



Java Development Intern

Task 1 Report:

To-Do List Application

MOHAU LIPHOKO

Table of Contents

Introduction.....	3
Overview.....	3
Prerequisites to run the application.....	4
How to run the application.....	4
i) Compile java classes.....	4
ii) Running the application.....	4
iii) Executing JUnits.....	4
Functionalities.....	6
1) Add task.....	6
2) View task.....	7
3) Update task.....	7
4) Delete task.....	9
5) Auto-Increment ID.....	10
6) Persistence.....	10
Special instructions.....	11
Conclusion.....	11

Introduction

Task management is essential when one would like to have their things in order. Throughout the thorough manual for the To-Do List Application, a JavaFX project created to provide you with a graphical user interface for effectively managing your activities, you will be guided and given a paradigm shift on how to handle your tasks more effectively and efficiently. This program ensures a smooth and user-friendly experience by utilizing a powerful mix of a SQL database for data storage and JavaFX for the front-end. You will discover how to launch the application, use its features, and take full advantage of all its features by following this guide. Whether you are adding, viewing, editing, or removing tasks, you can easily access each function with the help of this tutorial, which offers simple instructions.

Overview

The To-Do List Application is a Java-based application that allows users to manage their tasks. Users can add, view, update, and delete tasks. Each task has an ID, description, due date, and a completion status. The application uses a MySQL database to store tasks and JavaFX for the graphical user interface. The application allows users to perform the following operations:

- Add new tasks
- Retrieve a list of all tasks
- Update existing tasks
- Delete tasks

The application ensures data integrity by using a relational database to store task information. Each task has the following attributes:

- **id:** A unique identifier for the task.
- **description:** A brief description of the task.
- **dueDate:** The date by which the task should be completed.
- **completed:** A Boolean indicating whether the task is completed.

Prerequisite to run the application:

1. **Java Development Kit (JDK):** Ensure you have JDK 8 or above installed.

2. **MySQL Database:** Install and configure MySQL server.
3. **IntelliJ IDEA:** Install IntelliJ IDEA as your development environment.

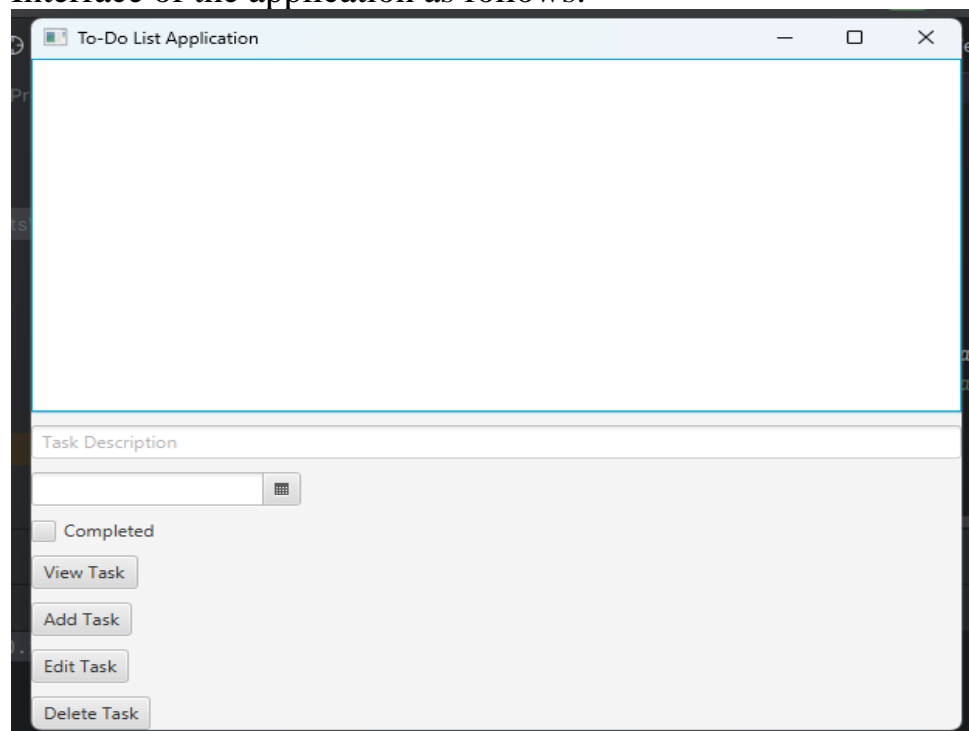
How to run the application

1. Compile the Java Classes:

- Ensure that all Java classes, including Database, ToDoListGUI, Task, DatabaseTest, and TestApp, are compiled without errors.

2. Running the Application:

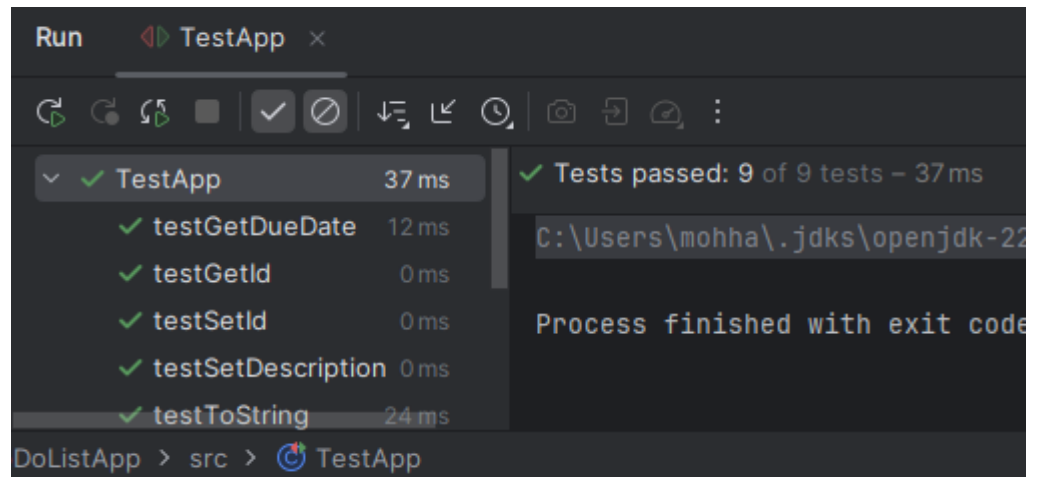
- The main entry point for the application should invoke methods from the Database class to perform various operations. Functionalities are pictorially presented as part of illustrating what happens when the application is run.
- Running the ToDoListGUI will open the Graphical User Interface of the application as follows:



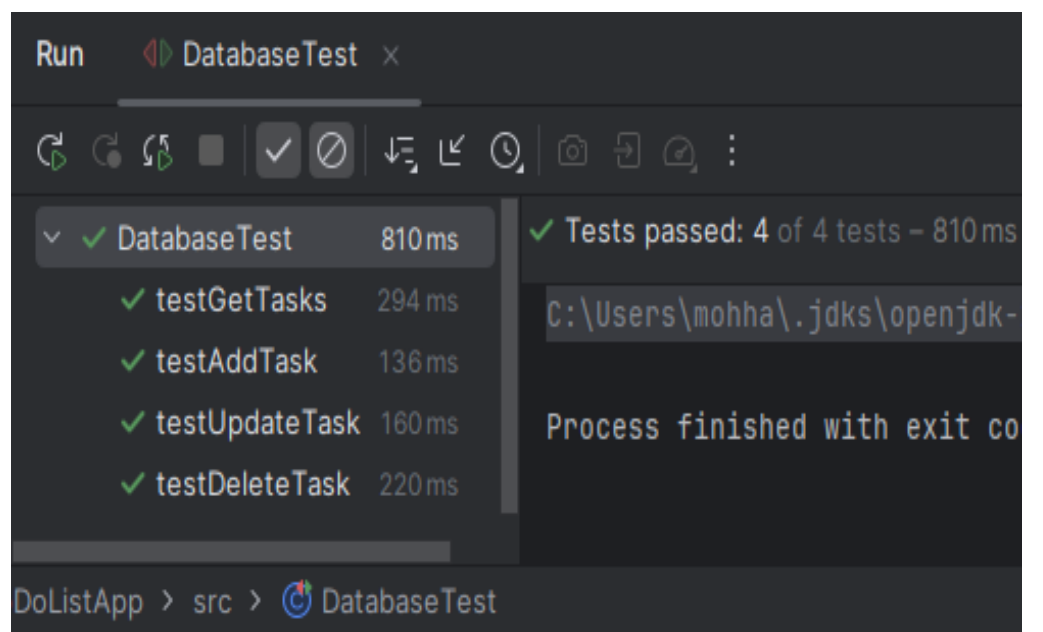
3. Executing JUnit Tests:

- The DatabaseTest and TestApp classes contain JUnit tests to validate the application's functionality.
- In IntelliJ IDEA, right-click on the test class and select "Run" to execute the tests.
- The following results will be present Junit testing for the respective java test classes:

TestApp/TaskTest class:

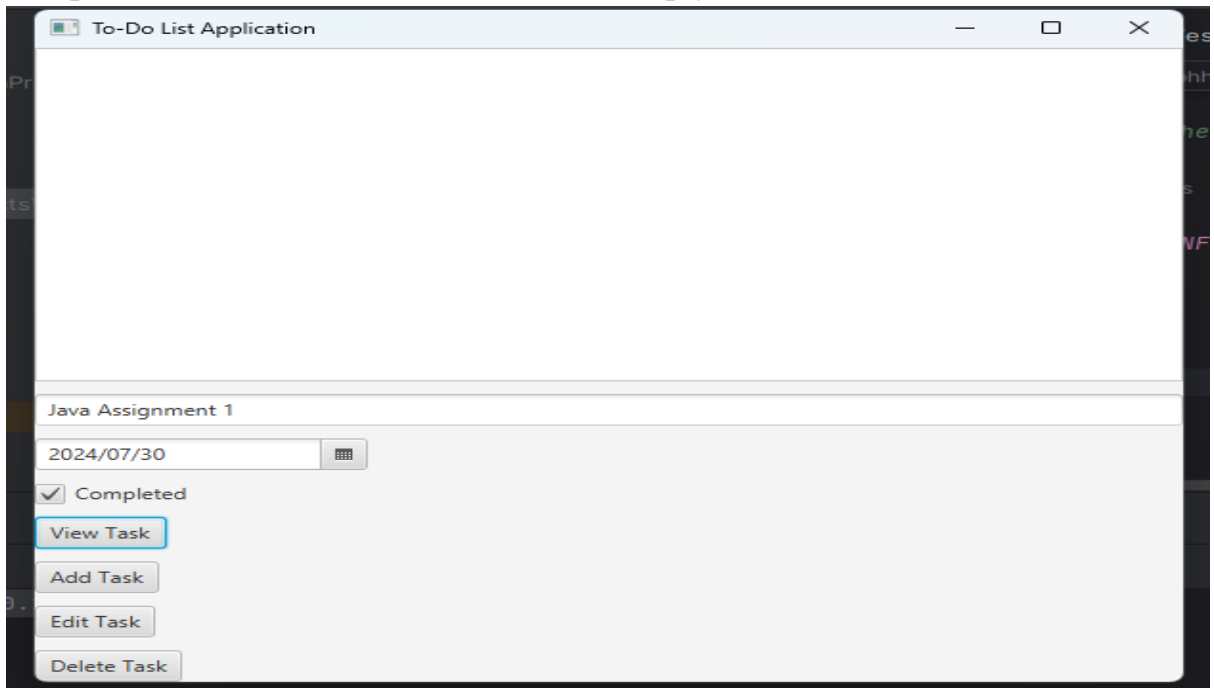


DatabaseTest class:



Functionalities

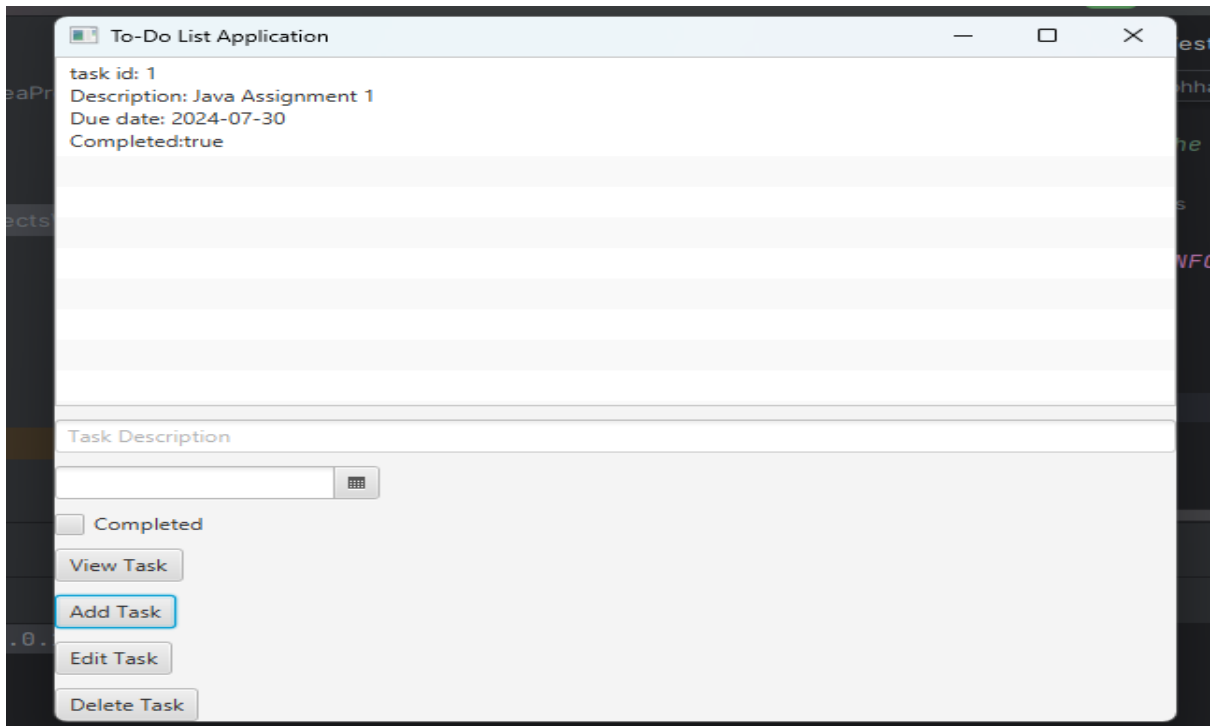
- **Add task:** Users can add a new task with a description, due date, and completion status. The database starts as empty at start:



The screenshot shows a web application window titled "To-Do List Application". The main area is empty. At the bottom, there is a form with the following fields and controls:

- A text input field containing "Java Assignment 1".
- A date input field containing "2024/07/30" with a calendar icon.
- A checkbox labeled "Completed" which is checked.
- A button labeled "View Task" (highlighted with a blue border).
- A button labeled "Add Task".
- A button labeled "Edit Task".
- A button labeled "Delete Task".

Result of a successful task addition:



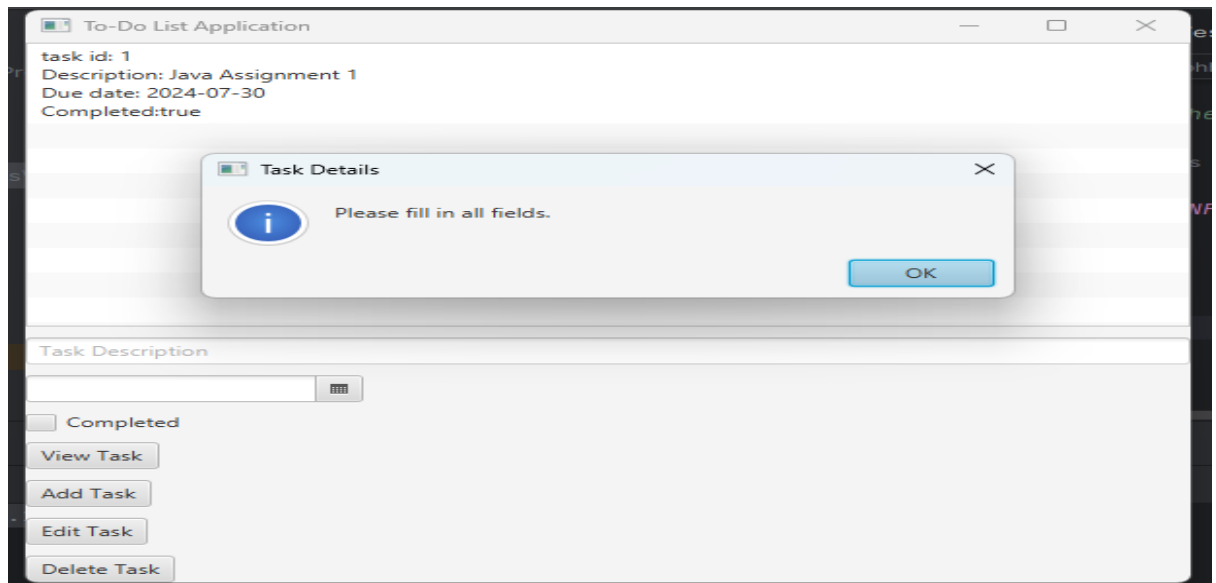
The screenshot shows the same "To-Do List Application" window. The main area now displays the details of the added task:

```
task id: 1
Description: Java Assignment 1
Due date: 2024-07-30
Completed:true
```

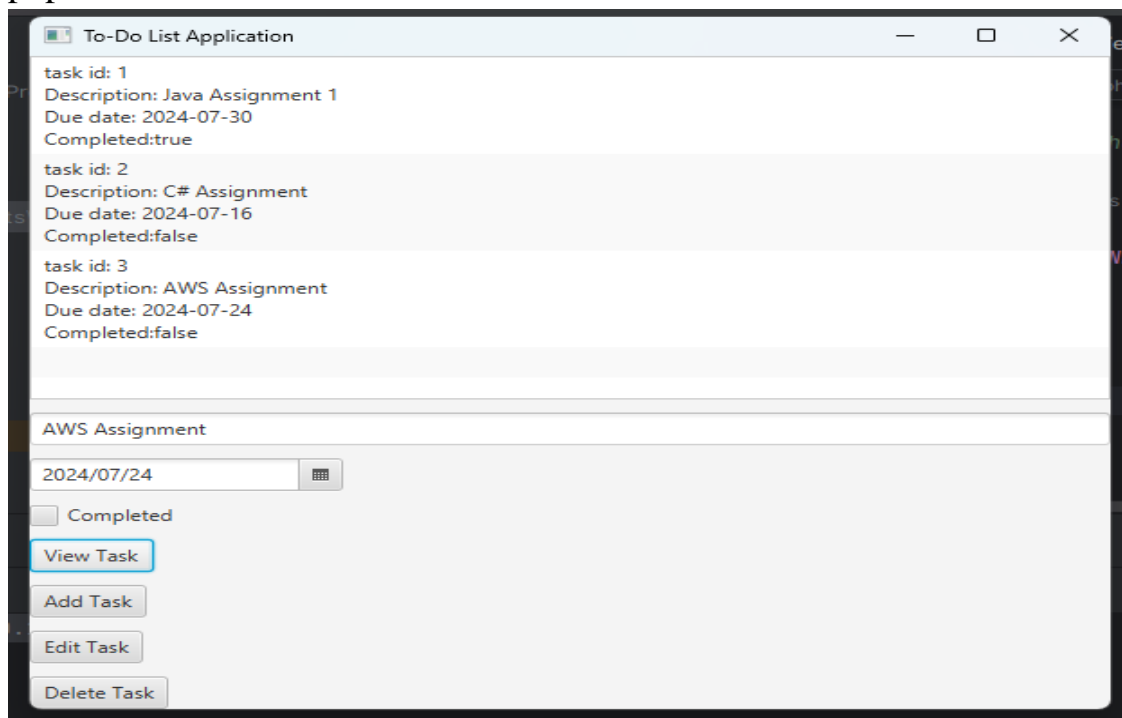
Below this information is a table with 5 empty rows. At the bottom, the form is reset with the following fields and controls:

- A text input field containing "Task Description".
- A date input field with a calendar icon.
- A checkbox labeled "Completed" which is unchecked.
- A button labeled "View Task".
- A button labeled "Add Task" (highlighted with a blue border).
- A button labeled "Edit Task".
- A button labeled "Delete Task".

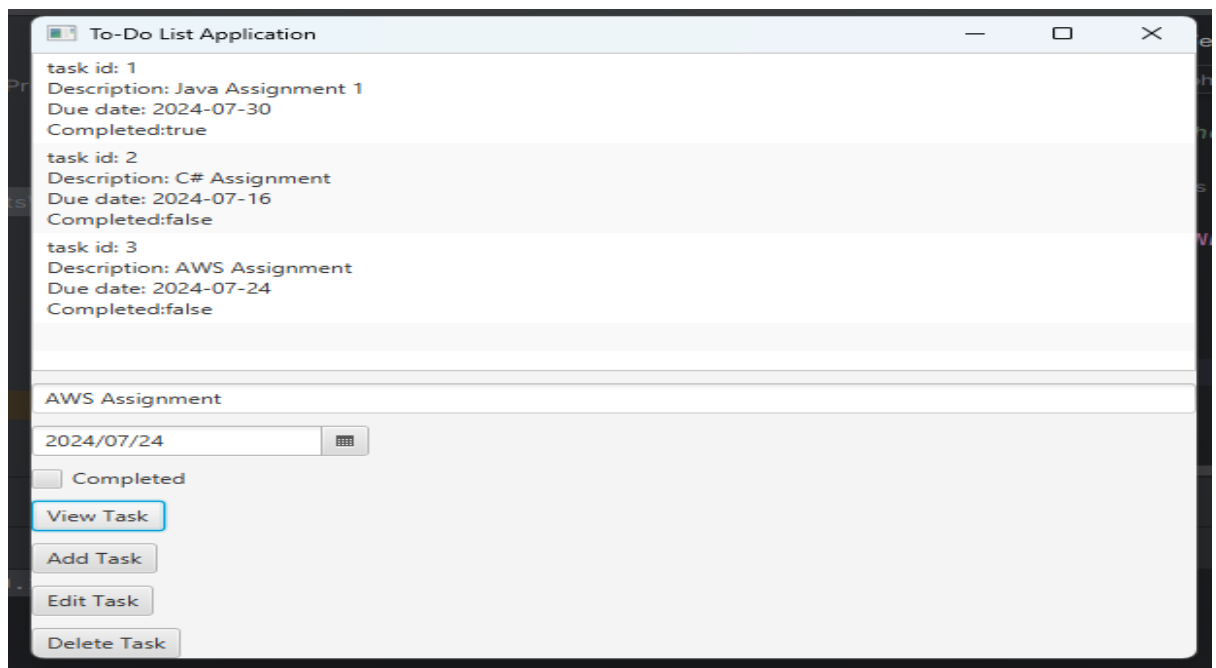
Result of an unsuccessful task addition due to empty input fields:



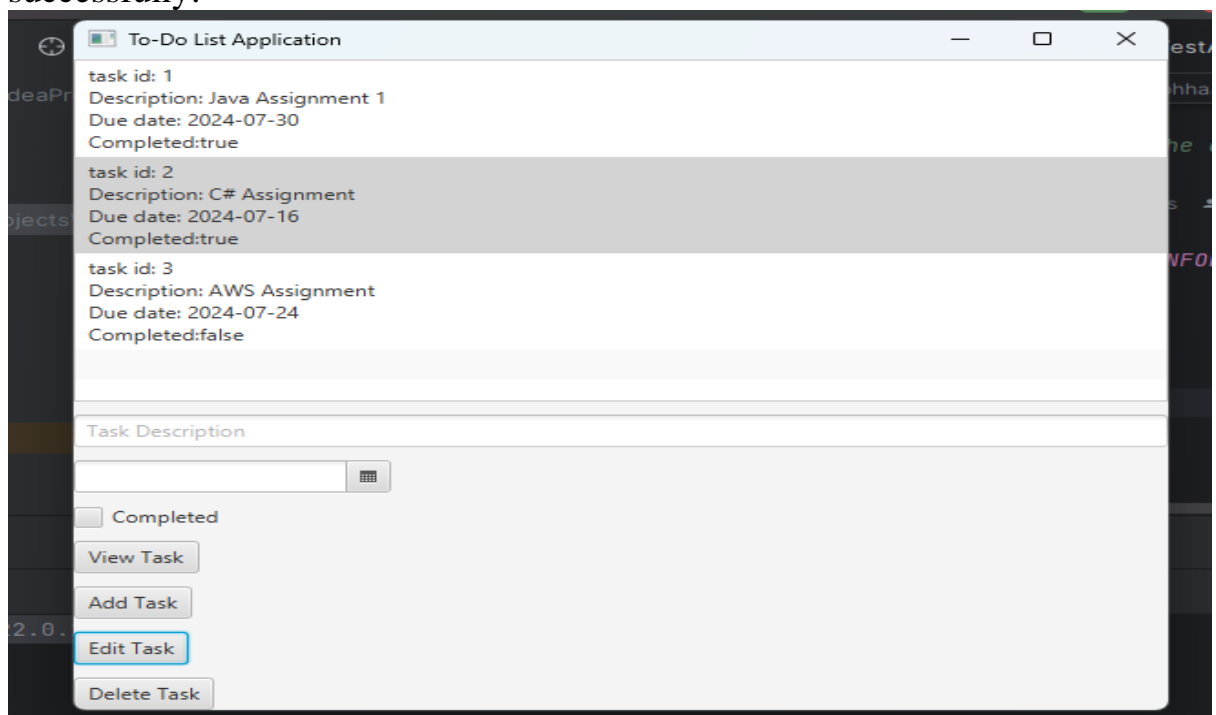
- **View task:** Users can view all tasks stored in the database. Result of populated tasks:



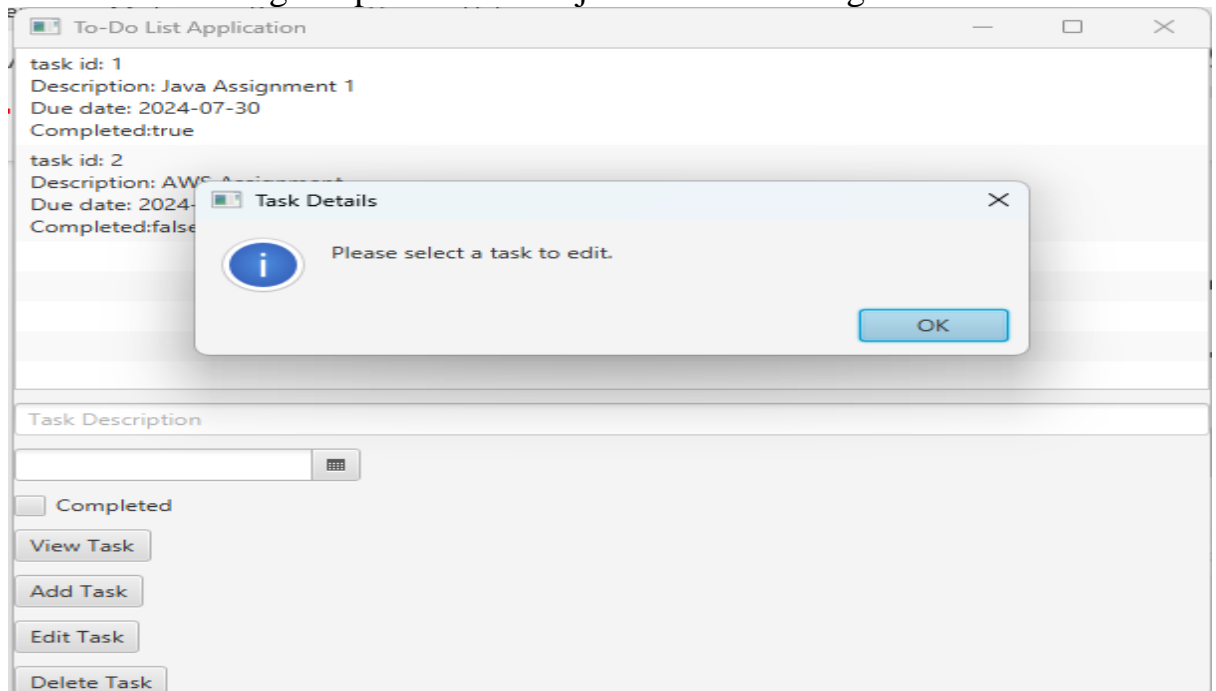
- **Update Task:** Users can update the description, due date, and completion status of an existing task.
Current results before updating:



Results after updating task object 2's completion status to true successfully:

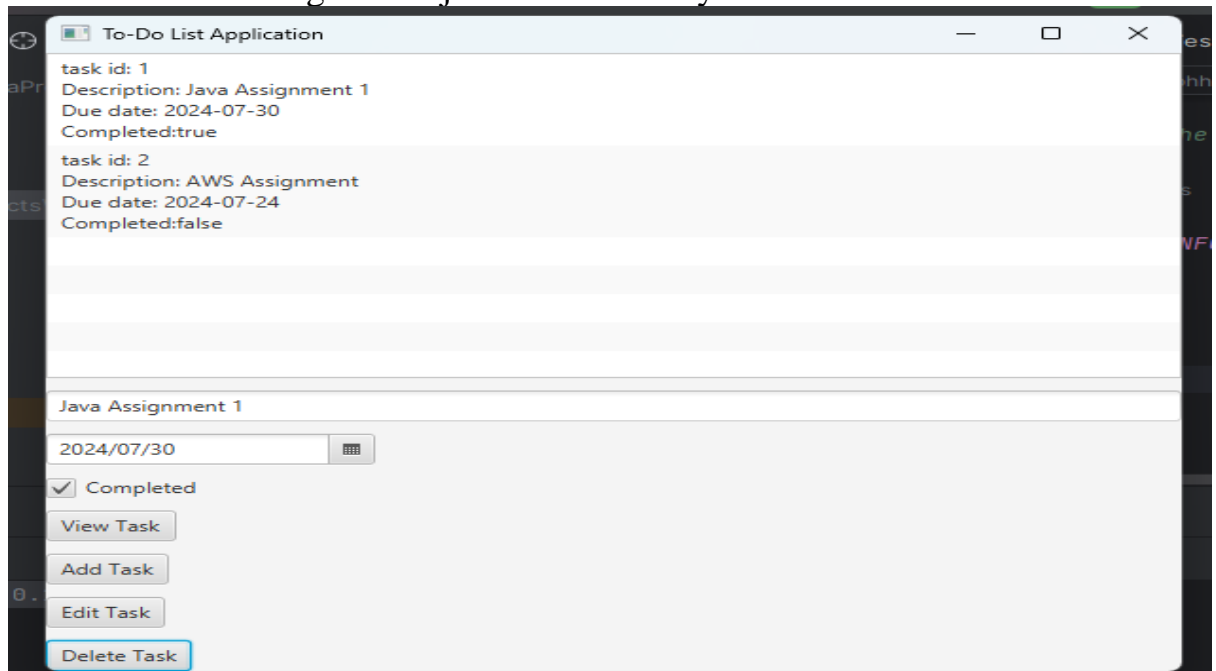


Results of wanting to update a task object without having selected it:

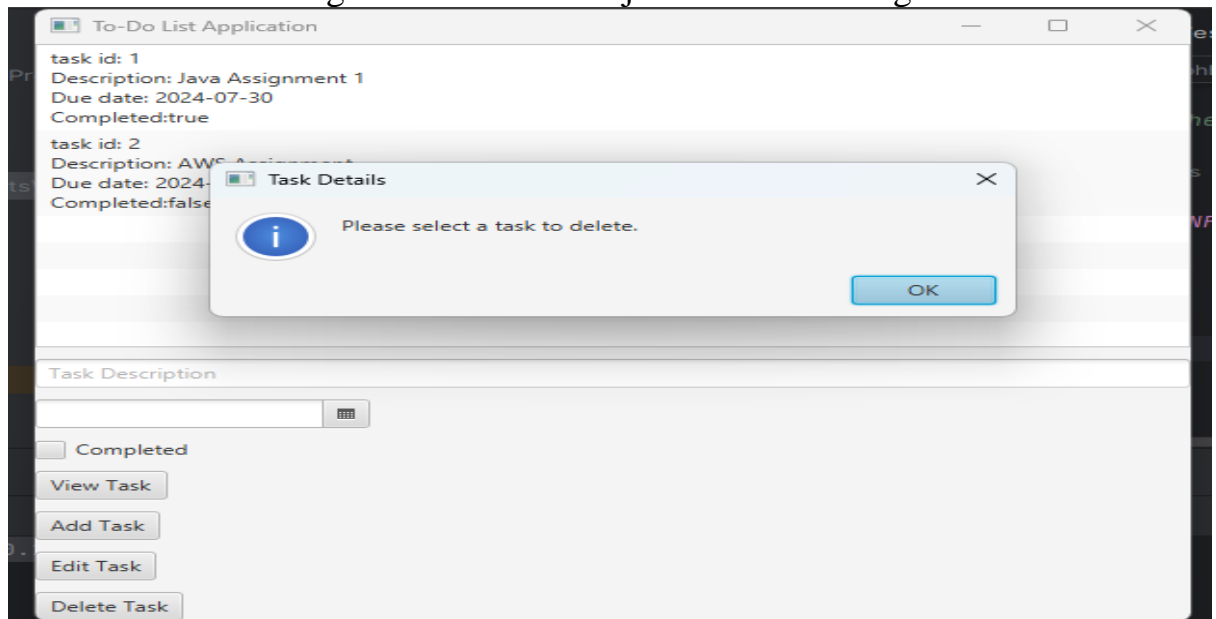


- **Delete Task:** Users can delete a task.

Results after deleting task object 2 successfully:



Results of not wanting to delete a task object without having selected it:



- **Auto-Increment ID:** Task IDs start from 1 and increment by 1. When a task is deleted, the IDs of remaining tasks are re-assigned to ensure they remain contiguous. Look at the delete task example as task object 3 becomes task object 2 after deleting the task object 2 that had the task id 2 at first.
- **Persistence:** All tasks are stored in a MySQL database.

Special instructions

- Ensure that the database connection is correctly configured in the Database class. The getConnection method should establish a connection to your database.
- Make sure that the database schema is set up correctly, with a table named tasks that has columns corresponding to the Task attributes (id, description, dueDate, completed).
- If running the application for the first time, you may need to initialize the database table. This can be done by calling a method that creates the table if it does not exist.
- Always handle SQL exceptions appropriately to avoid application crashes and ensure smooth operation.
- Regularly run the JUnit tests to verify that the application's functionality is working as expected.

Conclusion

In conclusion, the goal of this guide is to provide you a comprehensive grasp of how to use and navigate the features of the application. You ought to be able to handle your responsibilities more skilfully at this point, leading to improved productivity and organization using the guidelines that were supplied. I hope this application proves to be helpful and easy to use.