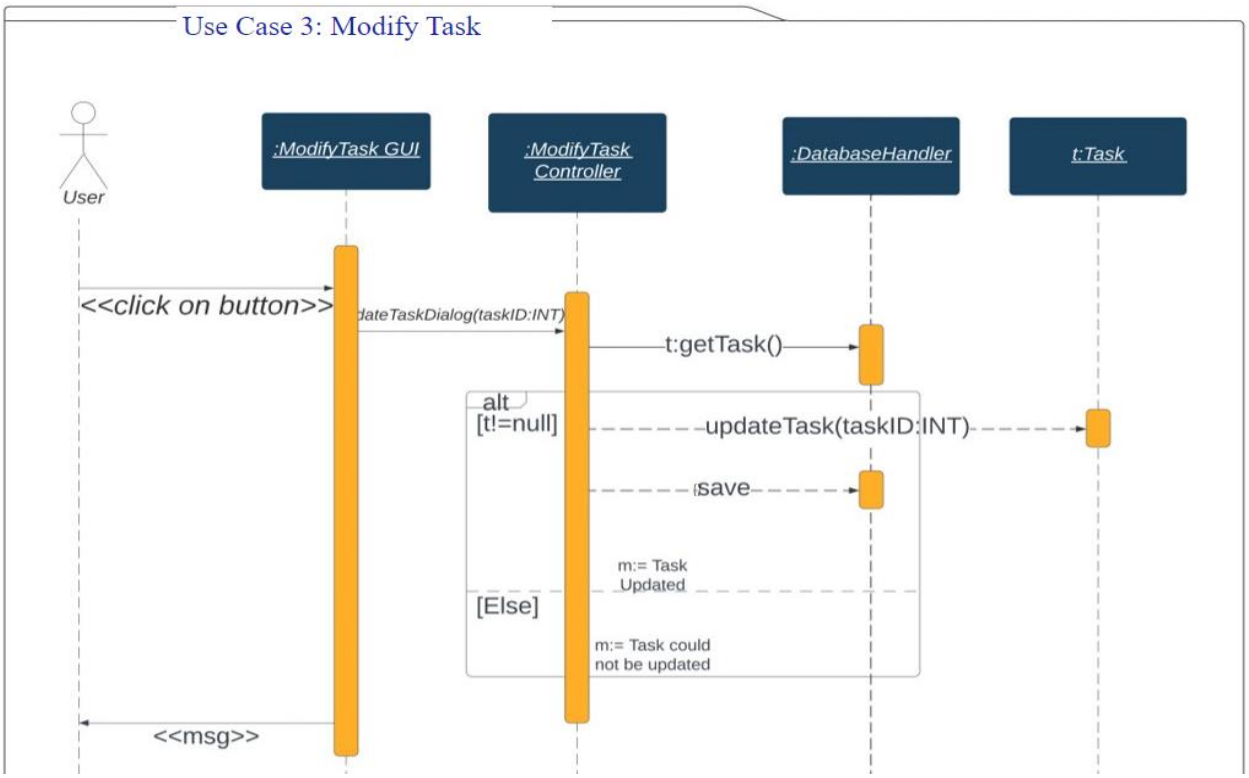

EUC-2 Sequence Diagram



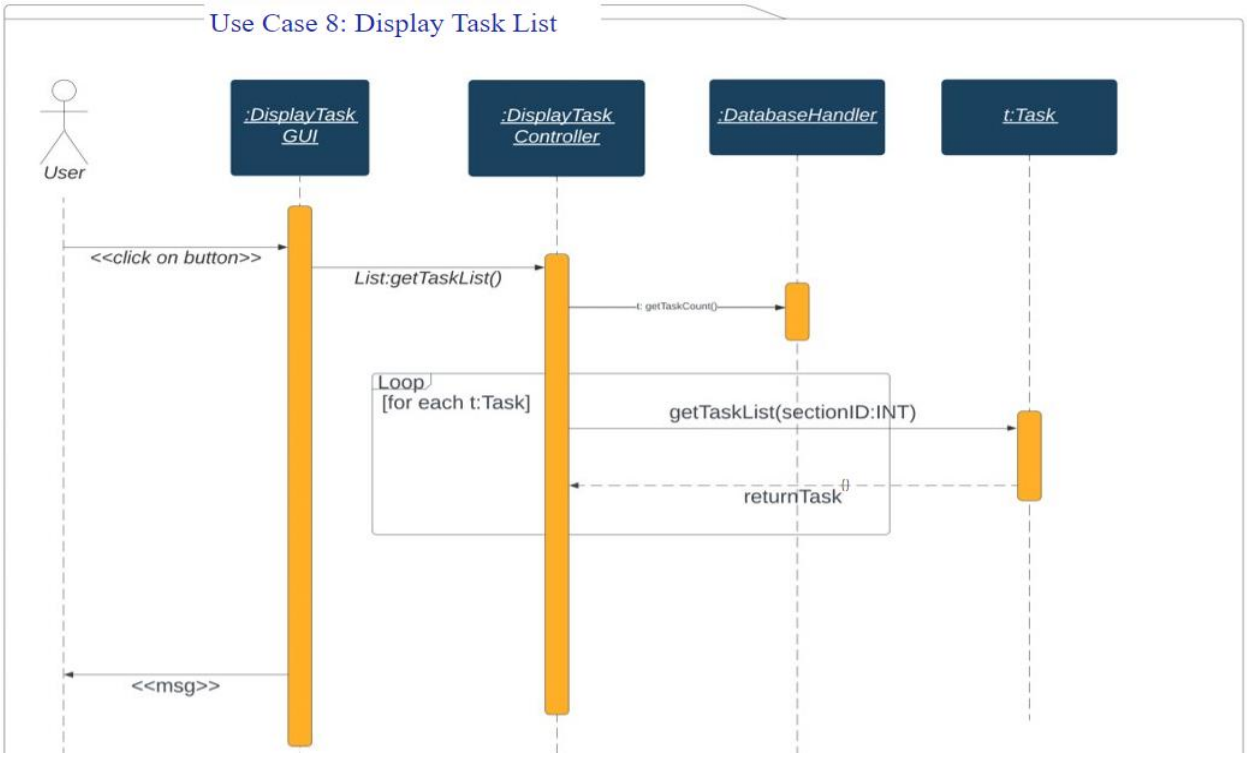
EUC-4 Sequence Diagram



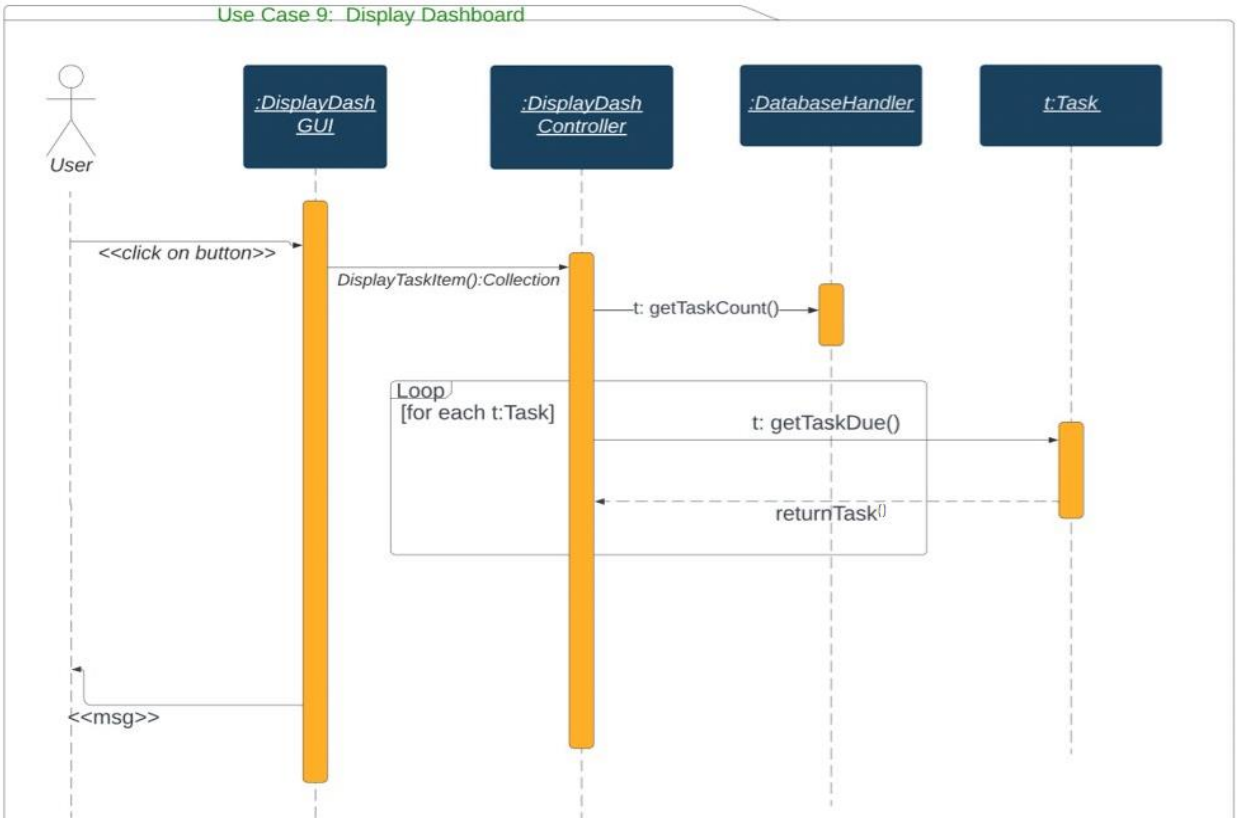

EUC-3 Sequence Diagram



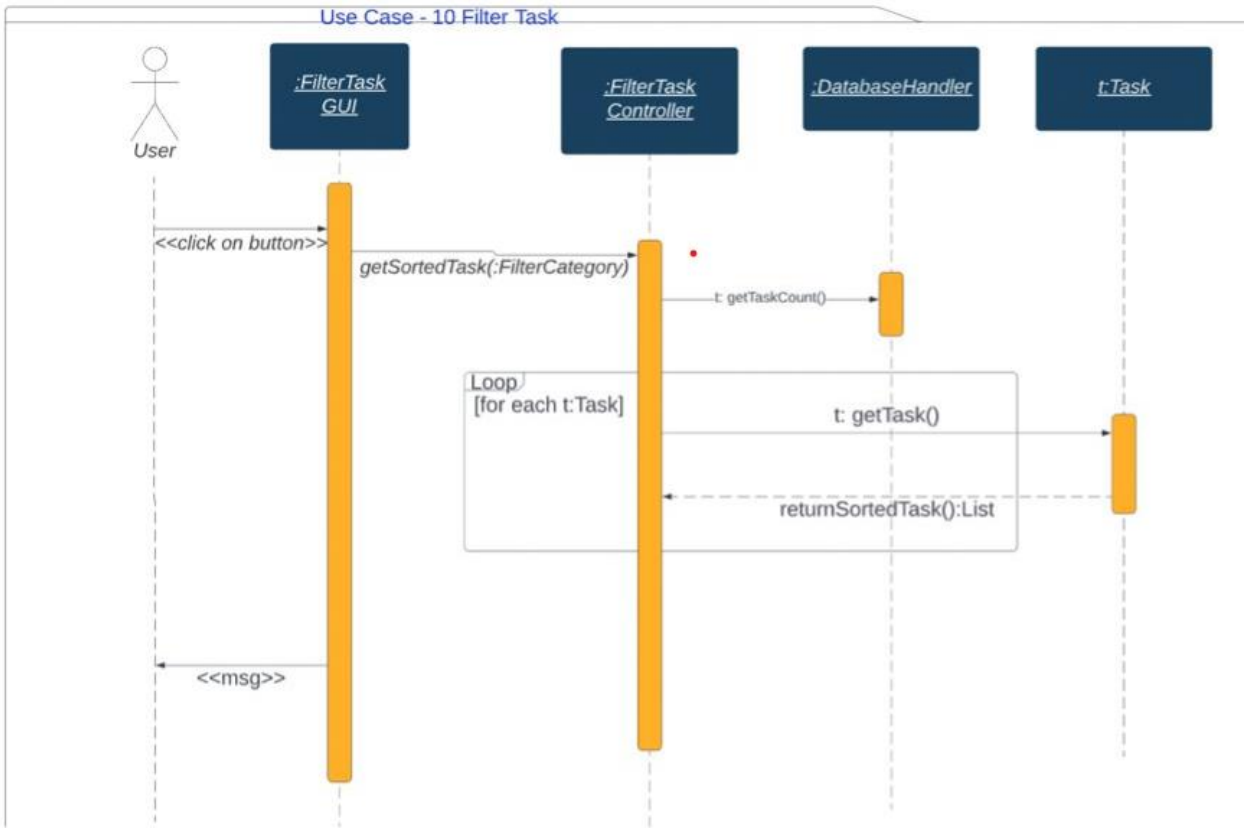
EUC-8 Sequence Diagram



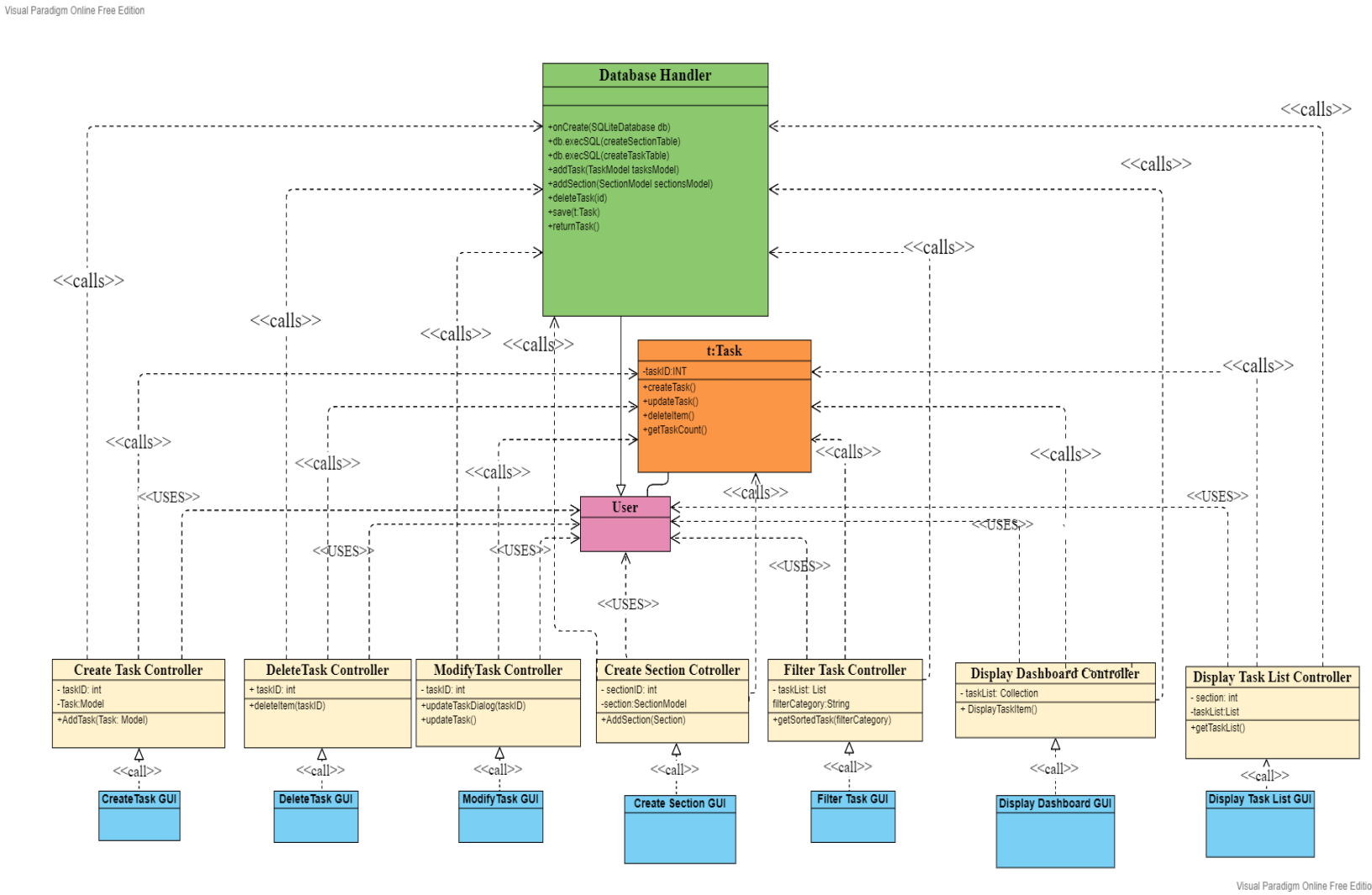

EUC-9 Sequence Diagram



EUC-10 Sequence Diagram



# Design Class Diagram



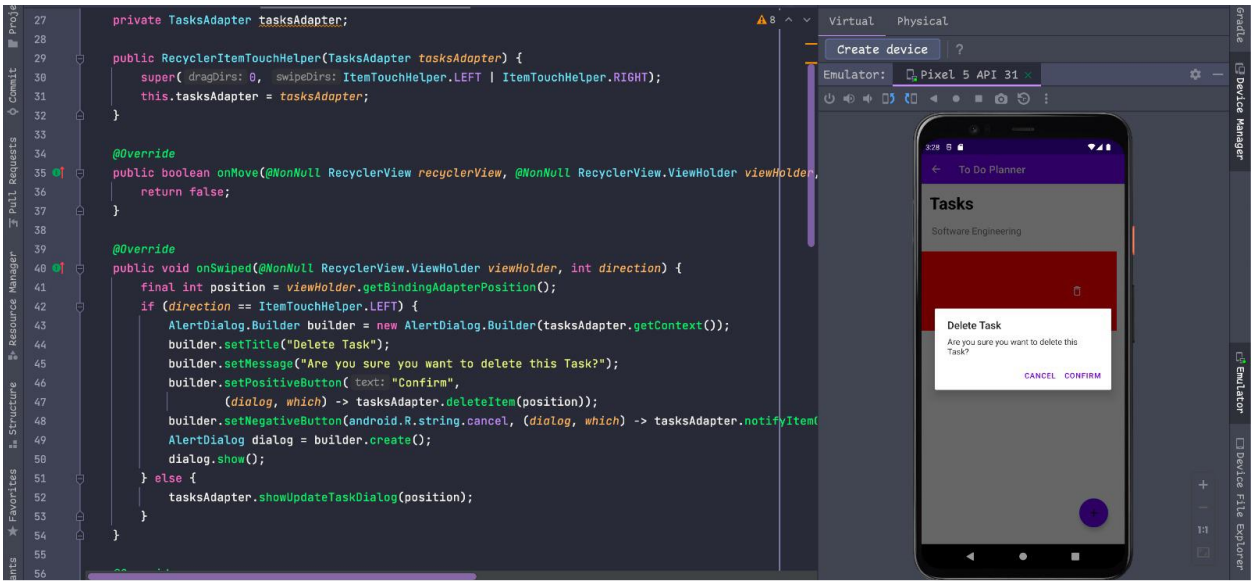
# Code Snippets

```
public void addTask(View view) {
    try {
        tasksModel = new TasksModel( id: -1, secID,  status: 0, spinner.getSelectedItem().toString(), enterTaskName.getText().to
        Toast.makeText( context: this, tasksModel.toString(), Toast.LENGTH_SHORT).show();
    } catch (Exception ignored) {
        Toast.makeText( context: this,  text: "Error creating" + ignored, Toast.LENGTH_SHORT).show();
    }

    DatabaseHandler databaseHandler = new DatabaseHandler( context: this);
    boolean success = databaseHandler.addTasks(tasksModel);

    Toast.makeText( context: this,  text: "Task Added" + success, Toast.LENGTH_SHORT).show();
}
}
```

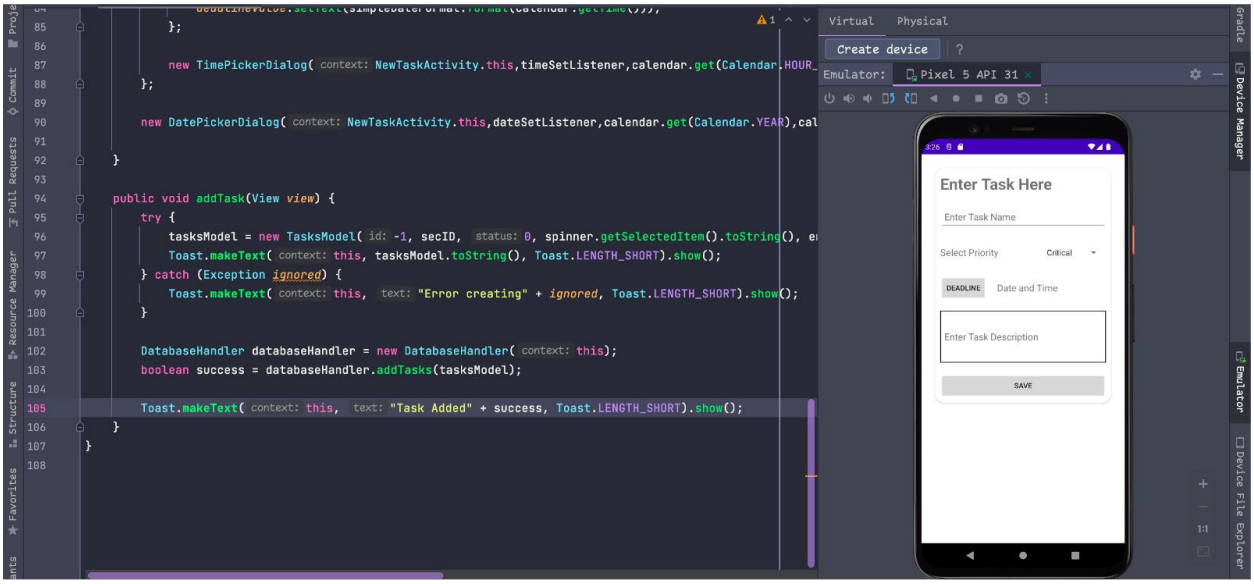
## Use Case 1: Create Task

```
public void addSection(View view) {
    try {
        sectionModel = new SectionModel( id: -1, et_sectionName.getText().toString());
        if (et_sectionName.getText().toString().isEmpty()) {
            Toast.makeText( context: this,  text: "Section Name cant be empty", Toast.LENGTH_SHORT).show();
        }
    } catch (Exception e) {
        Toast.makeText( context: AddNewSectionActivity.this,  text: "Error creating", Toast.LENGTH_SHORT).show();
    }

    DatabaseHandler databaseHandler = new DatabaseHandler( context: this);
    boolean success = databaseHandler.addSection(sectionModel);

    Toast.makeText( context: this,  text: "Section Added" + success, Toast.LENGTH_SHORT).show();
}
}
```

## Use Case 4: Create Section




# Use Case 2: Delete Task




# Use Case 5: Add Task Description

## YouTube Video Link

**Key Points.**

1. In this Iteration all Use Cases have been implemented which were mentioned in Incremental Matrix.
2. Expanded Use Case, Sequence Diagrams and UI Prototype are update for the use cases that were mentioned in Incremental Matrix.
3. Use Cases implemented were UC-3, UC-8, UC-9, UC-10.
4. To depict the progress of Iteration#2 and Iteration#3 the text and diagrams in Requirements, High Level Use Cases, Use Case Diagram and Domain Model have been highlighted. Where:
    a. **Red:** Use Case implemented in Iteration#2.
    b. **Blue:** Use Case was partially implemented in Iteration#2 and completed by Iteration#3.
    c. **Green:** Use Case was completely implemented in Iteration#3.
5. All Feedback and supporting documents provided have been reconsidered and all the suggestions have been updated for final submission.