MILITARY COLLEGE OF SIGNALS

NATIONAL UNIVERSITY OF SCIENCES & TECHNOLOGY

OPEN-ENDED LAB

Developing a TODO List Desktop Application in Java

SUBMITTED TO: LE AQSA

SUBMISSION DATE	**11.17.2025**
COURSE	**BESE 29-B**
SUBMITTED BY	AMNA NOOR

My Todo App

1. Introduction

The TODO Application is a simple, yet effective task-management tool designed to help users organize their daily responsibilities. The primary objective of this project is to create a user-friendly interface where users can add tasks, mark them as completed, and delete them when no longer needed.
This project demonstrates foundational concepts in frontend development, state management, and interactive UI design, making it ideal for beginner-level software engineering coursework.

2. Objective

Develop a simple TODO list desktop application (similar to Microsoft Notes / Google Keep) using Java, demonstrating the use of object-oriented programming concepts such as classes, objects, inheritance, and polymorphism.
Additional goals include building a GUI, implementing CRUD functionalities, supporting task prioritization, applying modern IDE tools, and using Git for version control

3. Requirements Gathering & Feature Definition (CLO 2 – WA4)

Before implementation, the problem was investigated and analyzed. The following requirements were identified:

3.1 Core Functional Requirements

1. **Create / Add Tasks**
    - Title, description, priority, category, and deadline.
2. **Read / View Tasks**
    - Display tasks in a table format.
3. **Update / Edit Tasks**
    - Modify any field of an existing task.
4. **Delete / Remove Tasks**
    - Permanently remove tasks from the list.
5. **Mark as Completed**

    - Toggle “Done” status using a checkbox.

3.2 Additional Functional Features

- **Task Prioritization**
  Color-coded priority levels (High, Medium, Low).
- **Filtering**
  Filter tasks by category or show completed tasks.
- **Deadline Awareness**
  Automatically highlight:
    - Past deadlines (red)
    - Near deadlines (orange)

3.3 Non-Functional Requirements

- **Usability:** Intuitive and aesthetically pleasing GUI.
- **Persistence:** Ability to save tasks to a file using serialization.
- **Maintainability:** Modular structure using OOP.
- **Responsiveness:** Full-screen and well-spaced layout.
- **Scalability:** Ability to extend with new categories or fields.

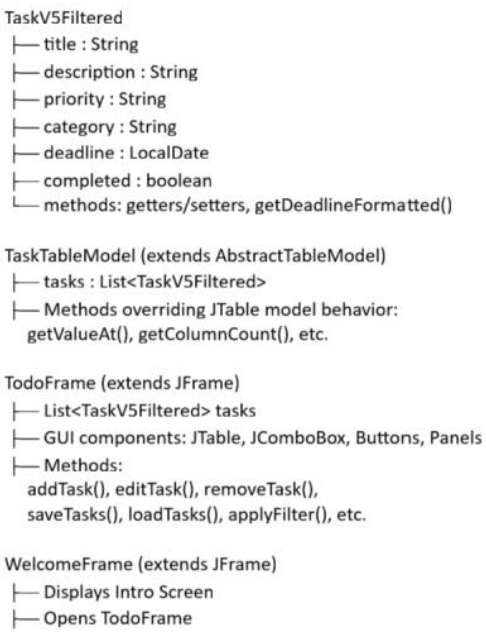
4. System Design & Application Architecture (CLO 2 – WA4)

To fulfill the functional requirements, the system was broken into multiple components.

This document describes the folder and file structure of the My Todo App Java project.
The structure is organized into logical packages for the model, UI, and the application entry point.

4.1Project Directory Structure

4.2 Class Diagram Overview



4.3 OOP Concepts Demonstrated

Concept	Usage
Classes/Objects	Task objects, frame objects, model objects
Encapsulation	Private attributes + getters/setters in TaskV5Filtered
Inheritance	JFrame, AbstractTableModel
Polymorphism	Overriding methods like getValueAt() & renderer behaviors
Serialization	Saving and loading task objects

4.4 Data Structures Used

- ArrayList for storing tasks
- JTable + Custom TableModel for rendering data
- HashMap for text styling attributes in renderer

5. Tools & Technologies Used (CLO 4 – WAS)

Tool	Purpose
java 24.0.1 2025-04-15	Base programming language
Swing	GUI development
Visual Studio Code	IDE for coding, debugging, building
Git / GitHub	Version control and iterative development

5.1 Version Control Using Git & GitHub Desktop (CLO 4 – WAS)

Version control was implemented using Git with GitHub Desktop as the graphical interface throughout application development. GitHub Desktop provided an organized workflow for tracking changes, synchronizing code, and managing updates efficiently.

The following practices were adopted:

Commit Management

- Frequent commits were made after completing every small, functional change.
- Descriptive commit messages (e.g., *"Added custom table renderer for priority colors"*) ensured clear traceability of development progress.

Branching & Experimentation

- Separate branches were used for testing UI improvements and feature upgrades.
- Only tested and stable changes were merged into the *main* branch.

Backup & Synchronization

- The entire project was continuously backed up to a GitHub repository.
- GitHub Desktop ensured synchronized changes across devices when needed.

Error Recovery

- Version control allowed rolling back to earlier commits if unexpected behavior occurred.
- This protected the project from accidental file corruption or flawed updates.

Modern Tool Integration

Using GitHub Desktop aligns with CLO 4, demonstrating the ability to use modern software engineering tools for professional-grade development, collaboration, and iterative improvement.

6. Implementation Summary (CLO 4 – WAS)

The application was implemented in modular Java packages:

6.1 Model Layer

TaskV5Filtered.java

- Represents a single task with attributes and formatted deadline.
- Serializable to allow file persistence.

6.2 Table Model Layer

TaskTableModel.java

- Custom table model controlling how tasks appear in the JTable.
- Supports checkbox editing for "completed" field.

6.3 GUI Layer

TodoFrame.java

- Main window of the application.
- Includes:
    - Form for new tasks
    - Table view
    - Filters & category selection
    - Buttons: Add, Edit, Remove, Save, Load
- Includes custom renderer for:
    - Priority colors
    - Deadline warnings
    - Completed task strikethrough

WelcomeFrame.java

- Intro screen.
- Launches TodoFrame on button press.

6.4 File Persistence

Tasks are saved using:

```
ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("todo_tasks_v5_filtered.ser"));
```

and loaded using:

```
ObjectInputStream ois = new ObjectInputStream(new FileInputStream("todo_tasks_v5_filtered.ser"));
```

7. Testing & Validation (CLO 2 – WA4)

Multiple test cycles were performed:

Feature	Result
Add task	Passed
Edit task	Passed
Delete task	Passed
Toggle completion	Passed
Apply category filter	Passed
Deadline highlighting	Passed
Show completed tasks	Passed
Save	Passed
Load	Passed

7.2 Usability Testing

- GUI was checked for responsiveness in different window sizes.
- Fonts, colors, and spacing were tested for readability.

7.3 Robustness Testing

- Empty title warning works correctly.
- Invalid or missing fields do not crash the app.
- Serialization works even after dozens of tasks.

8. Results

The final application fully meets the objectives:

- Supports complete CRUD operations
- Provides a rich and intuitive GUI
- Implements OOP properly
- Uses modern tools (IDE, Git, Swing, serialization)
- Allows multiple ways of organizing and filtering tasks
- UI design enhances user experience

9. Conclusion

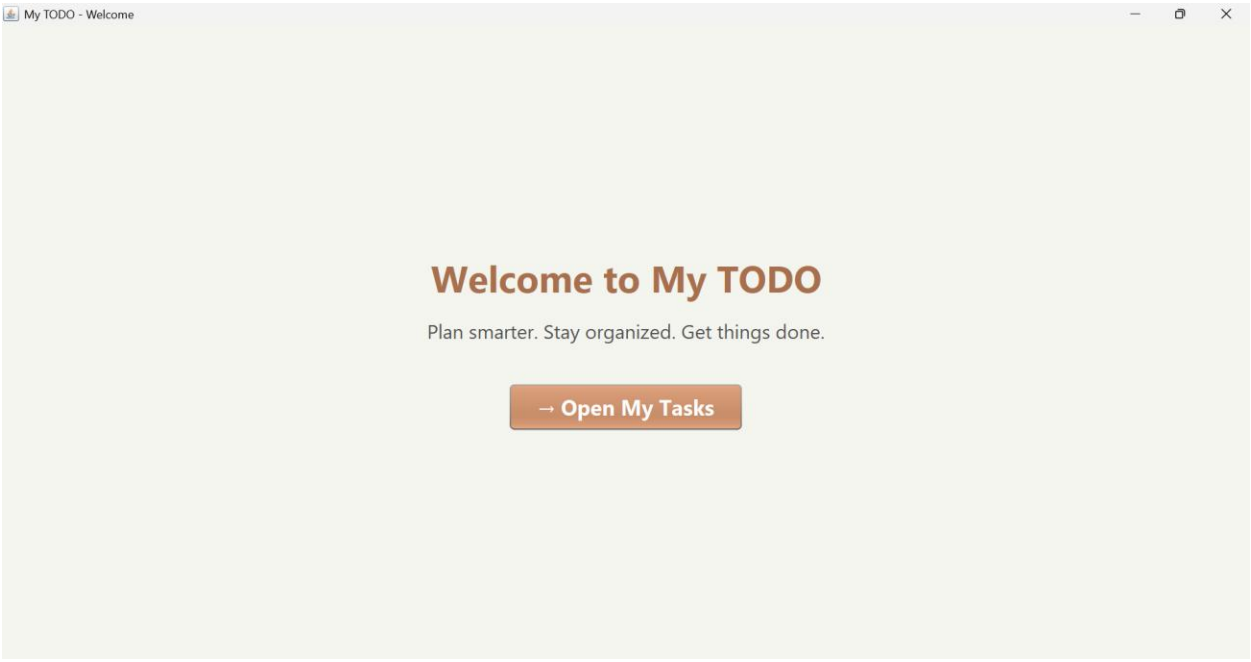
In conclusion, this application was developed by following a structured and systematic approach that covered requirements analysis, design, implementation, and testing. Throughout the development cycle, version control played a central role in maintaining project integrity. Git and GitHub Desktop were used to manage code changes, track revisions, and ensure safe, organized collaboration. Regular commits, clear commit

messages, and branch management allowed for efficient development and error recovery whenever needed.

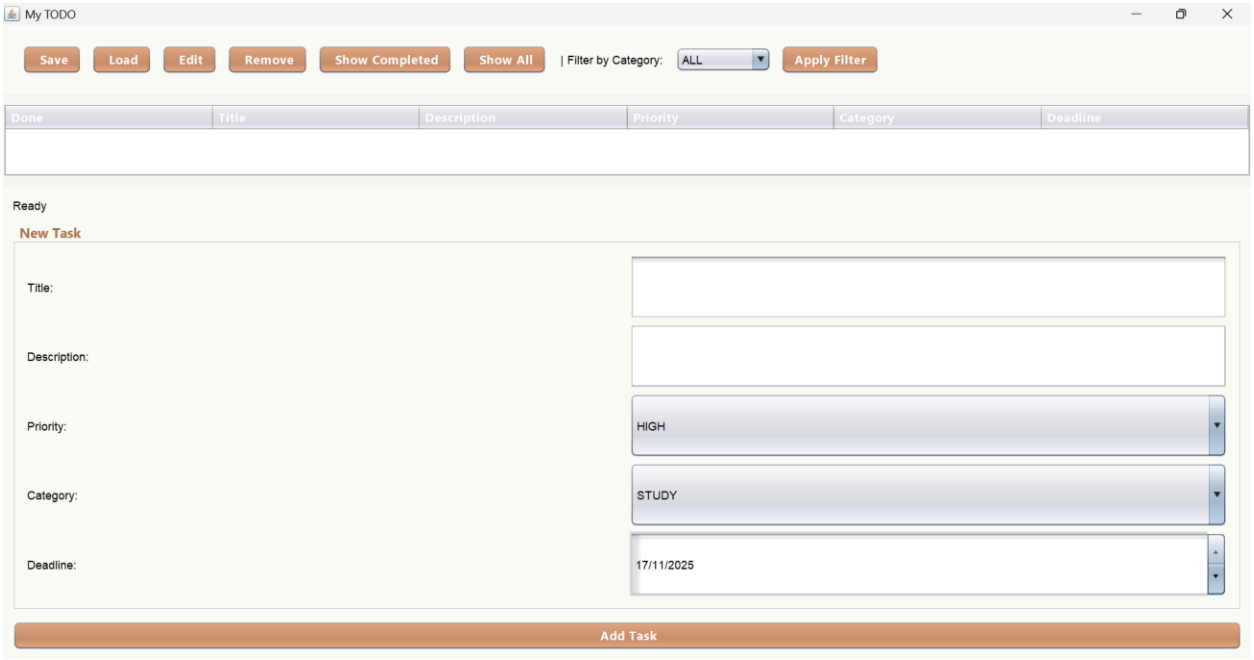
Overall, the project successfully meets its defined objectives and provides a stable, functional solution. The use of proper development practices—supported by version control—ensured a smooth workflow and created a strong foundation for future enhancements or feature extensions.

10. Screenshots

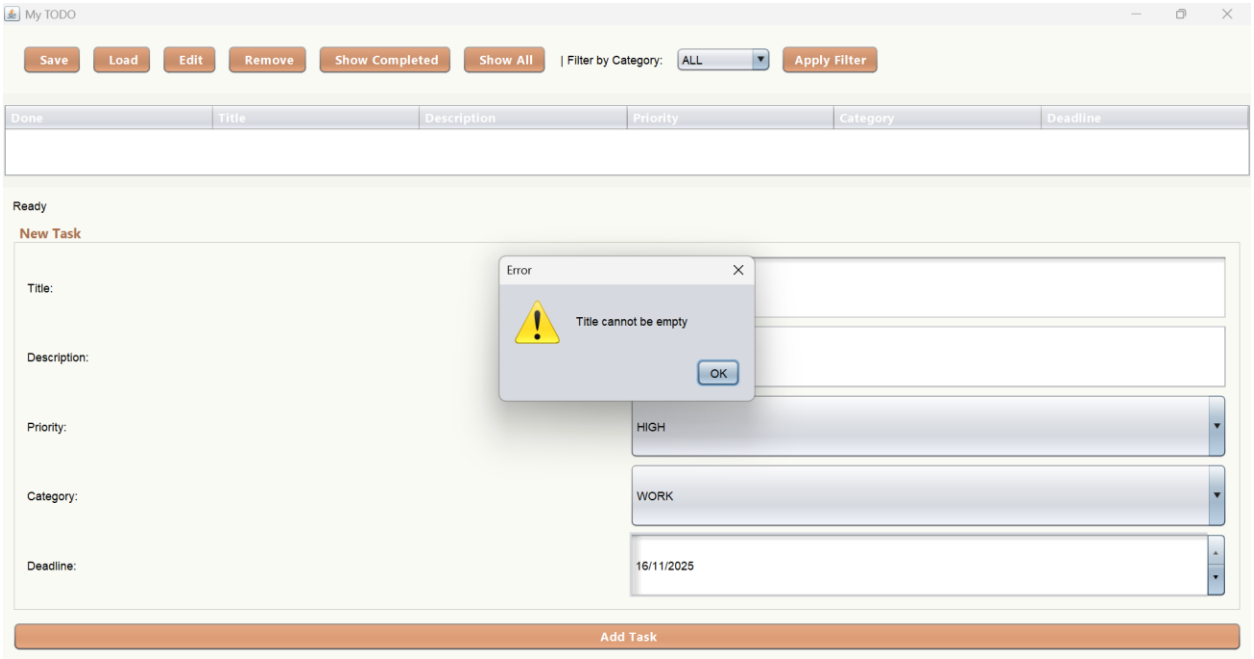
- Welcome screen

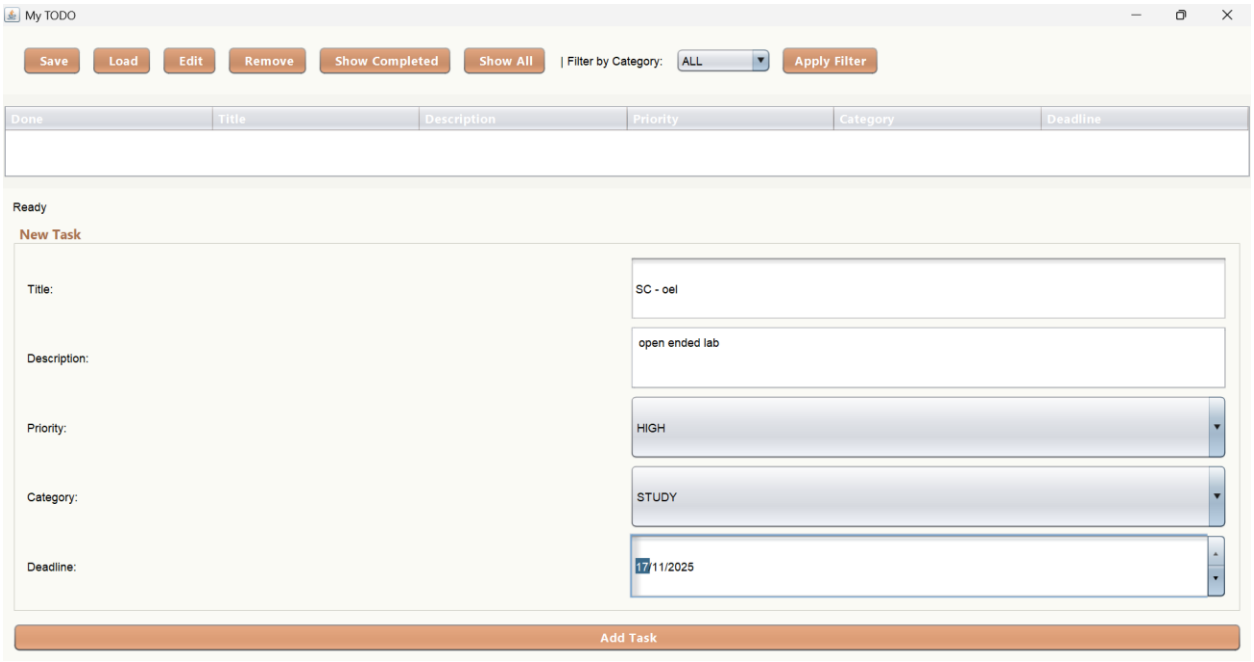


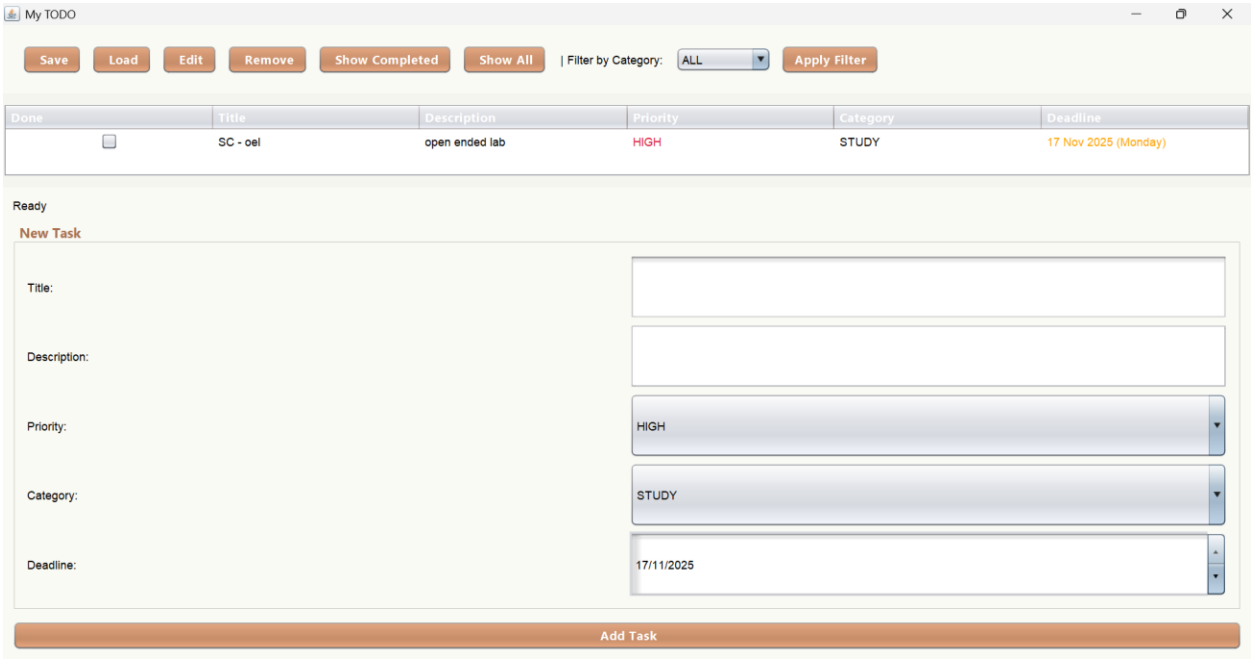
- Main TODO window



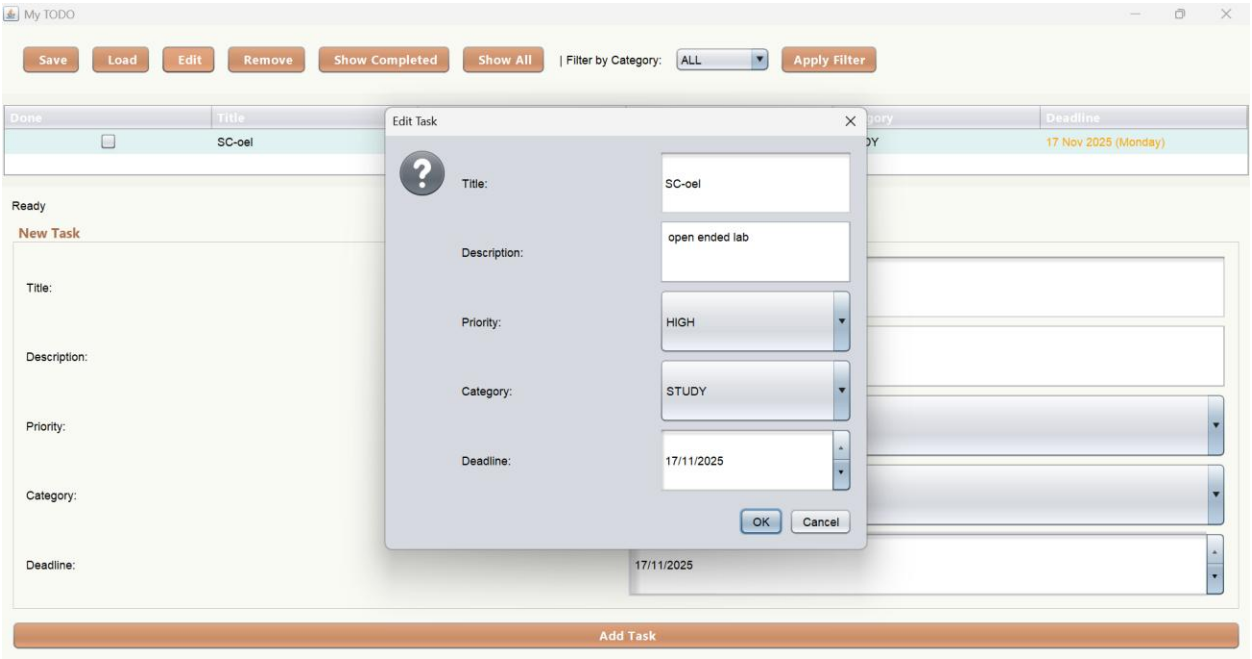
- Add task dialog

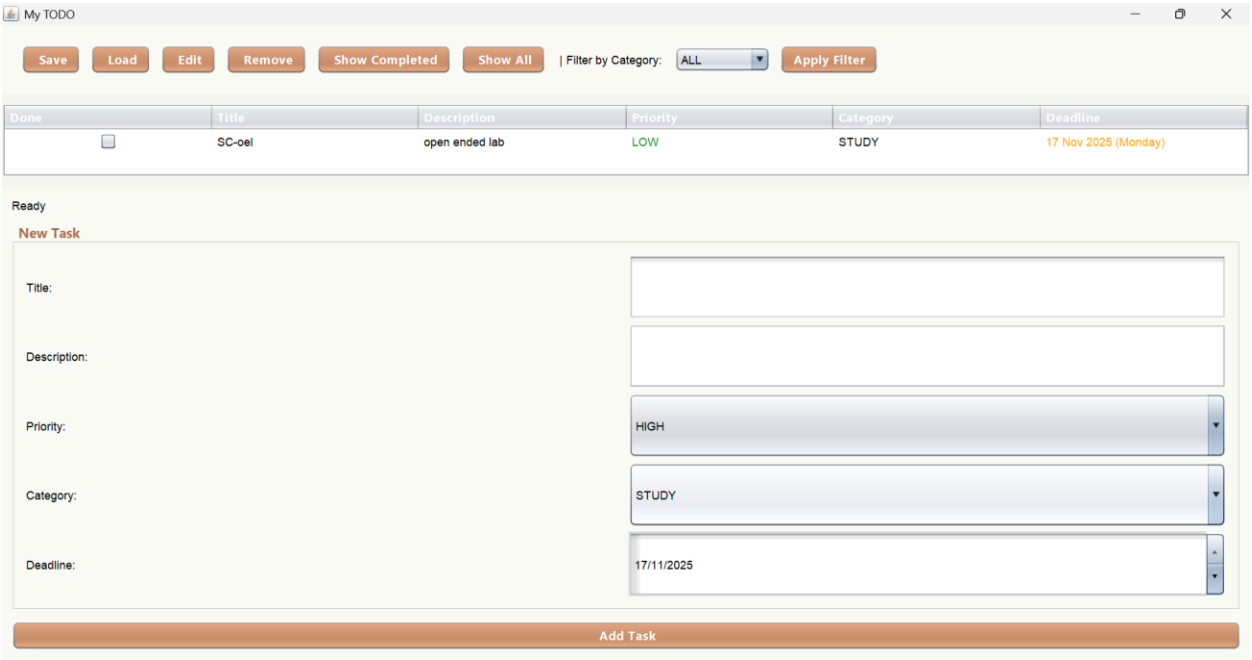



My TODO
Save
Load
Edit
Remove
Show Completed
Show All
| Filter by Category:
ALL
Apply Filter
Done
Title
Description
Priority
Category
Deadline
Ready
New Task
Title:
SC - oel
Description:
open ended lab
Priority:
HIGH
Category:
STUDY
Deadline:
17/11/2025
Add Task

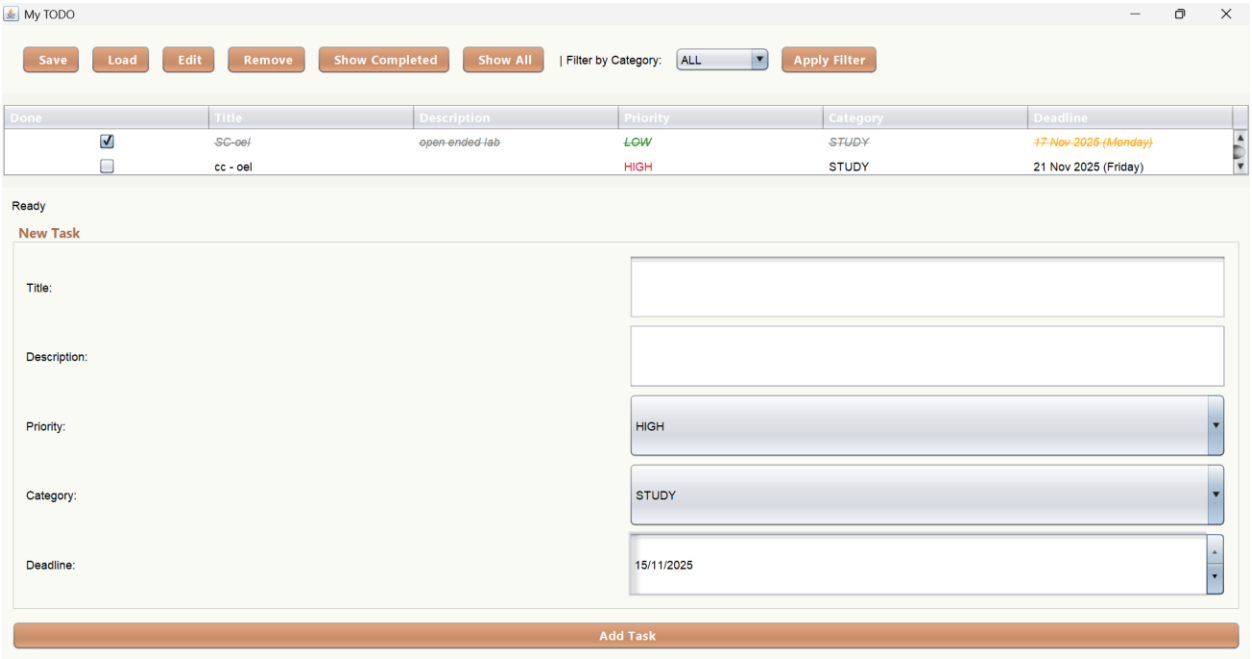

My TODO
Save
Load
Edit
Remove
Show Completed
Show All
| Filter by Category:
ALL
Apply Filter
Done
Title
Description
Priority
Category
Deadline
SC - oel
open ended lab
HIGH
STUDY
17 Nov 2025 (Monday)
Ready
New Task
Title:
Description:
Priority:
HIGH
Category:
STUDY
Deadline:
17/11/2025
Add Task

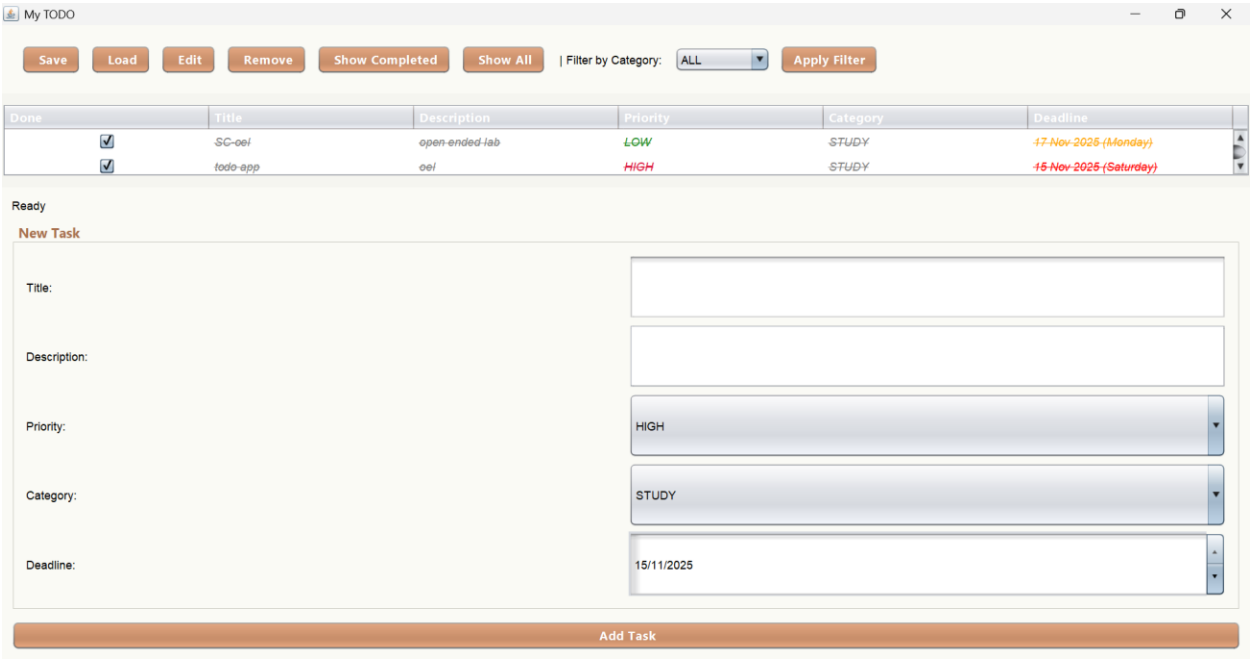
- Edited task


My TODO
Save
Load
Edit
Remove
Show Completed
Show All
| Filter by Category:
ALL
Apply Filter
Done
Title
Deadline
SC-oel
17 Nov 2025 (Monday)
Edit Task
Title:
SC-oel
Description:
open ended lab
Priority:
HIGH
Category:
STUDY
Deadline:
17/11/2025
OK
Cancel
Ready
New Task
Title:
Description:
Priority:
Category:
Deadline:
17/11/2025
Add Task

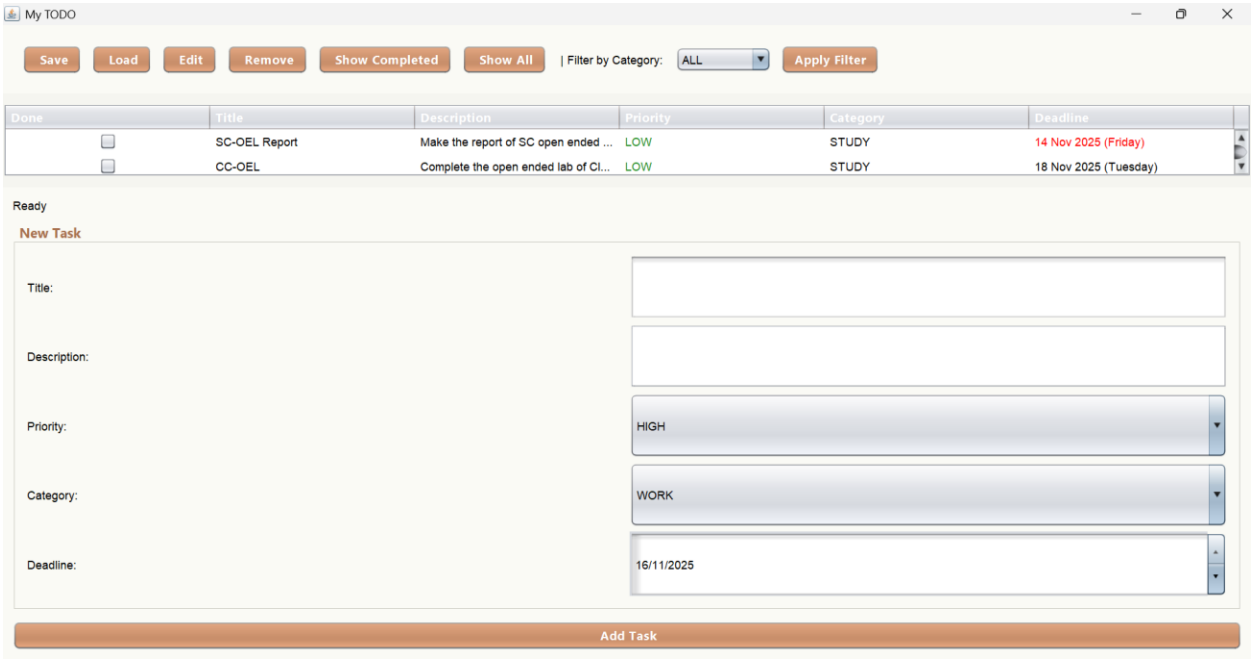

My TODO
Save
Load
Edit
Remove
Show Completed
Show All
| Filter by Category:
ALL
Apply Filter
Done
Title
Description
Priority
Category
Deadline
SC-oel
open ended lab
LOW
STUDY
17 Nov 2025 (Monday)
Ready
New Task
Title:
Description:
Priority:
HIGH
Category:
STUDY
Deadline:
17/11/2025
Add Task

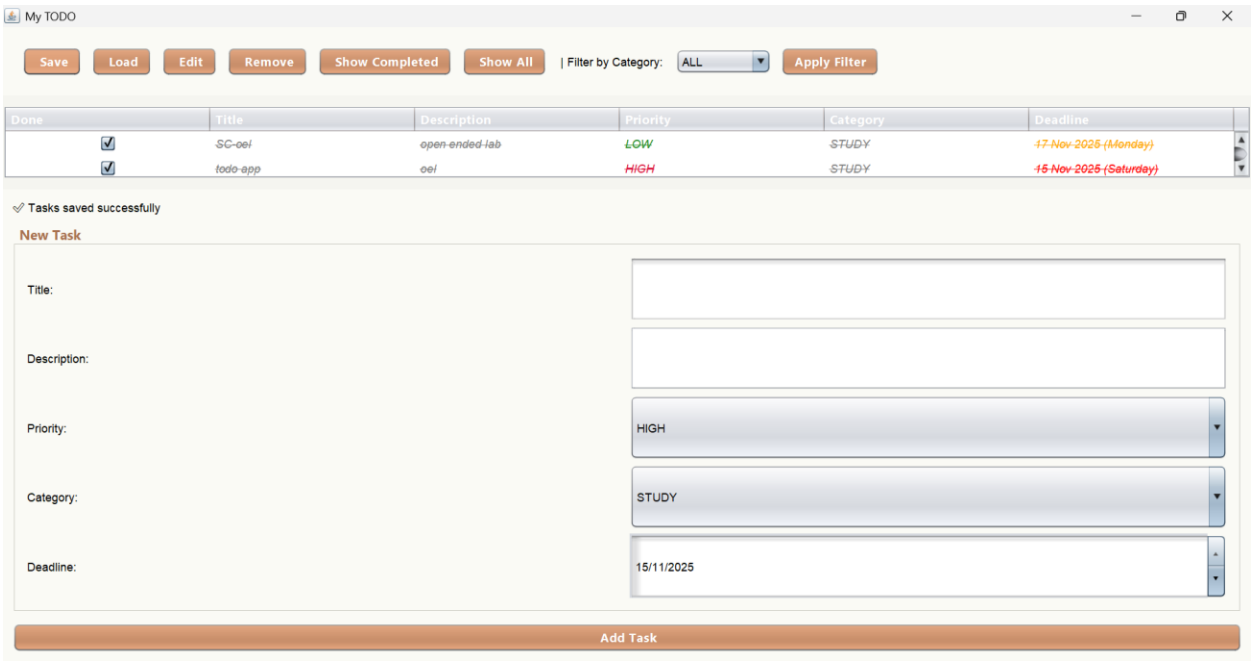
- Completed tasks view



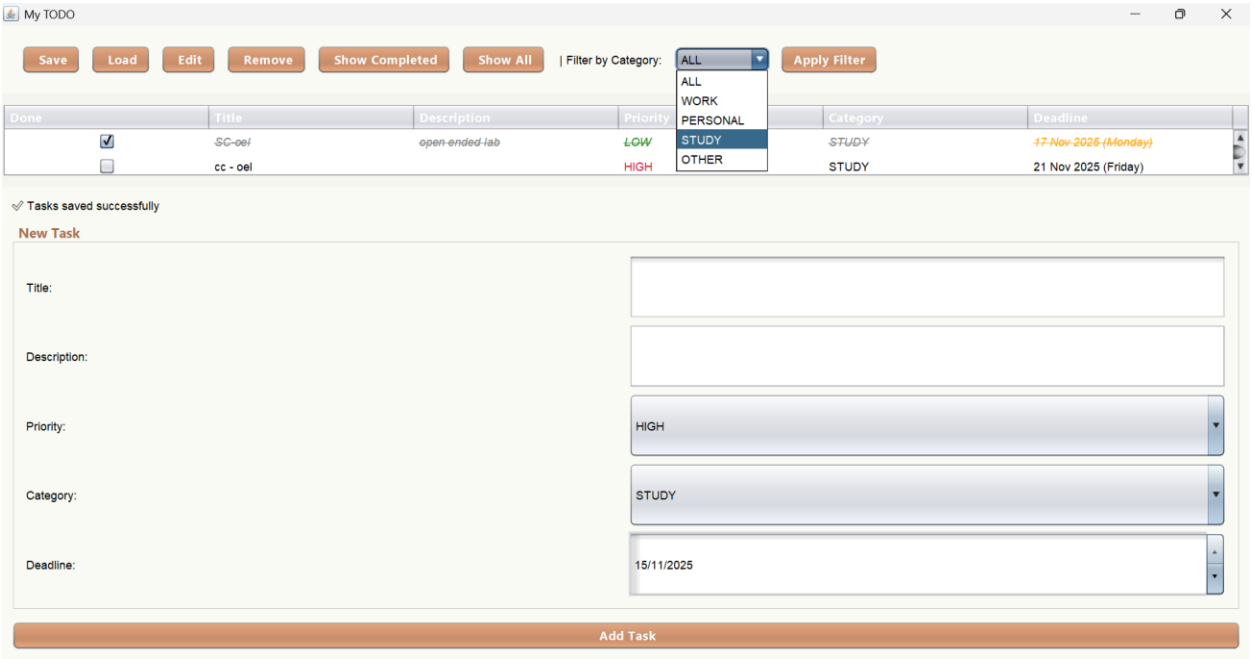


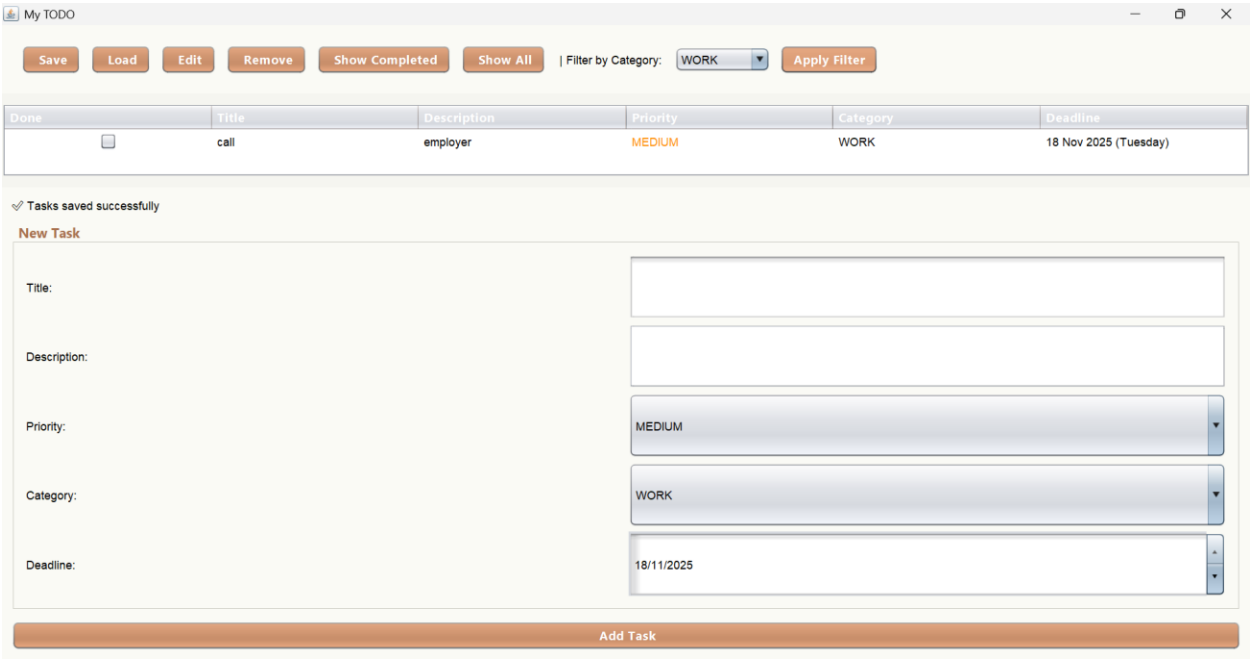
- Saved file in project directory



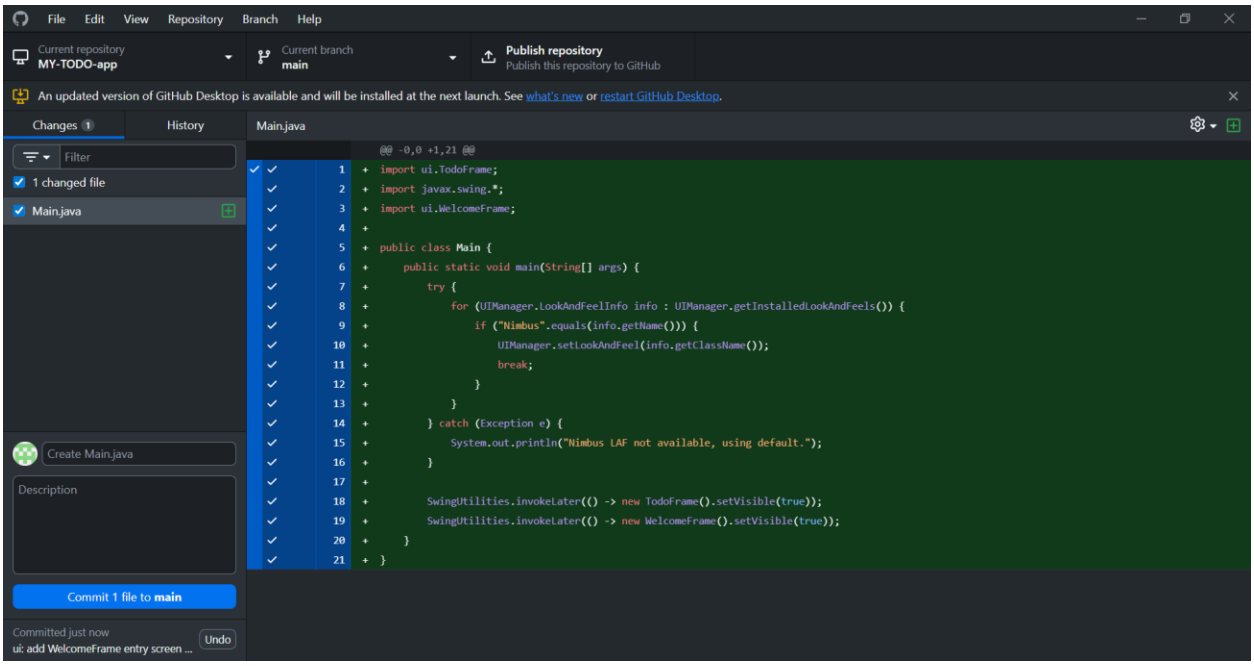


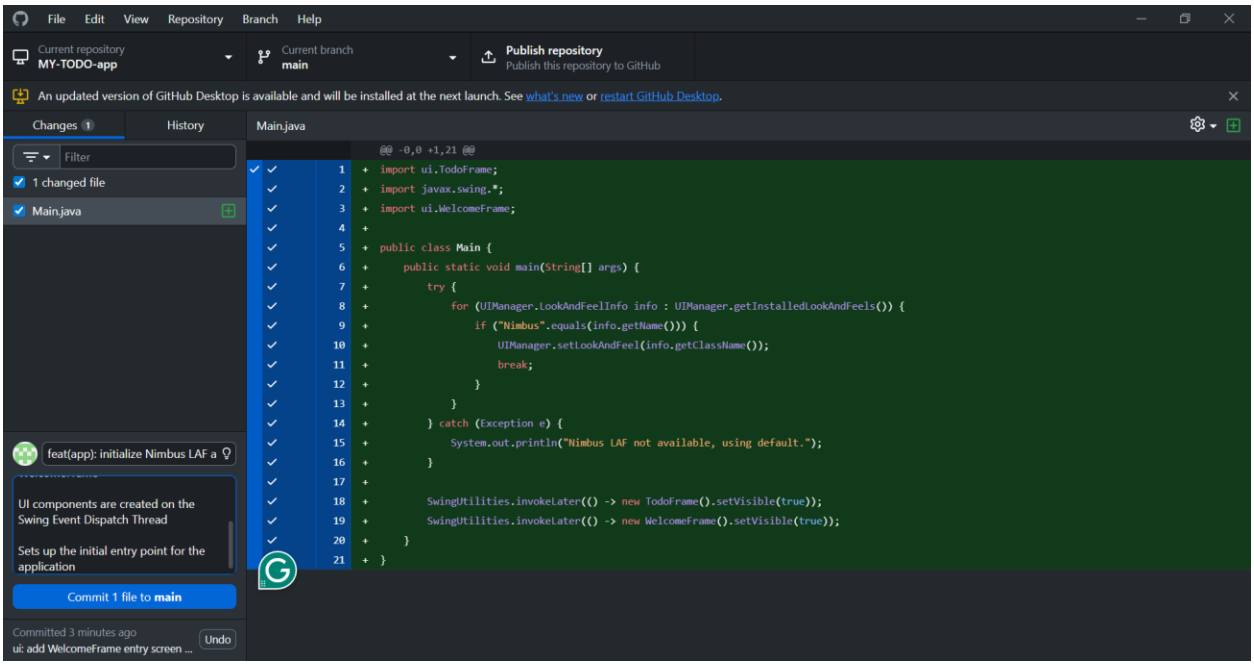
- Filter by Category

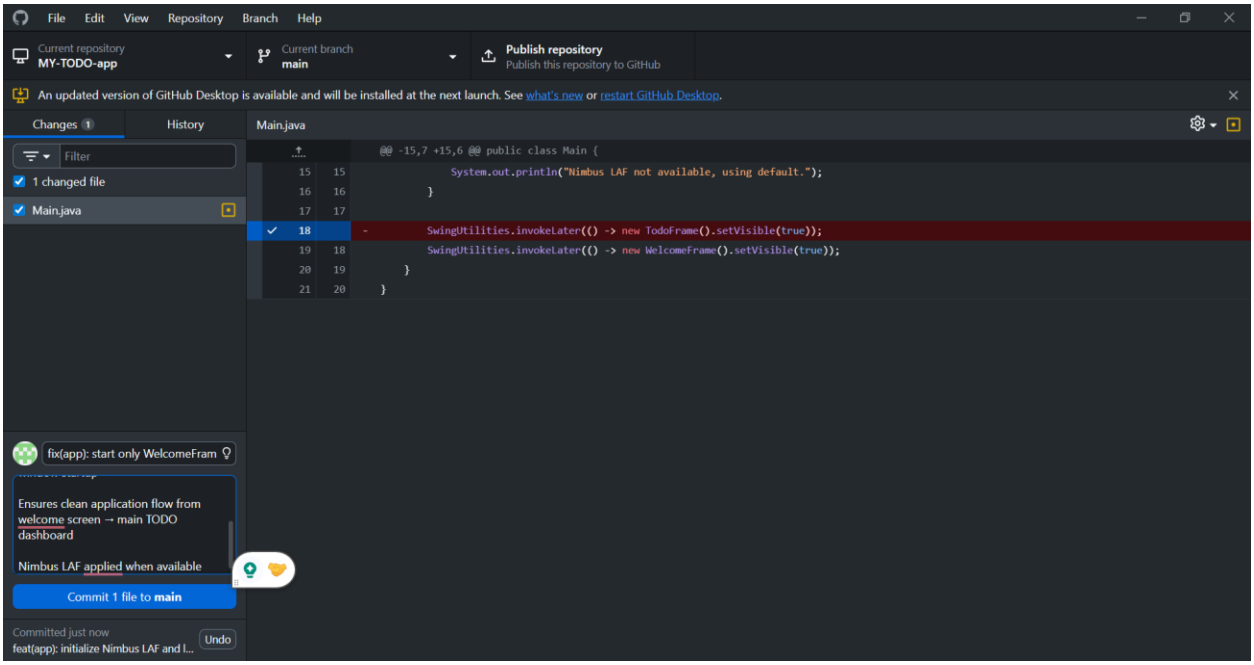


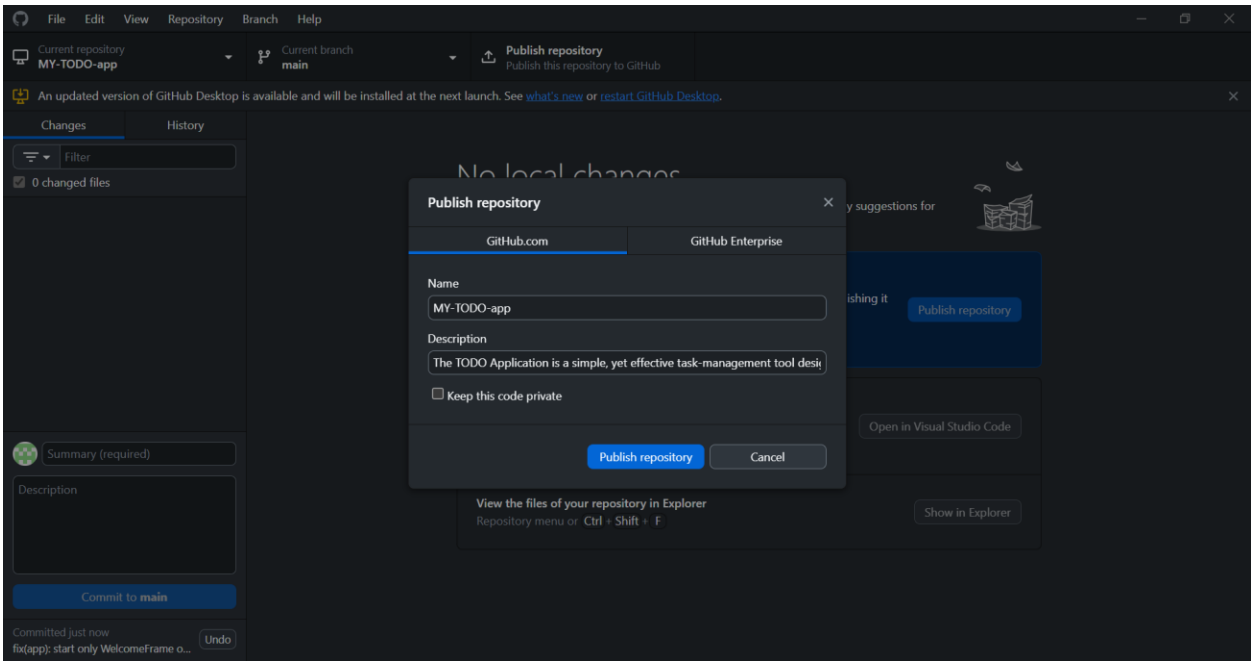


- Github Desktop

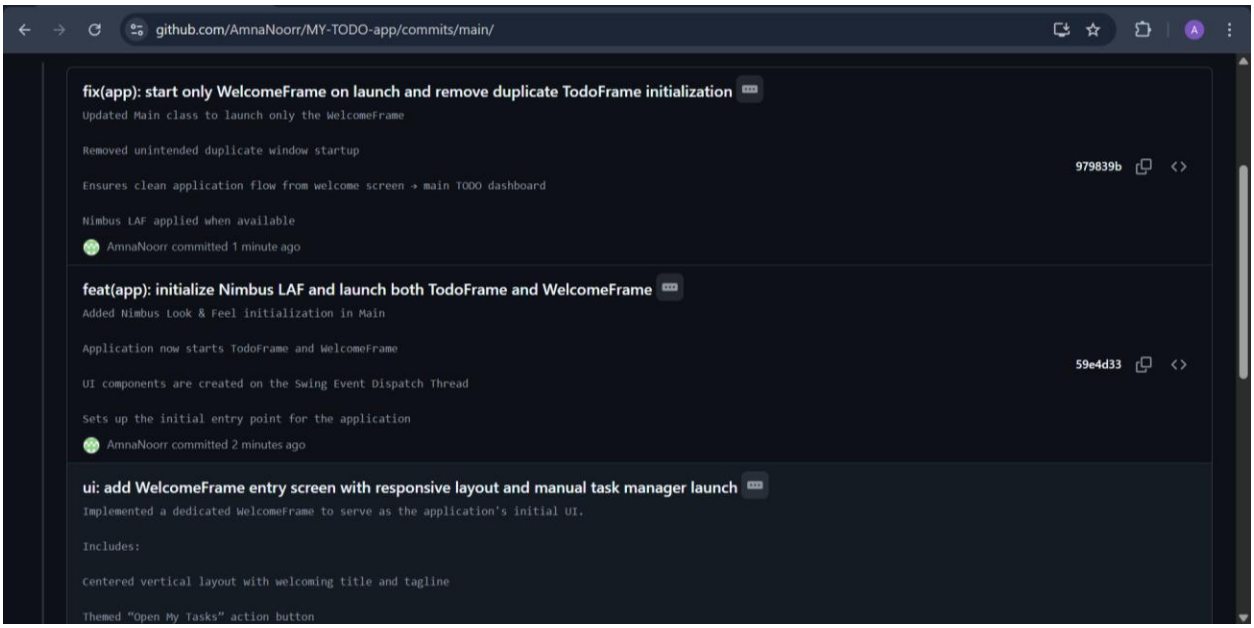


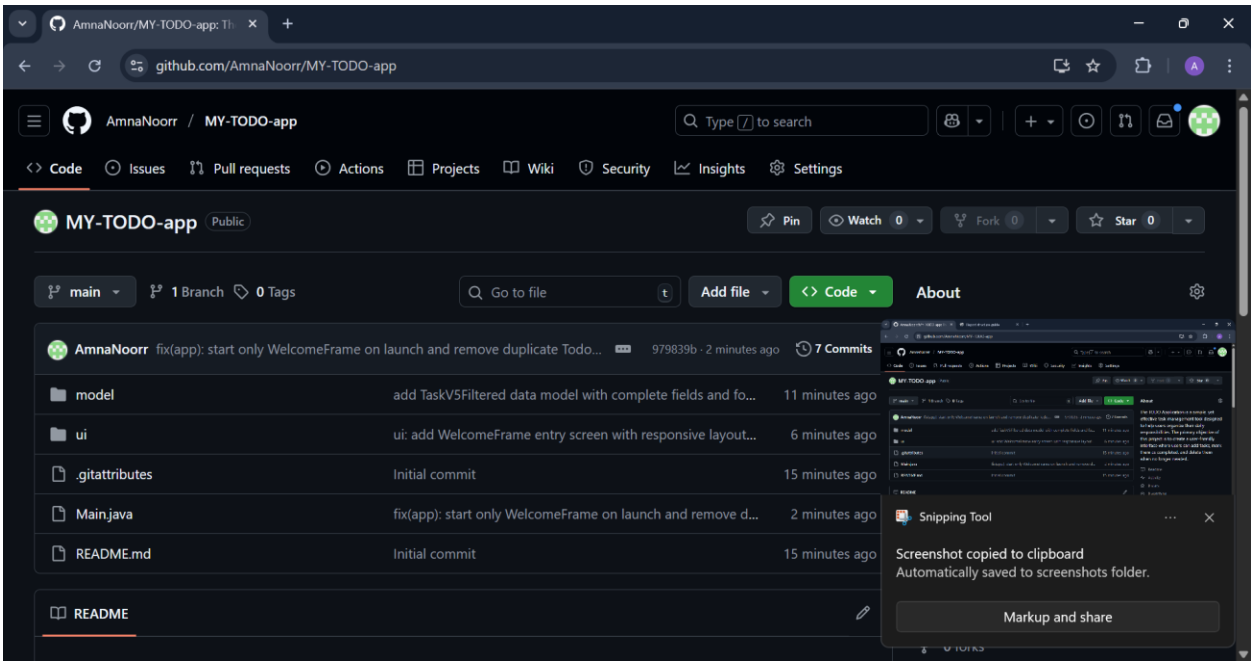



File Edit View Repository Branch Help
Current repository
MY-TODO-app
Current branch
main
Publish repository
Publish this repository to GitHub
An updated version of GitHub Desktop is available and will be installed at the next launch. See what's new or restart GitHub Desktop.
Changes 1
History
Filter
1 changed file
Main.java
Main.java
@@ -15,7 +15,6 @@ public class Main {
System.out.println("Nimbus LAF not available, using default.");
}
- SwingUtilities.invokeLater(() -> new TodoFrame().setVisible(true));
SwingUtilities.invokeLater(() -> new WelcomeFrame().setVisible(true));
}
}
fix(app): start only WelcomeFram
Ensures clean application flow from welcome screen → main TODO dashboard
Nimbus LAF applied when available
Commit 1 file to main
Committed just now
feat(app): initialize Nimbus LAF and l...
Undo


File Edit View Repository Branch Help
Current repository
MY-TODO-app
Current branch
main
Publish repository
Publish this repository to GitHub
An updated version of GitHub Desktop is available and will be installed at the next launch. See what's new or restart GitHub Desktop.
Changes
History
Filter
0 changed files
No local changes
Publish repository
GitHub.com
GitHub Enterprise
Name
MY-TODO-app
Description
The TODO Application is a simple, yet effective task-management tool desi
Keep this code private
Publish repository
Cancel
Open in Visual Studio Code
View the files of your repository in Explorer
Repository menu or Ctrl + Shift + F
Show in Explorer
Summary (required)
Description
Commit to main
Committed just now
fix(app): start only WelcomeFrame o...
Undo

- Github Web


github.com/AmnaNoorr/MY-TODO-app/commits/main/
fix(app): start only WelcomeFrame on launch and remove duplicate TodoFrame initialization
Updated Main class to launch only the WelcomeFrame
Removed unintended duplicate window startup
Ensures clean application flow from welcome screen → main TODO dashboard
Nimbus LAF applied when available
AmnaNoorr committed 1 minute ago
979839b
feat(app): initialize Nimbus LAF and launch both TodoFrame and WelcomeFrame
Added Nimbus Look & Feel initialization in Main
Application now starts TodoFrame and WelcomeFrame
UI components are created on the Swing Event Dispatch Thread
Sets up the initial entry point for the application
AmnaNoorr committed 2 minutes ago
59e4d33
ui: add WelcomeFrame entry screen with responsive layout and manual task manager launch
Implemented a dedicated WelcomeFrame to serve as the application's initial UI.
Includes:
Centered vertical layout with welcoming title and tagline
Themed "Open My Tasks" action button


github.com/AmnaNoorr/MY-TODO-app
AmnaNoorr / MY-TODO-app
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
MY-TODO-app
Public
Pin
Watch 0
Fork 0
Star 0
main
1 Branch
0 Tags
Go to file
Add file
Code
About
AmnaNoorr fix(app): start only WelcomeFrame on launch and remove duplicate Todo...
979839b · 2 minutes ago
7 Commits
model
add TaskV5Filtered data model with complete fields and fo...
11 minutes ago
ui
ui: add WelcomeFrame entry screen with responsive layout...
6 minutes ago
.gitattributes
Initial commit
15 minutes ago
Main.java
fix(app): start only WelcomeFrame on launch and remove d...
2 minutes ago
README.md
Initial commit
15 minutes ago
README
Snipping Tool
Screenshot copied to clipboard
Automatically saved to screenshots folder.
Markup and share