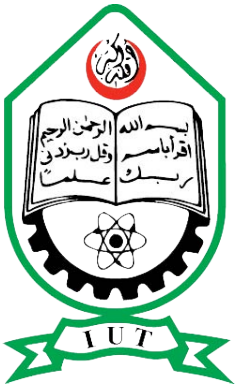
Islamic University of Technology



Visual Programming Lab

CSE 4402

TODO List

Author:
ahmed m. s. albreem 210041258
abdulmajeed Alqadami 210041272
Shuaibu Mustapha Musa 210041270

June 25, 2024

Contents

1 Introduction

The aim of the project: The main idea of the project is to build an easy to use javafx application that is created to manage and visualize data effectively. The application's goal is to solve the problem of organizing and analyzing the user's daily life data

Main functions and features:

Manage data by creating, deleting and modifying tasks.

User Interface: Intuitive and responsive design for easy navigation.

Database support for communication between us and users.

Pomodo timer to manage its task efficiently.

Task Calendar Every day has its own tasks if there are tasks on that day.

2 System Design

The task list system is created to work with three states namely create, update and delete which includes the MySQL database system which is embedded with the local database as host at this moment and every process is connected to the main database branch of the database The application actually contains the database, it contains some branches which One of them is to support the users in case there are any issues and also one is to backup the users data and it will be stored automatically with Java's io file system interaction for each process and the system has pomodo timer and also designed with calndar destribytion which every day has its tasks and data and the support part is The part that connects the user by textedit with btn when he will click on that he will send a request through the data to be stored and the role user data like send the user distribution request permission in the database owner and give it to him as a user he can just send it to this branch about the task branch there is a backup because The data is not necessary for us, we interact with the data that interests him at that time and we give him permission to edit, create and delete electronic data whenever he wants, he is allowed the data that he will provide + the data that he has at that moment + the information of the user information provider to give a successful creation request on the basis data.

TaskListSystem is a class that manages tasks. It has ways to create, update, and delete tasks, as well as handle support requests.

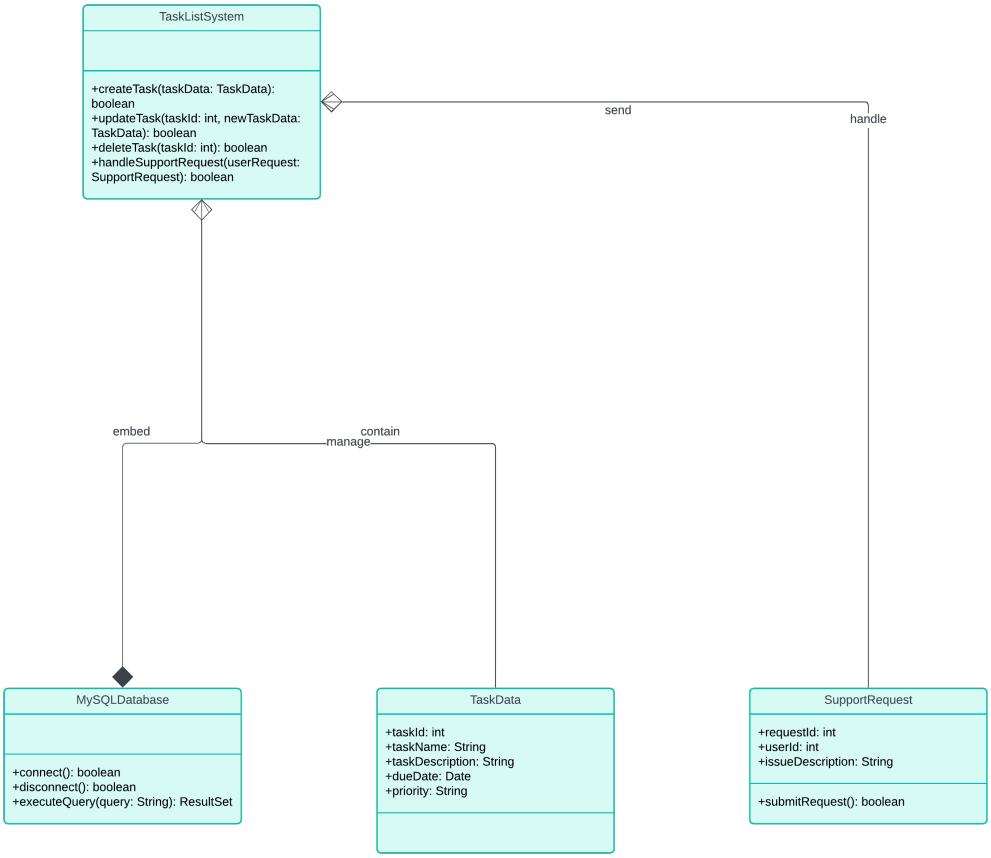


Figure 1: uml digram

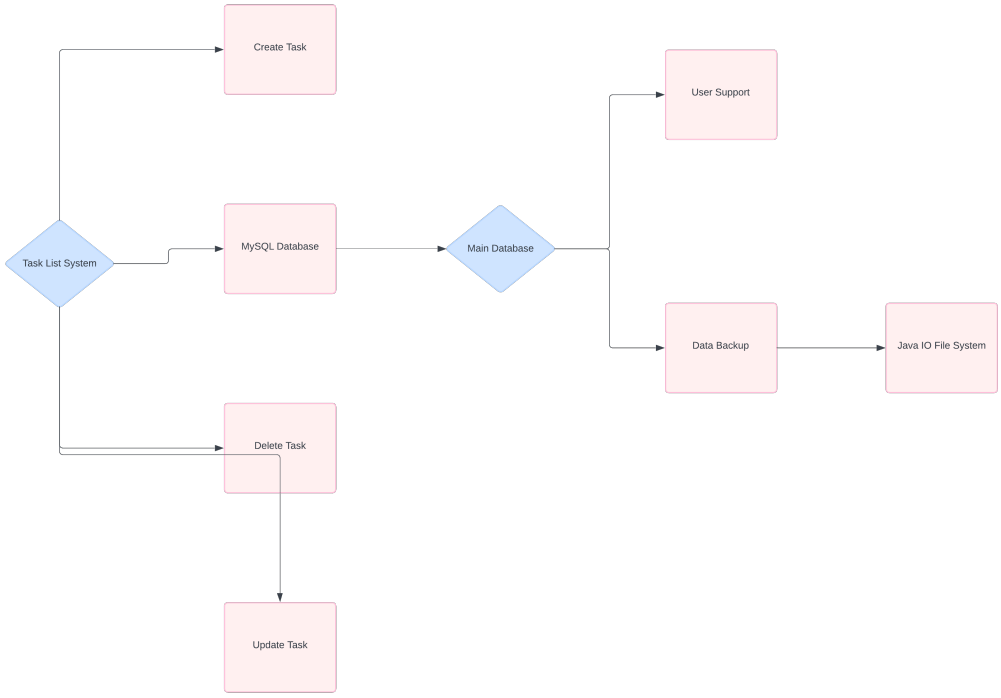


Figure 2: system architecture

TaskData is a class that stores information about a task. It contains attributes such as task ID, task name, task description, by date.

SupportRequest is a class that stores information about a support request.

the libraries is javafx and Scene Builder

3 Implementation Details

The backend logic handles data processing and storage, while the frontend provides an interactive interface for users.

above some of the implemention as code of the pervious part

above some of the functions of the sorting and searching which involve on the second child of the main tree of the operation procodure

The "creating task imp." code describes the creation of the task and how it will be stored as binary code inside the file that appends it every time it is called using this format. It will work with each call with a variable (con-

```
public boolean AppendToFileFiles() throws IOException,SQLException {
    String textToAppend = Date.toString() + " " + String.valueOf(id) + " " + String.valueOf(false) + " " + Task_title + " "  + Task_Details + "\n";
    Files.write(Paths.get(PATH), textToAppend.getBytes(), StandardOpenOption.APPEND);
    sort_add();
    return true;
}
```

Figure 3: creating task imp.

```
public void sort_tasks() throws IOException,SQLException {
    Path sourcePath = Paths.get("C:\\Users\\ahmed\\IdeaProjects\\todolist\\src\\main\\java\\com\\cse4404\\todolist\\tempFile.txt");
     if(sourcePath.toFile().exists()){
    Path destinationPath = Paths.get("C:\\Users\\ahmed\\IdeaProjects\\todolist\\src\\main\\java\\com\\cse4404\\todolist\\task.txt");
     try (InputStream inputStream = Files.newInputStream(sourcePath);
          OutputStream outputStream = Files.newOutputStream(destinationPath)) {

         int byteRead;
         while ((byteRead = inputStream.read()) != -1) {
             outputStream.write(byteRead);
         }
     } catch (IOException e) {
         e.printStackTrace();
     }


     ArrayList<String> lines = new ArrayList<>();
     Scanner scanner = new Scanner(new File(PATH));
     while (scanner.hasNextLine()) {
              lines.add(scanner.nextLine());
     }
             scanner.close();
             Collections.sort(lines);
             FileWriter writer = new FileWriter(PATH);
             for (String line : lines) {
                 writer.write(line + "\n");
             }
             writer.close();
             save_data(extract_text());}
     }
```

Figure 4: Sorting operation

tained)position that will be known and through the spacing between them it will be organized and known to the programmer to provide for structuring the program instructions for building the application.

"Sorting operation" of the code block focuses on updating the database and performing tasks which are organized by the date they are entered and also every time the operation is performed, it will be called at the top .

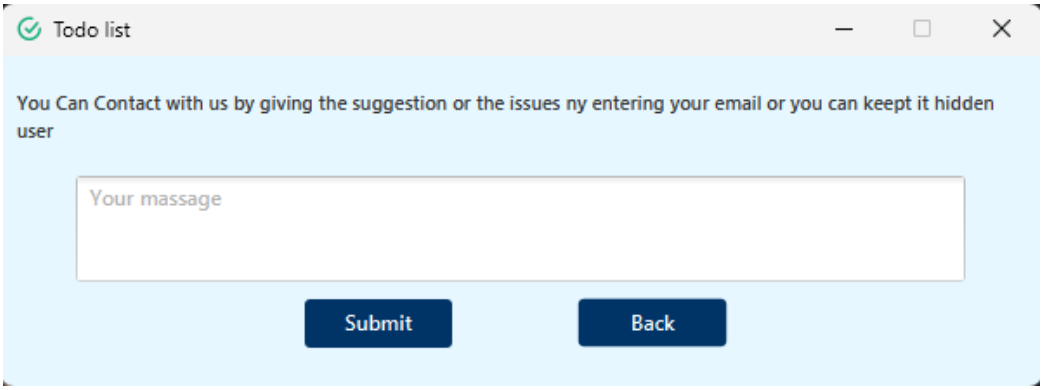


Figure 5: screenshot 1

4 GUI Design

User interface design: The UI follows Material Design principles such as design, focusing on the simplicity of the main display to make it more comfortable to use and interact with, which will increase the user's usage rate of the application, and the design is responsive. The layout is organized into different sections, including the top menu and the main content area such as list view and detail labels.

User interaction: Users interact with the application through different user interfaces and some of them are the same GUI components such as buttons, forms, and stickers.

Visual representation: There are some views that are the same as the view for the front end to save the storage of the source program as much as possible at first and there are some views for the main tasks and there are the same view for displaying the list by css file inside the resource style.css each style will be detailed there for further interaction by the front end And the user when clicking and moving from each view and others

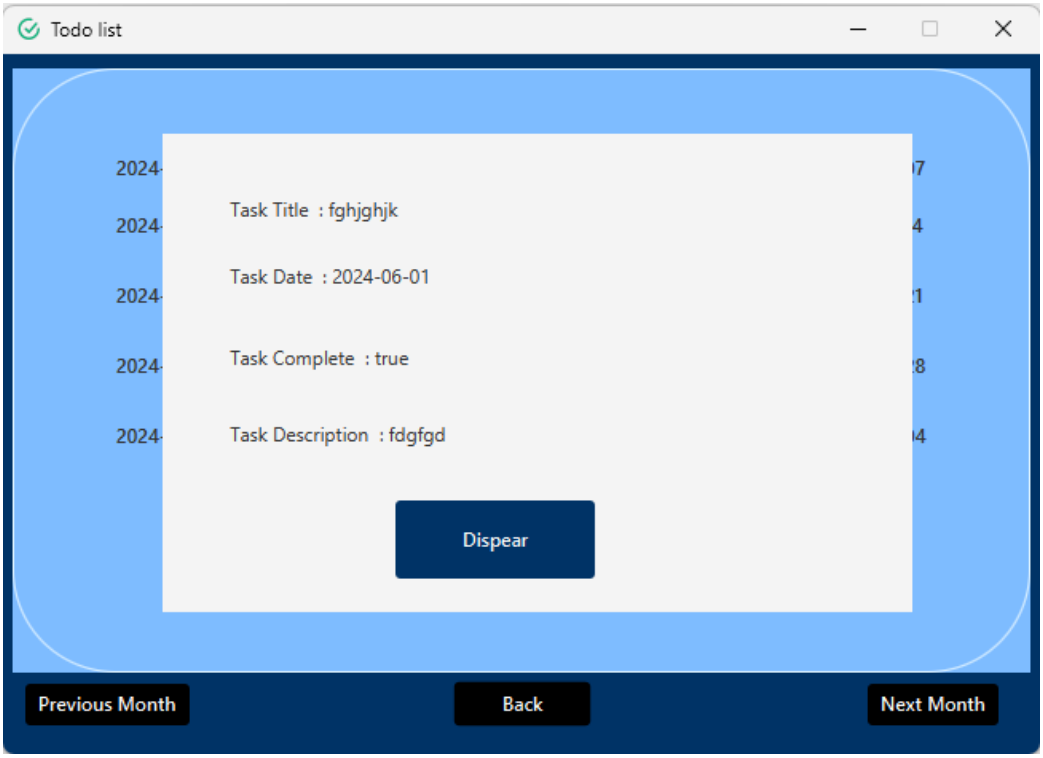


Figure 6: screenshot 2

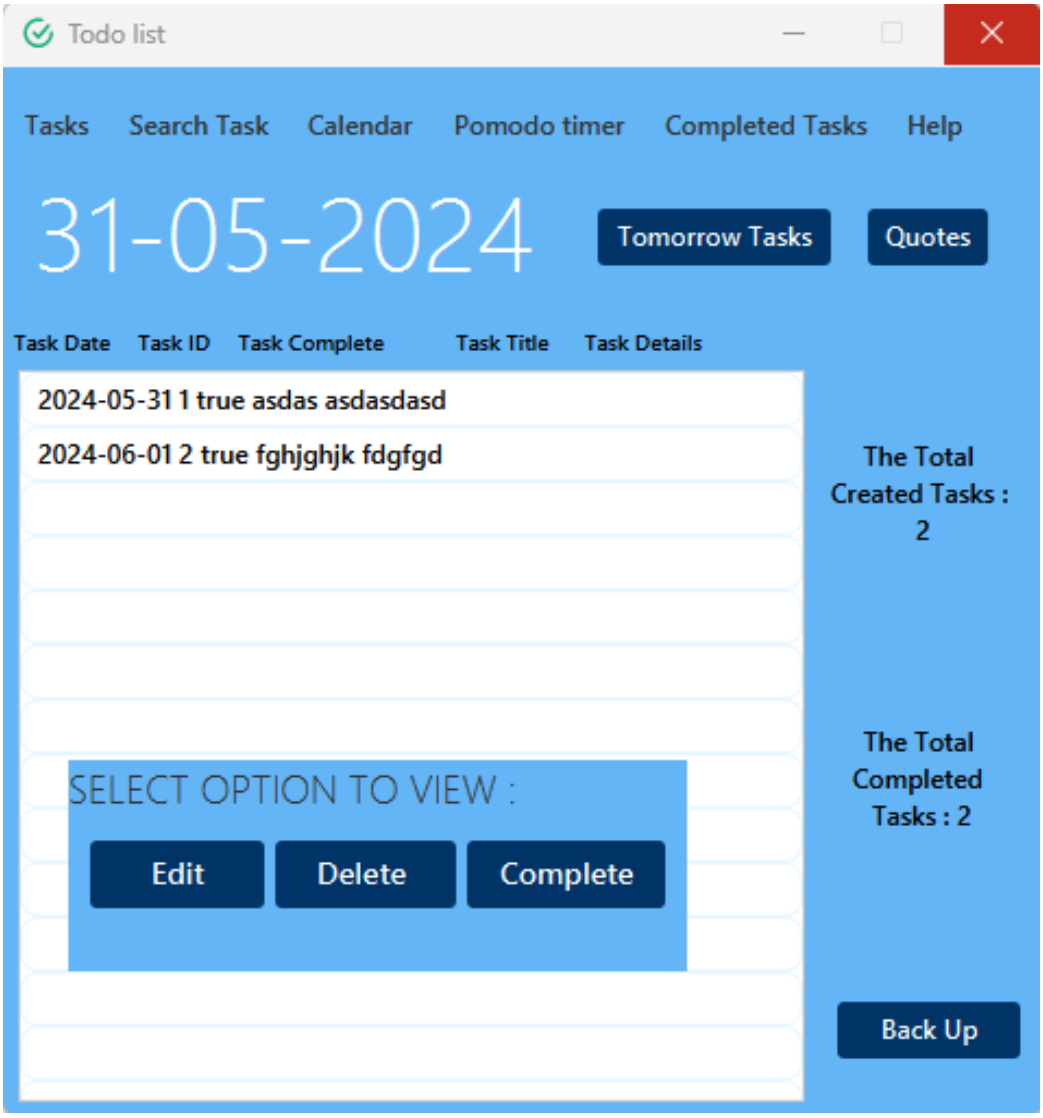


Figure 7: screenshot 3

```
.font-inin-main-page:hover
{
    -fx-font-family: 'Segoe UI Semibold';
    -fx-text-fill: white;
    -fx-underline: true;

}

.light-list-view .list-cell {
    -fx-font-family: 'Segoe UI Semibold';
    -fx-text-fill: #000000;
    -fx-background-color: #ffffff;
    -fx-border-style: solid;
    -fx-border-color:  #e6f7ff;
    -fx-border-radius : 2%;
    -fx-strikethrough: true;
}

.light-list-view .list-cell:hover {
    -fx-text-fill: #ffffff;
    -fx-background-color: #003366;
}
```

Figure 8: SC.BULIDER WITH CSS

5 Testing and Evaluation

this is part of the tasting which is testing for the random code block and the evaluate of the code by some certina of any programmer

```
    @FXML
public void editclick(ActionEvent event) throws IOException, SQLException {
    String taskId = task_id.getText().trim();
    String _a_ = new_data.getText().trim();
    String _b_ = "";
    if (taskId.isEmpty() || _a_.isEmpty()) {
        alter.setText("Please enter a Task ID and new data!");
        return;
    }
    Task task = new Task();
    String taskInfo = task.searchInfo(taskId);
    String editedTaskString = null;
    String[] parts = taskInfo.split(" ");
    String Date = parts[0];
    String id = parts[1];
    String comp = parts[2];
    String Task_title = parts[3];
    String Task_Details = parts[4];
    if (date_radio.isSelected()) {
        editedTaskString = buildEditedTaskString
        (_a_, id,comp, Task_title, Task_Details);
    } else if (title_radio.isSelected()) {
        editedTaskString = buildEditedTaskString
        (Date, id,comp, _a_, Task_Details);
    } else if (details_radio.isSelected()) {
        editedTaskString = buildEditedTaskString
        (Date, id,comp, Task_title, _a_);
    } else if (complete_radio1.isSelected()) {
        _b_ = _a_.toLowerCase();
        if(_b_.equals("true")||_b_.equals("false")){
            editedTaskString = buildEditedTaskString
            (Date, id,_b_, Task_title,Task_Details);
```

```
            }

            else{
                alter.setText("Check your Entered Data to Be Eddited !!");
                return;
            }
        }
        if (editedTaskString != null &&
        task.editTask(taskInfo,editedTaskString)) {
            onbackclick(event);
        } else {
            alter.setText("An error occurred while editing the task!");
        }
    }

    private String buildEditedTaskString
    (String date, String id, String comp, String title, String details) {
        return date + " " + id + " " + comp + " " + title + " " + details;
    }
```

These are some of the researched areas that will be developed, such as the sample I present above a few lines earlier. After research, I find that there are some good points of strength and areas for improvement.

strength point (the testing strategies): Checks the input that does not take the task ID itself and is not empty from the GUI before performing any operation in the specified process. Handling Radio Btns for each selection, each of them if one is called the other is empty. Error handling in input and MySQL issues. Separation of Concerns The buildEditedTaskString function keeps the task thread construction separate from the editing logic.

Areas for improvement (potential bugs):

Name variables for the function to improve readability The error message for edit failure can be more specific depending on the situation. While announcing throws, consider using catch attempt blocks to handle errors in a cleaner manner.

The project doesn't need any unit testing or tools, we just used intilj idea to do it with initial university account There were no challenges, the project was very good and simple at our fingertips, we did it with our simple and

good skills according to our schedule which we had broken down before.

6 Conclusion

the main key achievements of our project : Documentation project, User Support,Functionality for the tasks ,User Interface Clean and Responsive Design.
If he has more time, we redesign the project to make the GUI better and make the function better and the code also not good (like group skill) and add some function for main purpose and some better implementation daily life of user
The program worked well in the end in the way expected from our vision as students but we did not reach the teacher's vision to develop it in this way but the main thing is "Thanks to our teacher for his efforts and it was a good class to learn new skills together".

References

main referance of the program desgined by three tools which is the
CSS by https://developer.mozilla.org/en-US/docs/Web/CSS
JavaFX by https://openjfx.io
Scene Builder by https://gluonhq.com/products/scene-builder/