Department of Computer Science and Engineering, University of Dhaka.

Project Report

Object Oriented Programming (CSE-2112)

Project Name: Pragmata: To-Do List App

Course Code - (CSE-2122)

Team Name:

Md. Emon Khan-30

Saadman Moyeed-33

Mahmudul Hasan-60

Submitted to:

Dr. Muhammad Ibrahim

Assistant Professor, Dept. of CSE, DU

Md. Ashraful Islam

Assistant Professor, Dept. of CSE, DU

Introduction:

In our daily lives, we often find ourselves struggling to remember important tasks that need to be completed. However, taking brief notes on these tasks can greatly enhance our productivity and help us make the most of our time. This is where ***Pragmata***, a handy to-do list app, comes in. With Pragmata, users can jot down the necessary tasks and even assign them a specific deadline for completion.

Purpose and Intentions

By utilizing this digital tool, users can ensure that they never forget important tasks or appointments. Pragmata acts as a reliable companion in their daily routine, providing reminders and keeping them on track. With its intuitive interface and customizable features, Pragmata is an effective solution for anyone seeking to streamline their workflow and increase efficiency.
The purpose of creating a to-do list app is to provide individuals with a digital tool that helps them effectively manage their tasks, stay organized, and improve their productivity. The intention behind developing such an app is to simplify the process of task management and provide users with a centralized platform where they can easily capture, track, and *prioritize their to-do items. The primary goal is to enable users to have a clear overview of their tasks and deadlines, allowing them to plan and allocate their time efficiently. The app seeks to alleviate the mental burden of trying to remember all tasks and ensure that nothing falls through the cracks. By having a comprehensive and easily accessible to-do list, users can reduce stress and gain peace of mind, knowing that their tasks are properly organized and managed. Ultimately, the intention is to empower individuals to be more productive, accomplish their goals, and strike a balance between work and personal life. The to-do list app serves as a valuable tool in helping users prioritize their tasks, maintain focus, and achieve a sense of fulfillment by completing their to-do items.

Platform, Library & Tools

- **Platform**: Java is a general-purpose programming language that is widely used for developing various applications, including desktop, web, and mobile applications.
- **Library**: JavaFX is a Java library used for building this application and it provides a modern, lightweight alternative to Swing. JavaFX provides a set of UI controls, graphics, multimedia, and animation capabilities for creating visually appealing applications.
- **GitHub**: GitHub is used to control and collaborate, primarily used for hosting and managing Git repositories. It allows us to store, share, and collaborate on code projects, making it easier to track changes, manage branches, and work on code together with others.
- **VSCode**: Visual Studio Code (VSCode) is a popular source code editor developed by Microsoft. While it supports multiple programming languages, including Java, it provides a lightweight and customizable environment for editing and debugging code. It offers features such as syntax highlighting, code completion, debugging support, and integration with version control systems like Git.

Features

The App provides several useful features such as:

1. Task Creation: Users should be able to create new tasks by providing a task description, and due date.
2. Task Editing: Users should be able to modify existing tasks by updating their descriptions, due date, and other attributes.
3. Task Deletion: Users should have the ability to delete tasks that are no longer relevant or necessary.
4. Task Completion: The application should provide a mechanism for users to mark tasks as completed, indicating that they have been finished or accomplished.
5. Task Sorting: Users should be able to sort tasks based on various criteria, such as due date, priority, or alphabetical order, to easily prioritize and manage their to-do list. (**Not Implemented yet)
6. Task Reminders: The application can offer the option to set reminders for tasks, allowing users to receive notifications or alerts at specific times or intervals. (**Not Implemented yet)
7. Task Notes or Attachments: The ability to add notes, comments, or attachments in the description box which can be useful for providing additional details or reference material.
8. Data Backup/Restore: It is essential to provide a mechanism to back up and restore task data to prevent data loss.

Class Description

App

Attributes:

- primaryStage: Stage: This attribute represents the primary stage or window of the application. It is of type "Stage," which is typically used to manage the application's graphical user interface (GUI) and handle window-related operations.
- ViewTask viewTask: This attribute represents creating a class name viewTask.
- Task: This attribute represents a specific task within the application. It is of type "Task," indicating that it holds information or data related to a task that the application deals with.

Methods:

- start(): This method is likely an overridden method from the JavaFX Application class. It serves as an entry point for the application and is responsible for initializing and configuring the primary stage and launching the application's user interface.
- main(String[] args): This method is the main entry point for the application. It is the starting point of execution and allows the application to be launched from the command line. It typically sets up any necessary configurations and calls the "launch()" method from the JavaFX Application class to start the application

ViewTask

Attributes:

- TaskList taskList: This attribute represents a task list, indicating that the taskList is associated with a collection of tasks.
- AddTaskButton addTaskButton: This attribute represents a button class used for adding tasks. It is of type "AddButton" class and is associated with a functionality to add new tasks to the task list.
- RemoveButton removeTaskButton: This attribute represents a button class used for removing tasks.
- EditButton editTaskButton: This attribute represents a button class used for editing tasks. It is of type "EditButton" associated with functionality to modify existing tasks in the task list.
- MarkAsCompleteButton markAsCompleteButton: This attribute represents a button class used to mark tasks as complete. It is of type "MarkAsCompleteButton" and is associated with functionality to indicate that a task has been finished or completed.
- ListView<String> taskListView: This attribute represents a list view component that displays the tasks. It is of type "ListView<String>" and is likely used to present the tasks in a user-friendly format.
- TextField taskDescField: This attribute represents a text field where the user can input or edit the description of a task. It is of type "TextField" and is used to capture or display the task descriptions.
- DatePicker taskDatePicker: This attribute represents a date-picker component. It allows the user to select or modify the due date of a task. It is of type "DatePicker" and is associated with managing task deadlines or due dates.

Methods:

- addTaskList(): This method is used to add a new task list or initialize the task list in some way.
- deleteTaskList(): This method is used to delete or remove the task list.
- edit task (): This method is used to edit or modify a specific task within the task list.
- markTaskAsComplete(): This method is used to mark a task as complete, indicating that it has been finished or accomplished.
- saveTask(): This method is used to save or update the changes made to a task.
- warn(String title, String message): This method is likely used to display warning messages or notifications to the user. It takes a title and a message as parameters to customize the content of the warning.

Task

Attributes

- String description: This attribute holds the description or title of the task. It provides a textual representation of what needs to be done.

- LocalDate dueDate: This attribute represents the due date of the task. It stores a date value indicating when the task should be completed.
- boolean completed: This attribute tracks whether the task has been completed or not. It is a boolean value that is set to true if the task is marked as completed and false otherwise.

Constructor
- Task(String description, LocalDate dueDate, boolean completed): This constructor is used to create a new Task object. It takes parameters for the task description, due date, and the completed status. These parameters are used to initialize the corresponding attributes of the Task object.

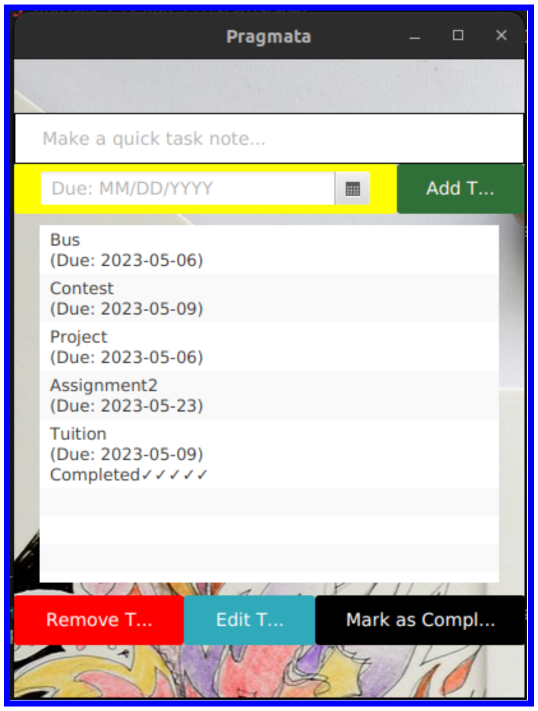
Methods
- String getDescription(): This method returns the description of the task. It allows other parts of the code to retrieve and access the task's description.
- LocalDate getdueDate(): This method returns the due date of the task. It provides access to the task's due date to other parts of the code.
- String toString(): This method overrides the default toString() method to provide a string representation of the Task object. It returns a string that includes the task's description, due date, and completion status.
- String toString2(): This method is a separate implementation of the toString() method. It provide a different formatting or representation of the Task object, depending on the specific needs of the application

TaskList

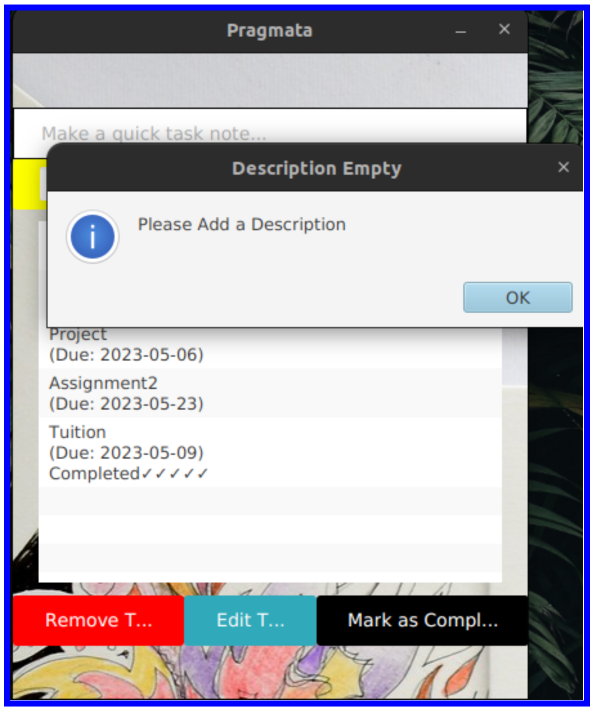
Attributes:
- List<Task> tasks: This attribute represents the list of tasks within the task list. It is of type "List<Task>", indicating that it is a collection that holds objects of the "Task" class. The "List" interface provides methods for adding, removing, and accessing tasks in the list.

Methods:
- addTask(Task task): This method is used to add a new task to the task list. It takes a "Task" object as a parameter and adds it to the "tasks" list.
- removeTask(int index): This method is used to remove a task from the task list based on its index. It takes an index parameter that specifies the position of the task to be removed from the "tasks" list.
- editTask(int index, Task task): This method is used to edit or modify an existing task within the task list. It takes an index parameter that specifies the position of the task to be edited and a "Task" object containing the updated task details.

User Interface

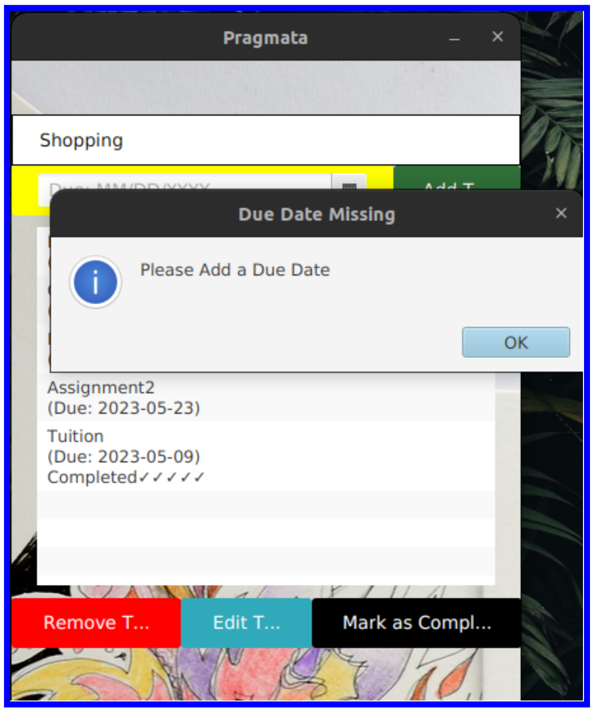


Bus
(Due: 2023-05-06)
Contest
(Due: 2023-05-09)
Project
(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓



Please Add a Description

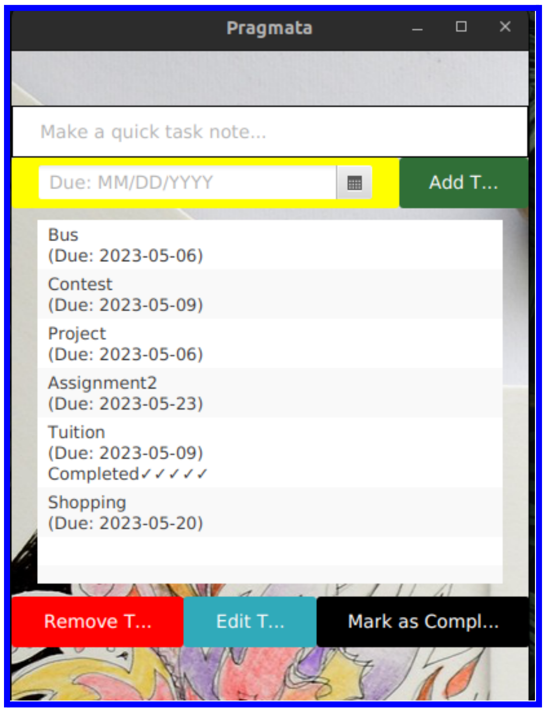
Project
(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓



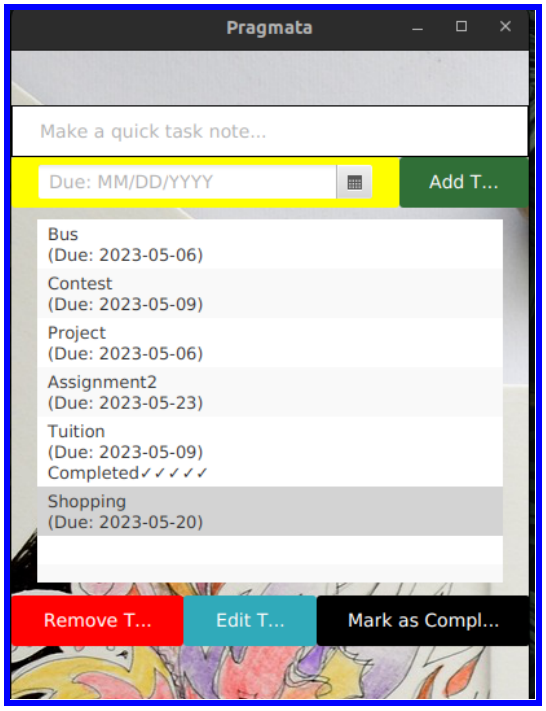
Shopping

Please Add a Due Date

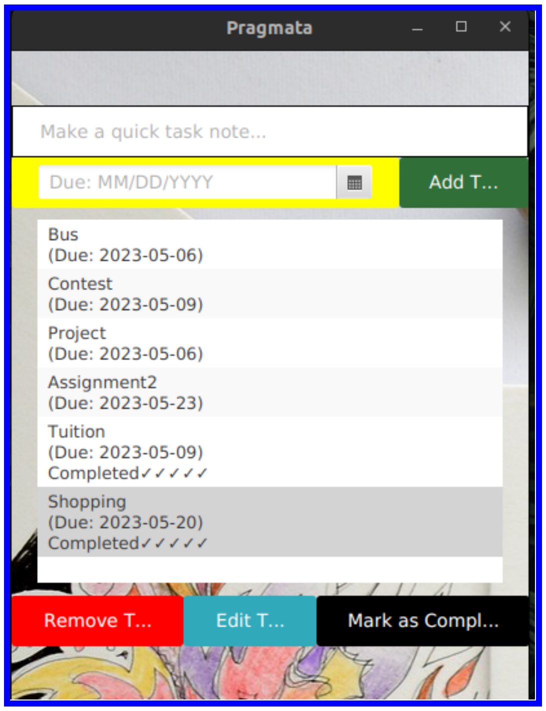
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓



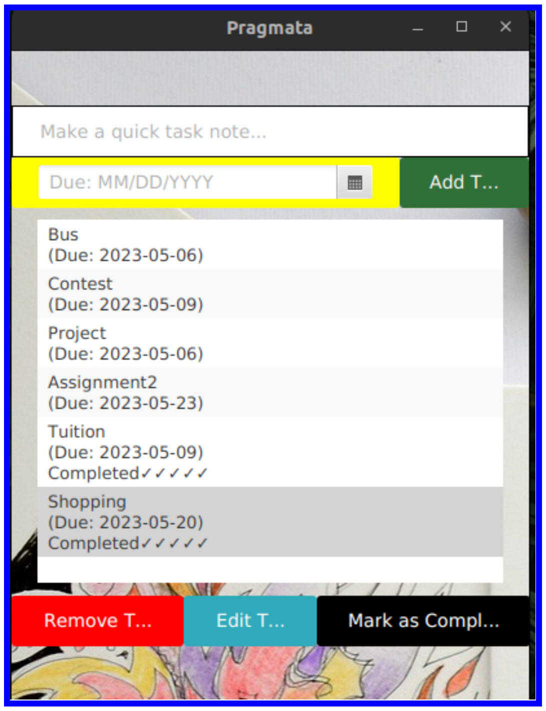
Bus
(Due: 2023-05-06)
Contest
(Due: 2023-05-09)
Project
(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓
Shopping
(Due: 2023-05-20)



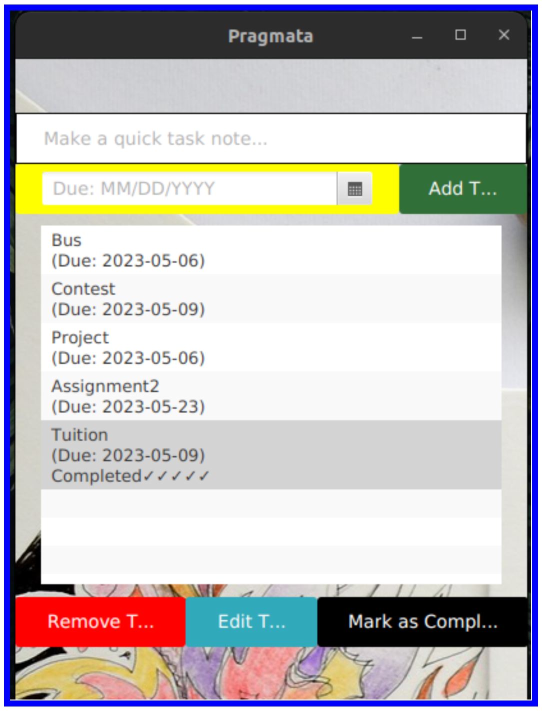
Bus
(Due: 2023-05-06)
Contest
(Due: 2023-05-09)
Project
(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓
Shopping
(Due: 2023-05-20)



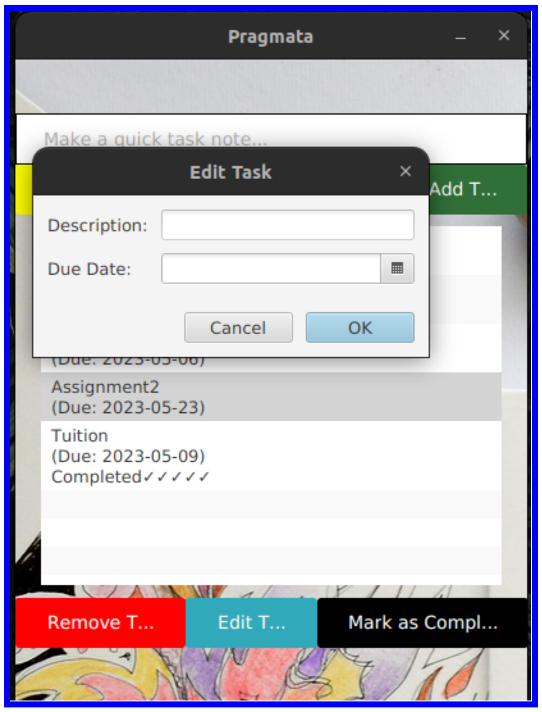
Bus
(Due: 2023-05-06)
Contest
(Due: 2023-05-09)
Project
(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓
Shopping
(Due: 2023-05-20)
Completed✓✓✓✓✓



Bus
(Due: 2023-05-06)
Contest
(Due: 2023-05-09)
Project
(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓
Shopping
(Due: 2023-05-20)
Completed✓✓✓✓✓

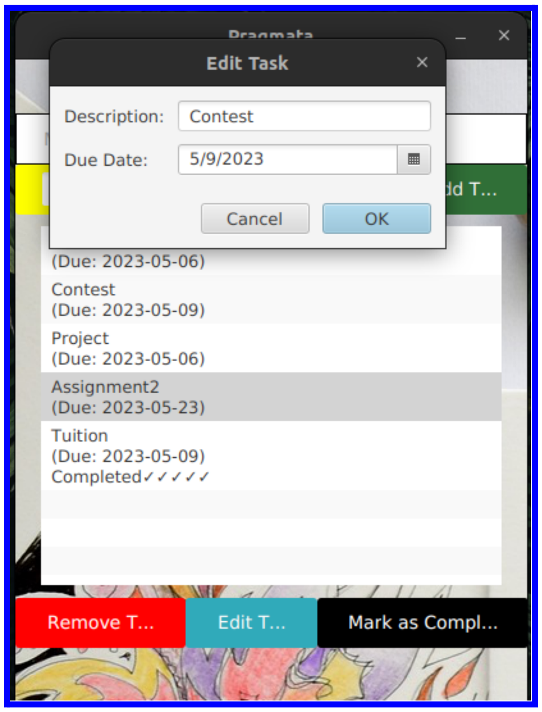


Bus
(Due: 2023-05-06)
Contest
(Due: 2023-05-09)
Project
(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓



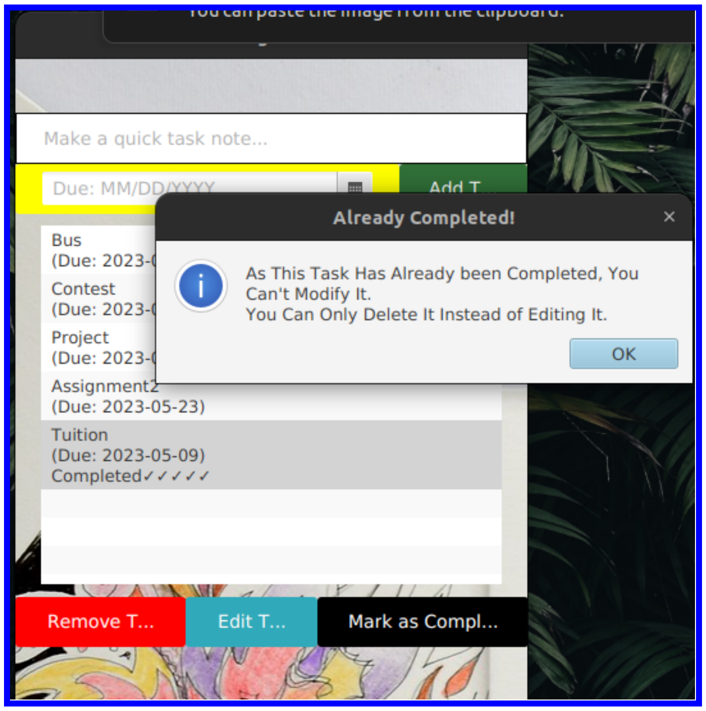
Description:
Due Date:

(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓



Description: Contest
Due Date: 5/9/2023

(Due: 2023-05-06)
Contest
(Due: 2023-05-09)
Project
(Due: 2023-05-06)
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓


You can paste the image from the clipboard.

Make a quick task note...

Due: MM/DD/YYYY

Add T...

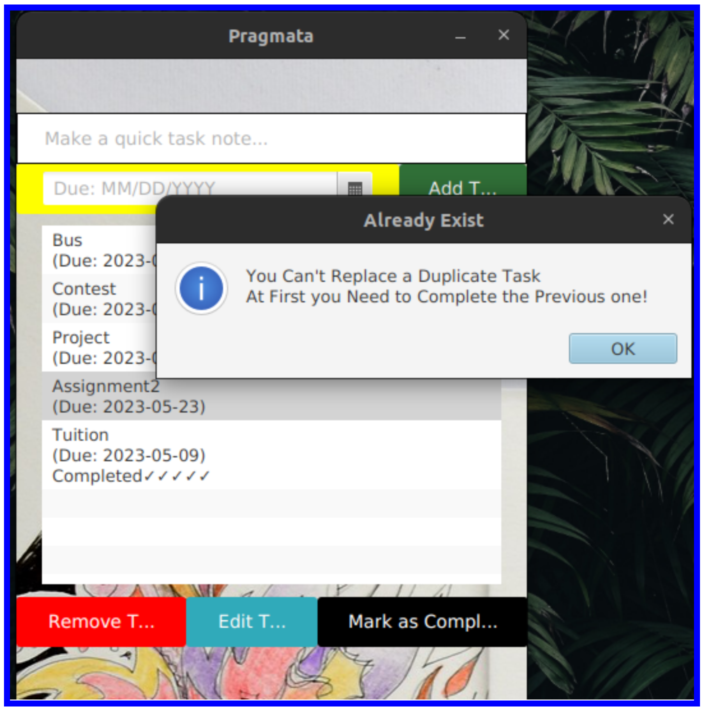
Already Completed!

As This Task Has Already been Completed, You Can't Modify It.
You Can Only Delete It Instead of Editing It.

OK

Bus
(Due: 2023-(
Contest
(Due: 2023-(
Project
(Due: 2023-(
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓

Remove T... | Edit T... | Mark as Compl...


Pragmata

Make a quick task note...

Due: MM/DD/YYYY

Add T...

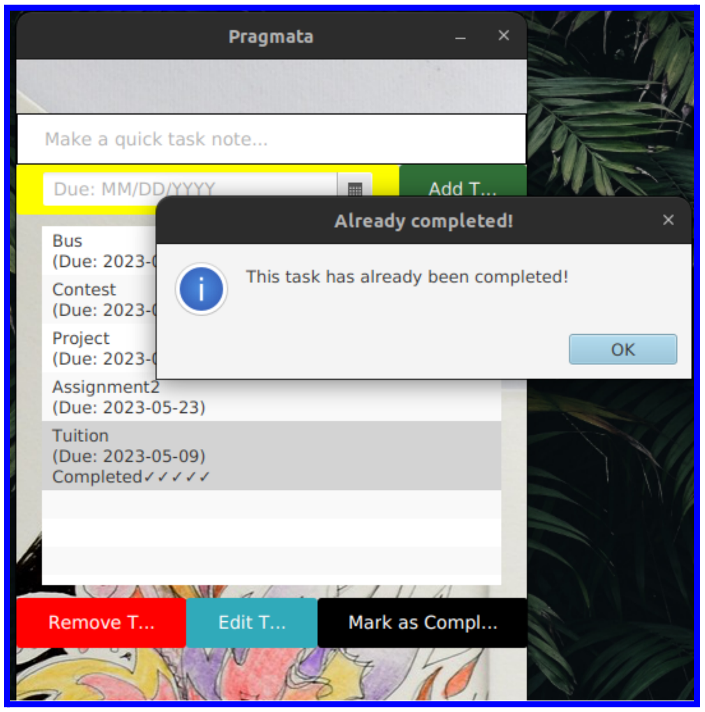
Already Exist

You Can't Replace a Duplicate Task
At First you Need to Complete the Previous one!

OK

Bus
(Due: 2023-(
Contest
(Due: 2023-(
Project
(Due: 2023-(
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓

Remove T... | Edit T... | Mark as Compl...


Pragmata

Make a quick task note...

Due: MM/DD/YYYY

Add T...

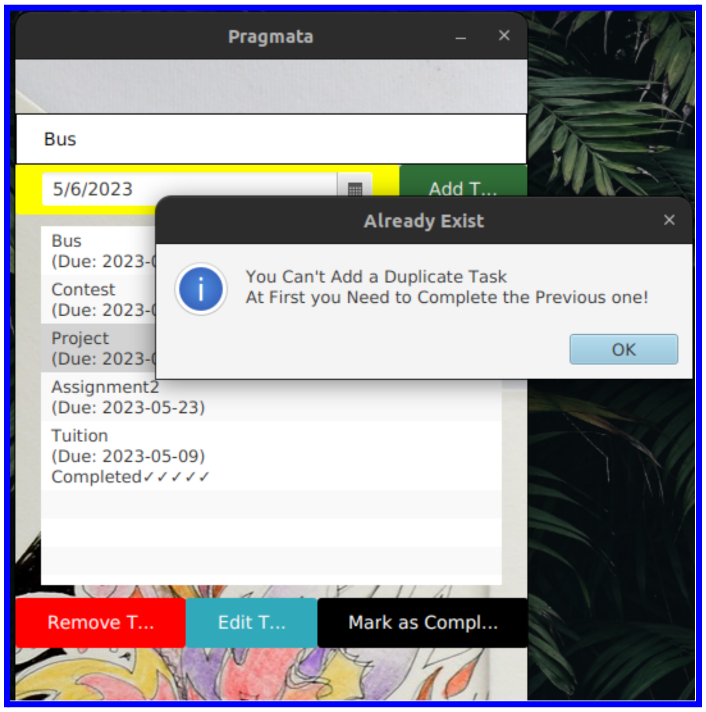
Already completed!

This task has already been completed!

OK

Bus
(Due: 2023-(
Contest
(Due: 2023-(
Project
(Due: 2023-(
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓

Remove T... | Edit T... | Mark as Compl...


Pragmata

Bus

5/6/2023

Add T...

Already Exist

You Can't Add a Duplicate Task
At First you Need to Complete the Previous one!

OK

Bus
(Due: 2023-(
Contest
(Due: 2023-(
Project
(Due: 2023-(
Assignment2
(Due: 2023-05-23)
Tuition
(Due: 2023-05-09)
Completed✓✓✓✓✓

Remove T... | Edit T... | Mark as Compl...

Code Feature:(OOP concepts)

Encapsulation:

- Private instance variables: Use private access modifiers for the data or information within the class to ensure data security.
- Public methods (getters and setters): Implement public getter and setter methods to allow controlled access to private data, enabling external code to retrieve and modify task details.

Inheritance: Used to reuse the code

Polymorphism

- Method Overriding: Implement method overriding by providing specialized implementations of common methods (e.g., toString()) in the subclasses, allowing them to override the behavior inherited from the other class.

Setter and Getter Methods:

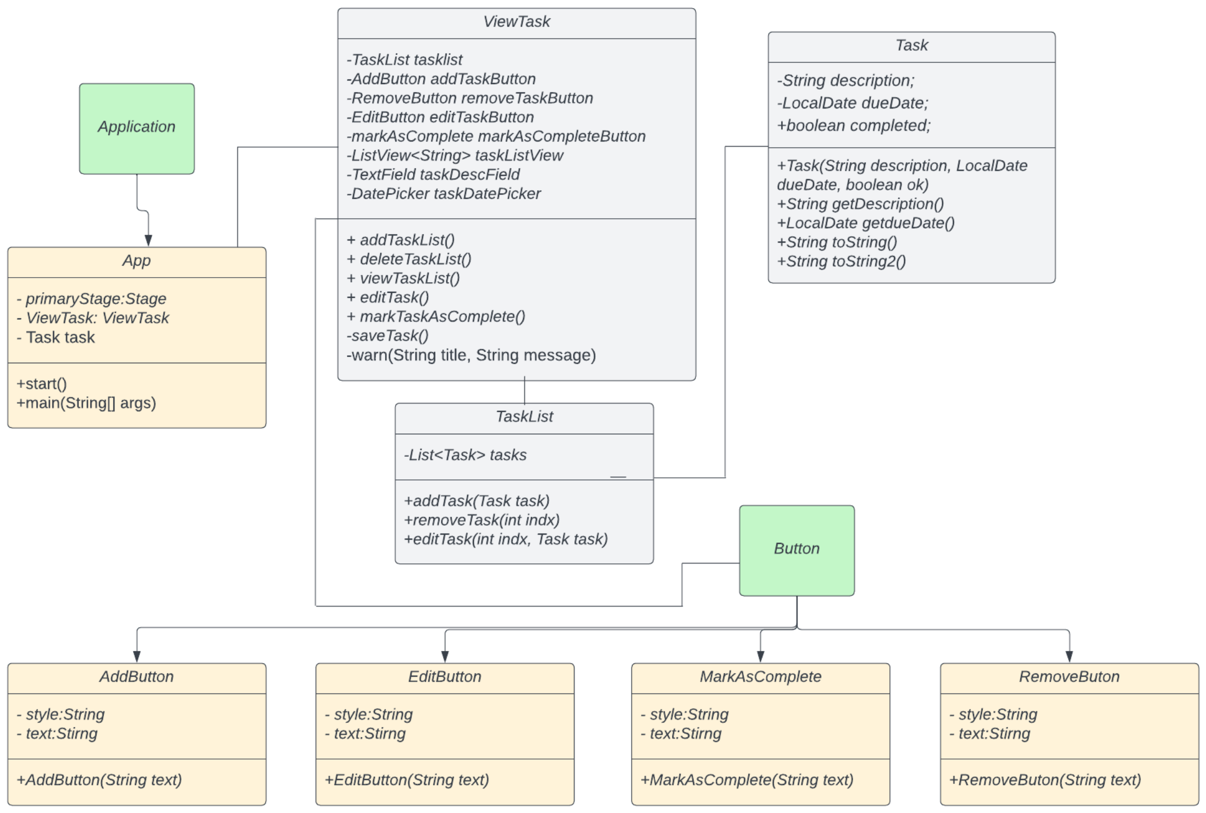
- Setters: Implement setter methods to set various data fields of a task, such as a task description, and due date.
- Getters: Implement getter methods to retrieve the values of the task attributes when needed, allowing other parts of the code to access and utilize the task data.

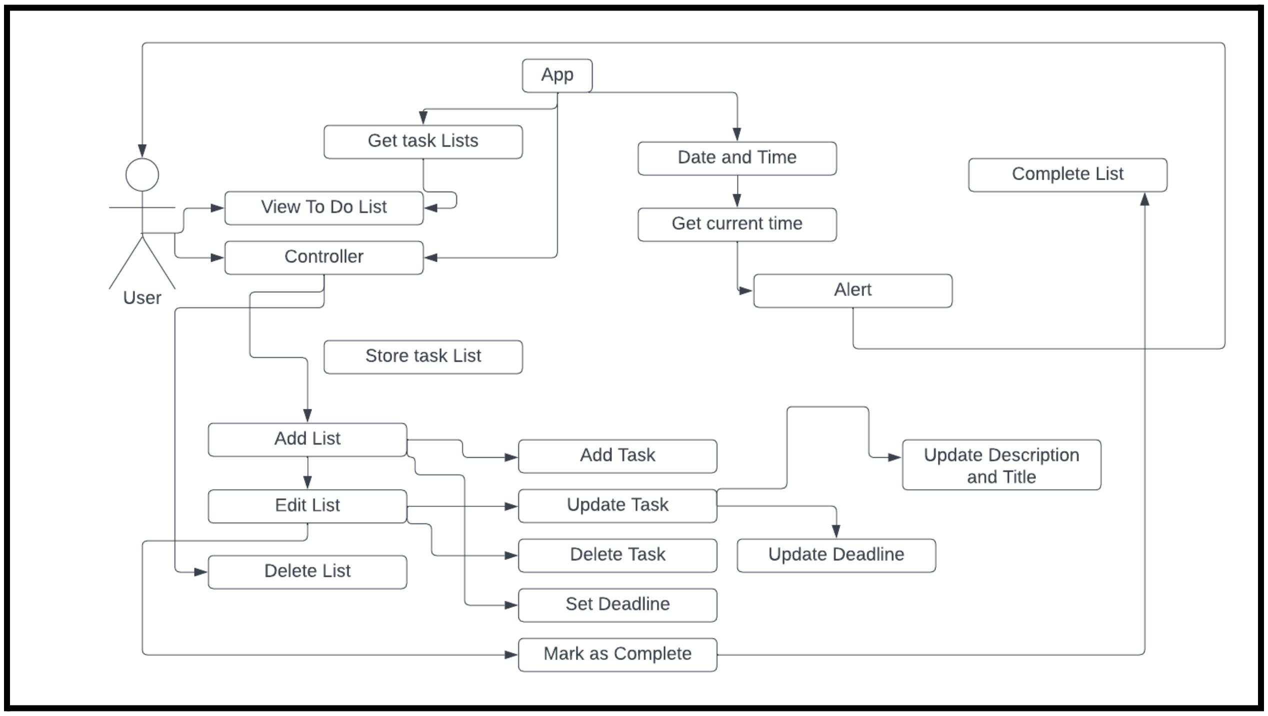
File Saving

Implement functionality to save the to-do list data to a text file. When the user adds, edits, or removes tasks, the changes are written to a file in a structured format

File Loading

Provide the ability to load the saved data from the file when the application starts or when the user requests it. Read the file and populate the to-do list with the previously saved tasks.


Application
App
- primaryStage:Stage
- ViewTask: ViewTask
- Task task
+start()
+main(String[] args)
ViewTask
-TaskList tasklist
-AddButton addTaskButton
-RemoveButton removeTaskButton
-EditButton editTaskButton
-markAsComplete markAsCompleteButton
-ListView<String> taskListView
-TextField taskDescField
-DatePicker taskDatePicker
+ addTaskList()
+ deleteTaskList()
+ viewTaskList()
+ editTask()
+ markTaskAsComplete()
-saveTask()
-warn(String title, String message)
Task
-String description;
-LocalDate dueDate;
+boolean completed;
+Task(String description, LocalDate dueDate, boolean ok)
+String getDescription()
+LocalDate getdueDate()
+String toString()
+String toString2()
TaskList
-List<Task> tasks
+addTask(Task task)
+removeTask(int indx)
+editTask(int indx, Task task)
Button
AddButton
- style:String
- text:Stirng
+AddButton(String text)
EditButton
- style:String
- text:Stirng
+EditButton(String text)
MarkAsComplete
- style:String
- text:Stirng
+MarkAsComplete(String text)
RemoveButon
- style:String
- text:Stirng
+RemoveButon(String text)


App
Get task Lists
Date and Time
Complete List
User
View To Do List
Get current time
Controller
Alert
Store task List
Add List
Add Task
Update Description and Title
Edit List
Update Task
Delete Task
Update Deadline
Delete List
Set Deadline
Mark as Complete

Project Source

- GitHub Repository: https://github.com/ignite312/Pragmata

Team member responsibility

1. Md. Emon Khan-30
   - Designer of Hierarchy of Classes
   - Build system and app control
   - Using File I/O for storing data
   - Code debug and Test.
   - Documentation of Code

2. Saadman Moyeed-33
   - Build system and app control
   - Documentation of Code
   - Code debug and Test.
   - Creating a Use Case Diagram

3. Mahmudul Hasan - 60
   - Class Designer
   - Build system and app control
   - Using File I/O for storing data
   - Creating user interface
   - Code debug and Test.

Limitations

- Task Organization: The application should not allow users to categorize or group tasks based on different criteria such as priority, due date, tags, or custom labels.
- Task Sorting: Users should not be able to sort tasks based on various criteria, such as due date, priority, or alphabetical order, to easily prioritize and manage their to-do list.
- Task Reminders: The application can’t offer the option to set reminders for tasks, allowing users to receive notifications or alerts at specific times or intervals.
- Task Search: Users should not be able to search for specific tasks based on keywords, tags, or other attributes to quickly locate desired tasks.
- Task Priority: Users can not assign priority levels (e.g., high, medium, low) to tasks to highlight their importance or urgency.

Future Plan

Our plans for this application are to focus on enhancing its functionality and user experience. Here are some features that we will improve:

Additional Functionality:

- Task Sorting: Enhance the application by enabling users to sort tasks based on various criteria, such as due date, priority, alphabetical order, or custom sorting options. This will help users prioritize their tasks effectively and find specific tasks quickly.
- Task Reminders: Introduce a feature that allows users to set reminders for tasks. Users can define specific times or intervals at which they would like to receive notifications or alerts, ensuring they stay on top of their tasks and deadlines.
- Task Search: Implement a robust search functionality that enables users to search for specific tasks based on keywords, tags, or other attributes. This will make it easier for users to locate desired tasks, especially when dealing with a large number of tasks.

Mobile Application Development:

An important future plan is to develop a mobile version of the "**Pragmata**" application for iOS and Android platforms. This will allow the application to reach a wider audience and provide a more convenient experience for users on their mobile devices.

User Feedback and Analytics:

It is essential to incorporate mechanisms for gathering user feedback to continuously improve the application. Implementing analytics tools will provide valuable insights into user behavior, application performance, and user preferences. By analyzing this data, the development team can make informed decisions and enhance the application accordingly.

Conclusion

The to-do list app is a powerful tool designed to enhance productivity and bring organization into the lives of its users. Its user-friendly interface and customizable features empower individuals to effectively manage their daily tasks, enabling them to accomplish more and make progress toward their goals. With this app, users can take control of their workload, prioritize tasks, and stay focused on what truly matters. Whether it's managing personal tasks, professional projects, or academic assignments, this to-do list app becomes an indispensable companion for optimizing time management and achieving success. By utilizing the app's intuitive features and harnessing its organizational capabilities, users can experience enhanced productivity, reduced stress, and a greater sense of accomplishment. Embracing this to-do list app is a step towards unlocking one's full potential and making the most out of each day

Overall, Pragmata is an essential app for anyone seeking to optimize their time management and achieve their goals.