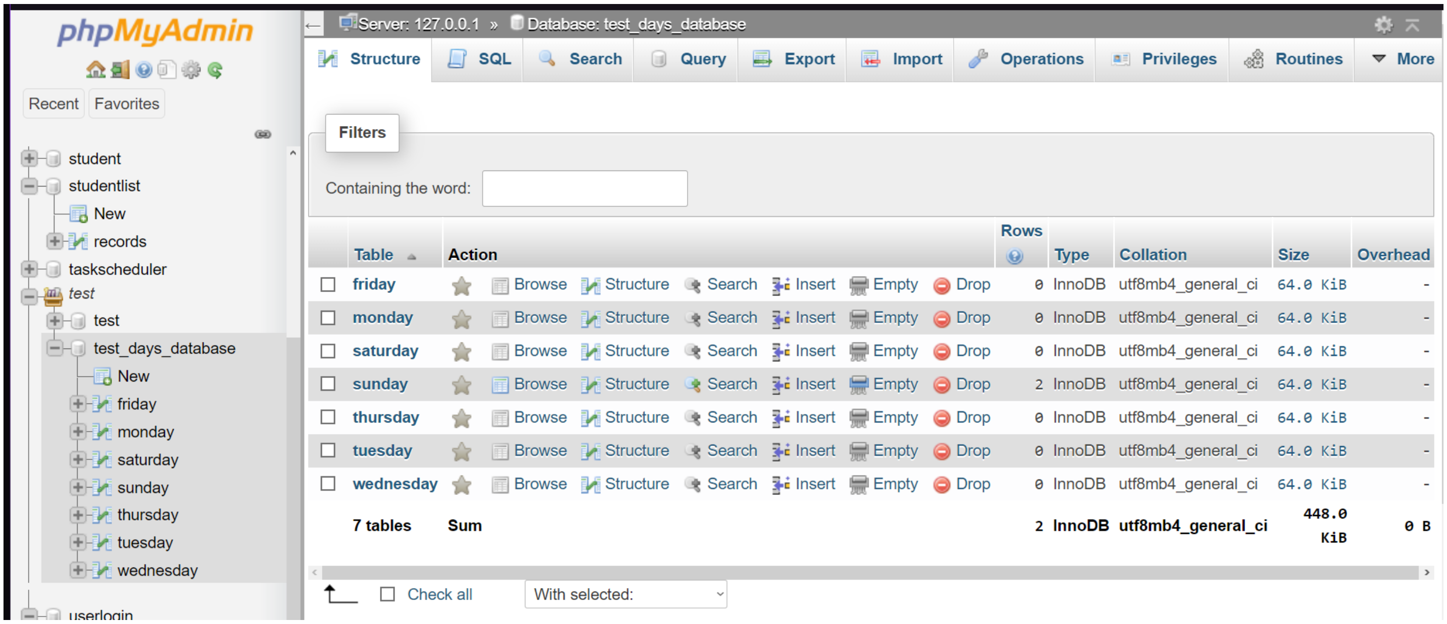
PROJECT NAME:

DAILY TASK SCHEDULER

PROJECT ABSTRACT:

This Application is basically a GUI version of TOLIST which was previously in terminal mode with some additional features. It has CRUD operations which are done using SQL database. A Pomodoro timer also included in this application to enhance the productivity to track the time on a specific task by the user. This Task Scheduler schedules the task according to the week days in accordance with the system date and time .

DATABASE:



Containing the word:

Table	Action						Rows	Type	Collation	Size	Overhead
☐ friday	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
☐ monday	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
☐ saturday	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
☐ sunday	Browse	Structure	Search	Insert	Empty	Drop	2	InnoDB	utf8mb4_general_ci	64.0 KiB	-
☐ thursday	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
☐ tuesday	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
☐ wednesday	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
7 tables	Sum						2	InnoDB	utf8mb4_general_ci	448.0 KiB	0 B

Test_days_database

Sunday,Monday,Tuesday,Wednesday,Thursday ,Friday,Saturday,sunday TABLE:

These Tables are used to store the task that are added by the user for each week days accordingly.

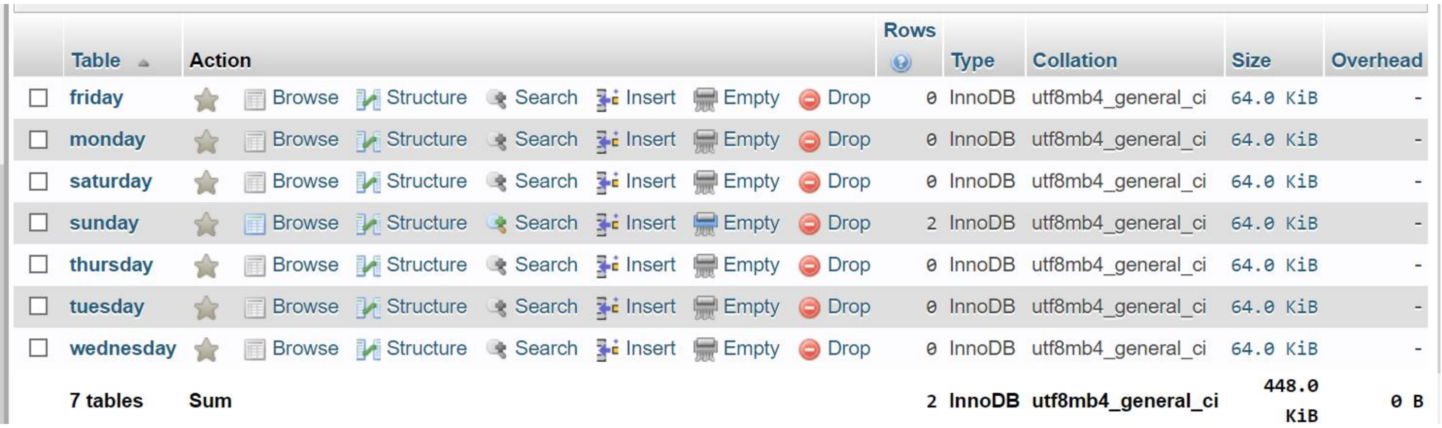
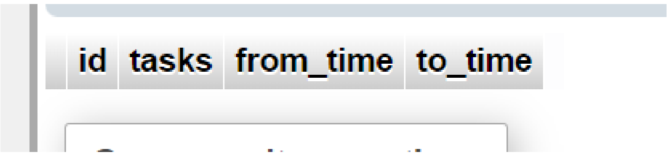


	Table	Action							Rows	Type	Collation	Size	Overhead
□	friday	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
□	monday	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
□	saturday	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
□	sunday	☆	Browse	Structure	Search	Insert	Empty	Drop	2	InnoDB	utf8mb4_general_ci	64.0 KiB	-
□	thursday	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
□	tuesday	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
□	wednesday	☆	Browse	Structure	Search	Insert	Empty	Drop	0	InnoDB	utf8mb4_general_ci	64.0 KiB	-
	7 tables	Sum							2	InnoDB	utf8mb4_general_ci	448.0 KiB	0 B



id	tasks	from_time	to_time

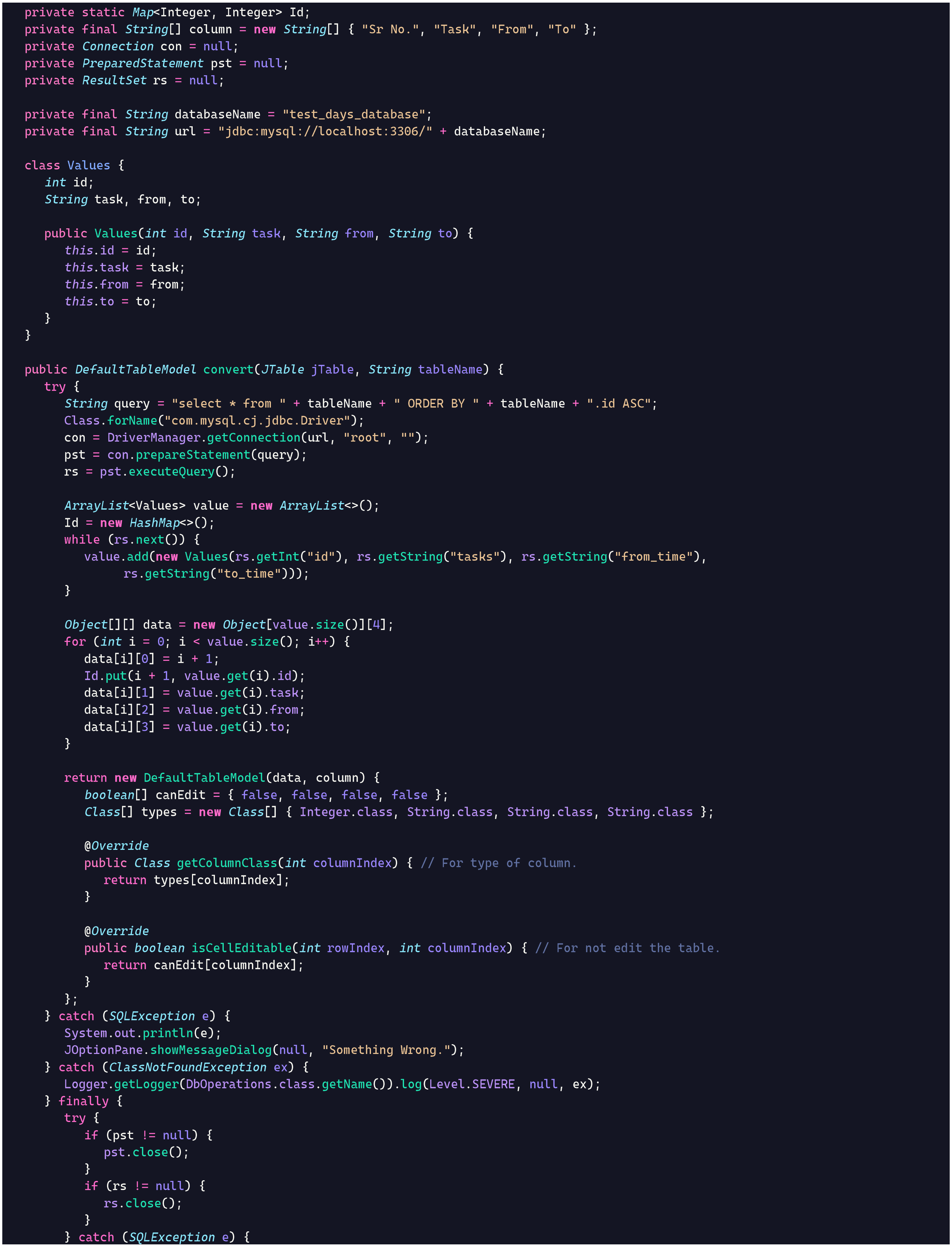
PSEUDO CODE:

DailyTaskScheduler.JAVA :

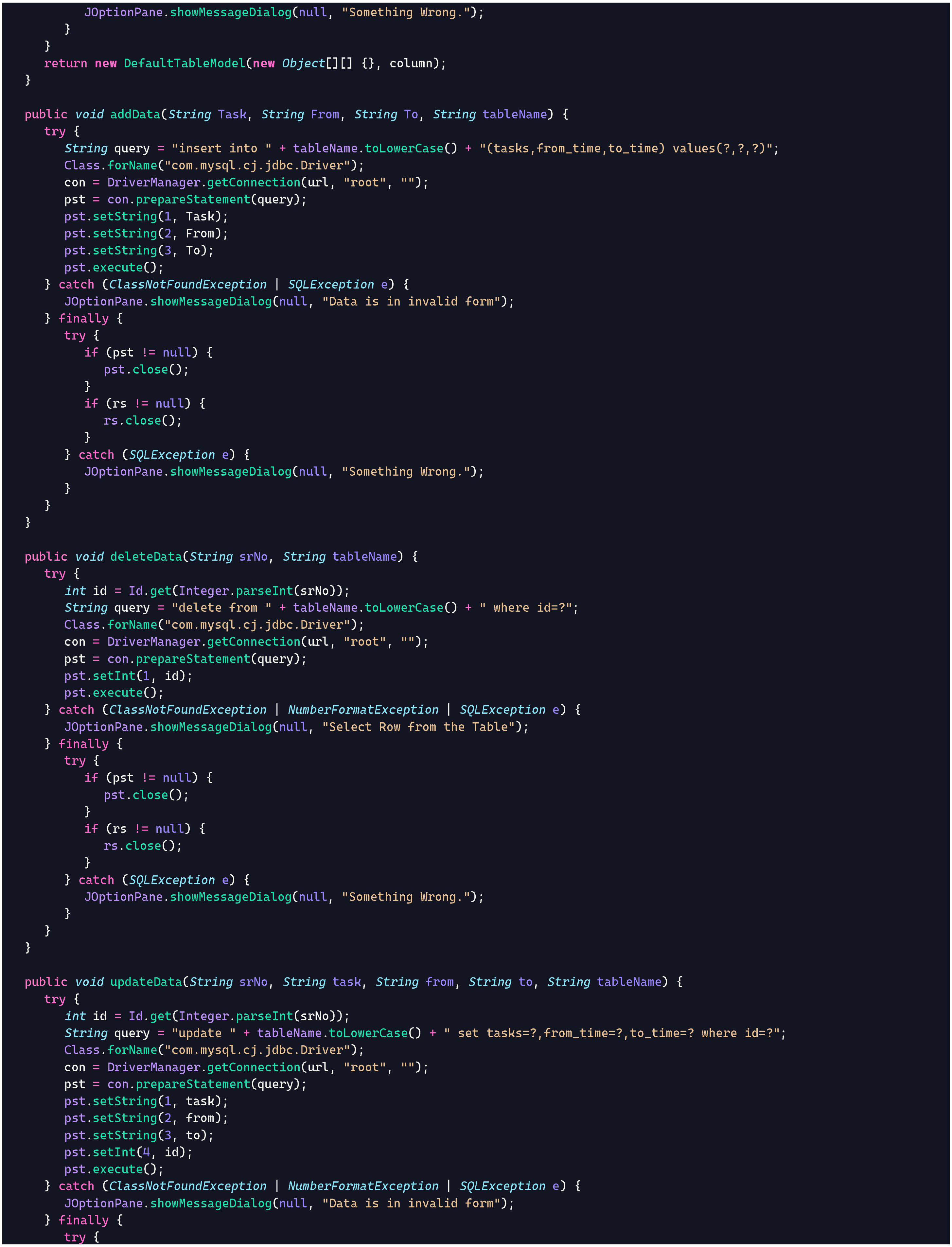
```
package daily.task.scheduler;

public class DailyTaskScheduler {

   public static void main(String[] args) {
       new HomePage().setVisible(true);
   }

}
```

DbOperations.JAVA:

```
/*
For Creating the table
CREATE TABLE `test_days_database`.`sunday` ( `id` INT NOT NULL AUTO_INCREMENT , `tasks` VARCHAR(30) NOT NULL , `from_time`
VARCHAR(5) NOT NULL , `to_time` VARCHAR(5) NOT NULL , PRIMARY KEY (`id`), UNIQUE (`tasks`, `from_time`, `to_time`)) ENGINE =
InnoDB;
*/

package daily.task.scheduler;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;

public class DbOperations {
```



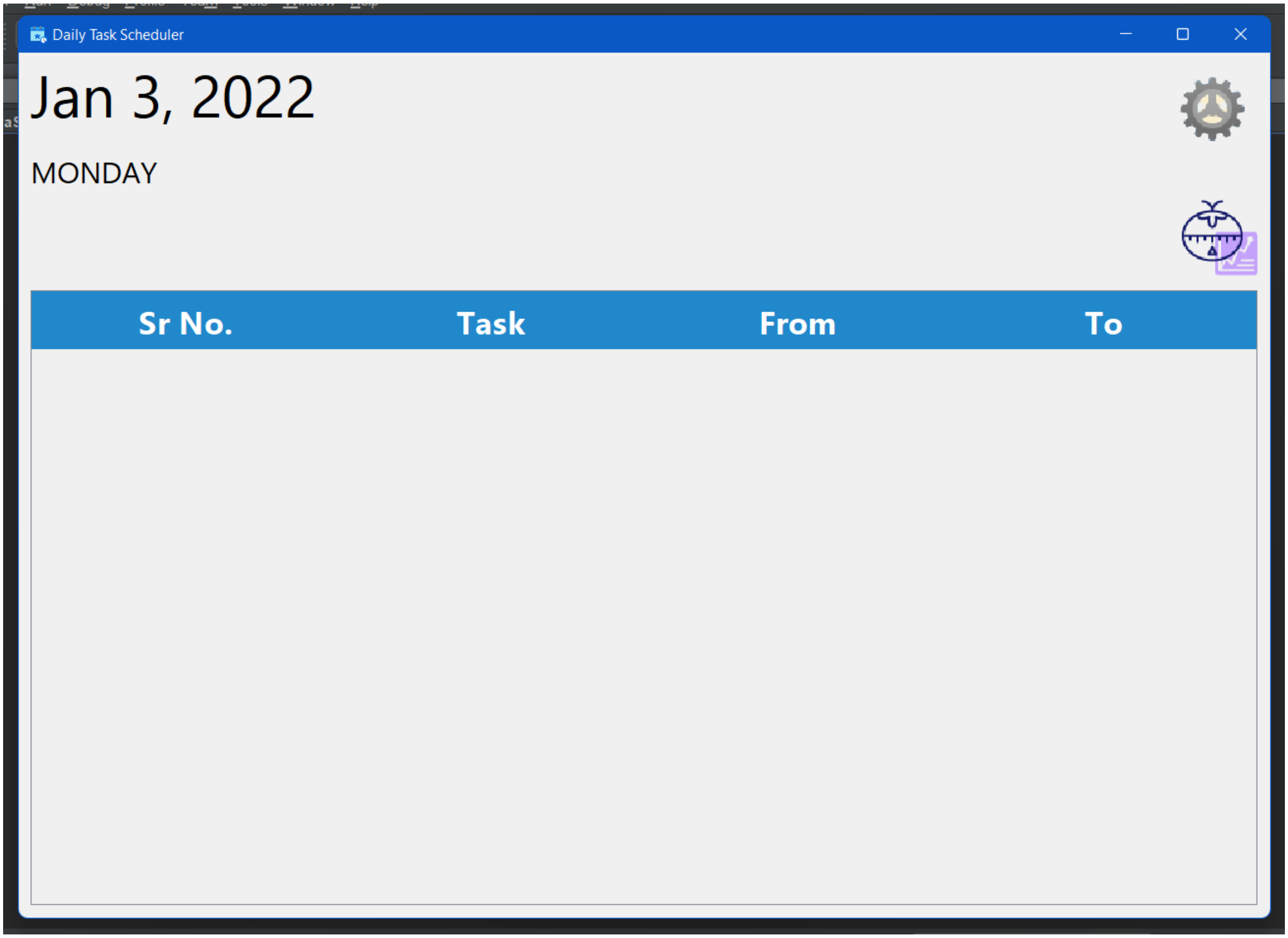
```
   private static Map<Integer, Integer> Id;
   private final String[] column = new String[] { "Sr No.", "Task", "From", "To" };
   private Connection con = null;
   private PreparedStatement pst = null;
   private ResultSet rs = null;

   private final String databaseName = "test_days_database";
   private final String url = "jdbc:mysql://localhost:3306/" + databaseName;

   class Values {
      int id;
      String task, from, to;

      public Values(int id, String task, String from, String to) {
         this.id = id;
         this.task = task;
         this.from = from;
         this.to = to;
      }
   }

   public DefaultTableModel convert(JTable jTable, String tableName) {
      try {
         String query = "select * from " + tableName + " ORDER BY " + tableName + ".id ASC";
         Class.forName("com.mysql.cj.jdbc.Driver");
         con = DriverManager.getConnection(url, "root", "");
         pst = con.prepareStatement(query);
         rs = pst.executeQuery();

         ArrayList<Values> value = new ArrayList<>();
         Id = new HashMap<>();
         while (rs.next()) {
            value.add(new Values(rs.getInt("id"), rs.getString("tasks"), rs.getString("from_time"),
                  rs.getString("to_time")));
         }

         Object[][] data = new Object[value.size()][4];
         for (int i = 0; i < value.size(); i++) {
            data[i][0] = i + 1;
            Id.put(i + 1, value.get(i).id);
            data[i][1] = value.get(i).task;
            data[i][2] = value.get(i).from;
            data[i][3] = value.get(i).to;
         }

         return new DefaultTableModel(data, column) {
            boolean[] canEdit = { false, false, false, false };
            Class[] types = new Class[] { Integer.class, String.class, String.class, String.class };

            @Override
            public Class getColumnClass(int columnIndex) { // For type of column.
               return types[columnIndex];
            }

            @Override
            public boolean isCellEditable(int rowIndex, int columnIndex) { // For not edit the table.
               return canEdit[columnIndex];
            }
         };
      } catch (SQLException e) {
         System.out.println(e);
         JOptionPane.showMessageDialog(null, "Something Wrong.");
      } catch (ClassNotFoundException ex) {
         Logger.getLogger(DbOperations.class.getName()).log(Level.SEVERE, null, ex);
      } finally {
         try {
            if (pst != null) {
               pst.close();
            }
            if (rs != null) {
               rs.close();
            }
         } catch (SQLException e) {
```



```
            JOptionPane.showMessageDialog(null, "Something Wrong.");
         }
      }
      return new DefaultTableModel(new Object[][] {}, column);
   }

   public void addData(String Task, String From, String To, String tableName) {
      try {
         String query = "insert into " + tableName.toLowerCase() + "(tasks,from_time,to_time) values(?,?,?)";
         Class.forName("com.mysql.cj.jdbc.Driver");
         con = DriverManager.getConnection(url, "root", "");
         pst = con.prepareStatement(query);
         pst.setString(1, Task);
         pst.setString(2, From);
         pst.setString(3, To);
         pst.execute();
      } catch (ClassNotFoundException | SQLException e) {
         JOptionPane.showMessageDialog(null, "Data is in invalid form");
      } finally {
         try {
            if (pst != null) {
               pst.close();
            }
            if (rs != null) {
               rs.close();
            }
         } catch (SQLException e) {
            JOptionPane.showMessageDialog(null, "Something Wrong.");
         }
      }
   }

   public void deleteData(String srNo, String tableName) {
      try {
         int id = Id.get(Integer.parseInt(srNo));
         String query = "delete from " + tableName.toLowerCase() + " where id=?";
         Class.forName("com.mysql.cj.jdbc.Driver");
         con = DriverManager.getConnection(url, "root", "");
         pst = con.prepareStatement(query);
         pst.setInt(1, id);
         pst.execute();
      } catch (ClassNotFoundException | NumberFormatException | SQLException e) {
         JOptionPane.showMessageDialog(null, "Select Row from the Table");
      } finally {
         try {
            if (pst != null) {
               pst.close();
            }
            if (rs != null) {
               rs.close();
            }
         } catch (SQLException e) {
            JOptionPane.showMessageDialog(null, "Something Wrong.");
         }
      }
   }

   public void updateData(String srNo, String task, String from, String to, String tableName) {
      try {
         int id = Id.get(Integer.parseInt(srNo));
         String query = "update " + tableName.toLowerCase() + " set tasks=?,from_time=?,to_time=? where id=?";
         Class.forName("com.mysql.cj.jdbc.Driver");
         con = DriverManager.getConnection(url, "root", "");
         pst = con.prepareStatement(query);
         pst.setString(1, task);
         pst.setString(2, from);
         pst.setString(3, to);
         pst.setInt(4, id);
         pst.execute();
      } catch (ClassNotFoundException | NumberFormatException | SQLException e) {
         JOptionPane.showMessageDialog(null, "Data is in invalid form");
      } finally {
         try {
            if (pst != null) {
               pst.close();
            }
            if (rs != null) {
               rs.close();
            }
         } catch (SQLException e) {
            JOptionPane.showMessageDialog(null, "Something Wrong.");
         }
      }
   }

   public void deleteRowsFromTable(String tableName) {
      try {
         String query = "delete from " + tableName;
         Class.forName("com.mysql.cj.jdbc.Driver");
         con = DriverManager.getConnection(url, "root", "");
         pst = con.prepareStatement(query);
         pst.execute();
      } catch (ClassNotFoundException | SQLException e) {
         JOptionPane.showMessageDialog(null, "Connection Error");
      } finally {
         try {
            if (pst != null) {
               pst.close();
            }
            if (rs != null) {
               rs.close();
            }
         } catch (SQLException e) {
            JOptionPane.showMessageDialog(null, "Something Wrong.");
         }
      }
   }

   public void deleteAllTableData() {
      try {
         String query = "delete from sunday";
         Class.forName("com.mysql.cj.jdbc.Driver");
         con = DriverManager.getConnection(url, "root", "");
         pst = con.prepareStatement(query);
         pst.execute();
         query = "delete from monday";
         pst = con.prepareStatement(query);
         pst.execute();
         query = "delete from tuesday";
         pst = con.prepareStatement(query);
         pst.execute();
         query = "delete from wednesday";
         pst = con.prepareStatement(query);
         pst.execute();
         query = "delete from thursday";
         pst = con.prepareStatement(query);
         pst.execute();
         query = "delete from friday";
         pst = con.prepareStatement(query);
         pst.execute();
         query = "delete from saturday";
         pst = con.prepareStatement(query);
         pst.execute();
      } catch (ClassNotFoundException | SQLException e) {
         JOptionPane.showMessageDialog(null, "Connection Error");
      } finally {
         try {
            if (pst != null) {
               pst.close();
            }
            if (rs != null) {
               rs.close();
            }
         } catch (SQLException e) {
            JOptionPane.showMessageDialog(null, "Something Wrong.");
         }
```

```
        }
    }
}
```

OUTPUT:

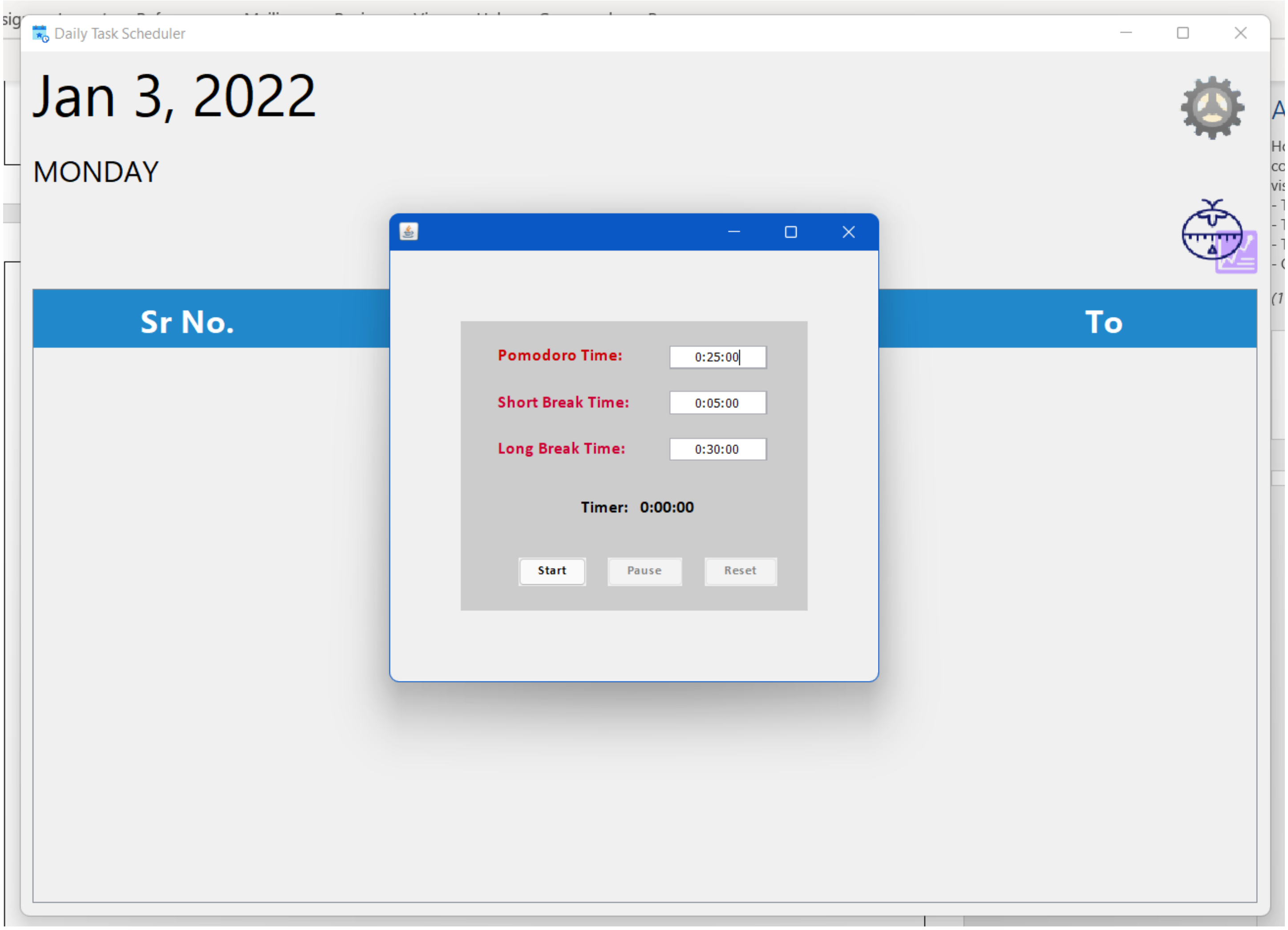


Daily Task Scheduler

Jan 3, 2022

MONDAY

Sr No.	Task	From	To



Jan 3, 2022

MONDAY

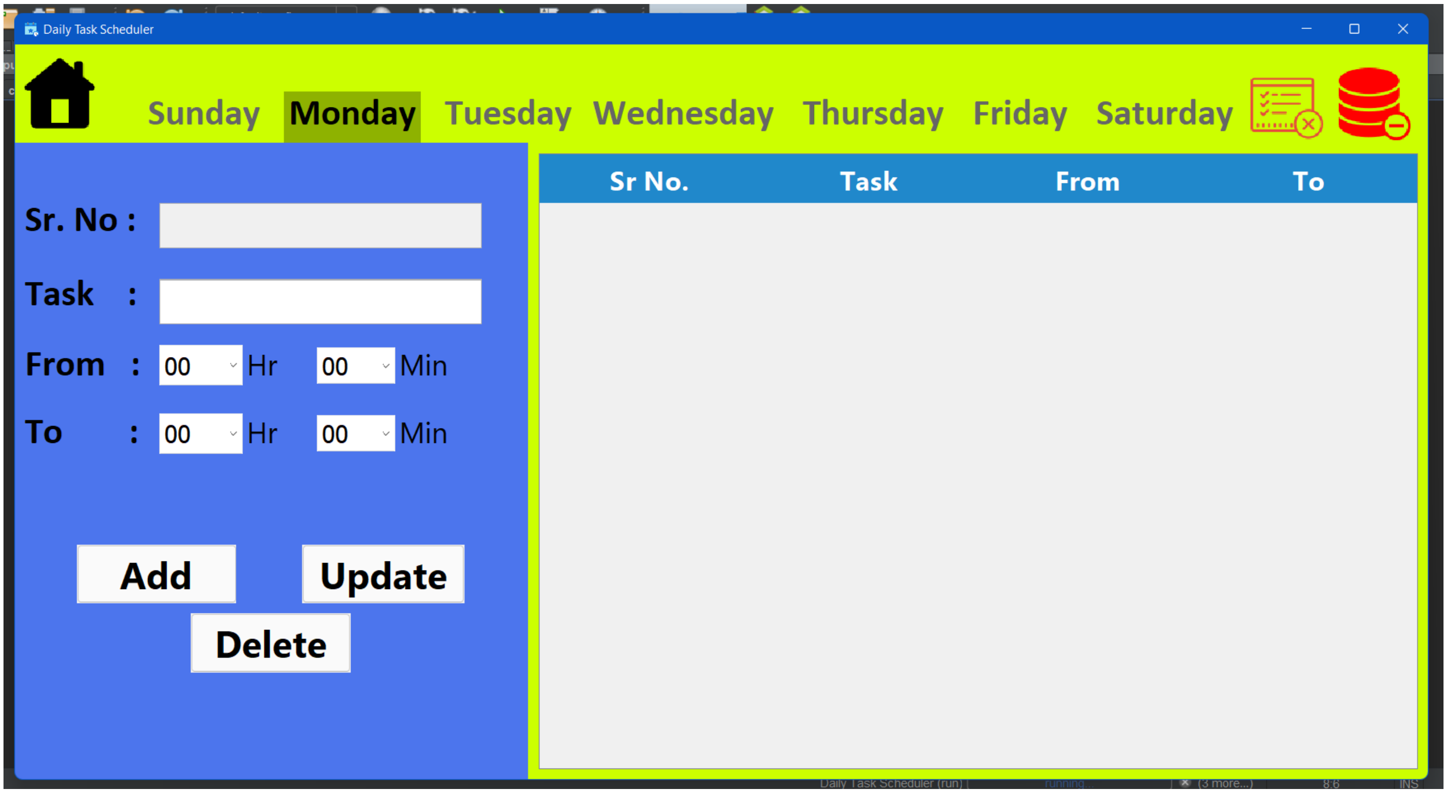
Sr No. To

Pomodoro Time: 0:25:00

Short Break Time: 0:05:00

Long Break Time: 0:30:00

Timer: 0:00:00



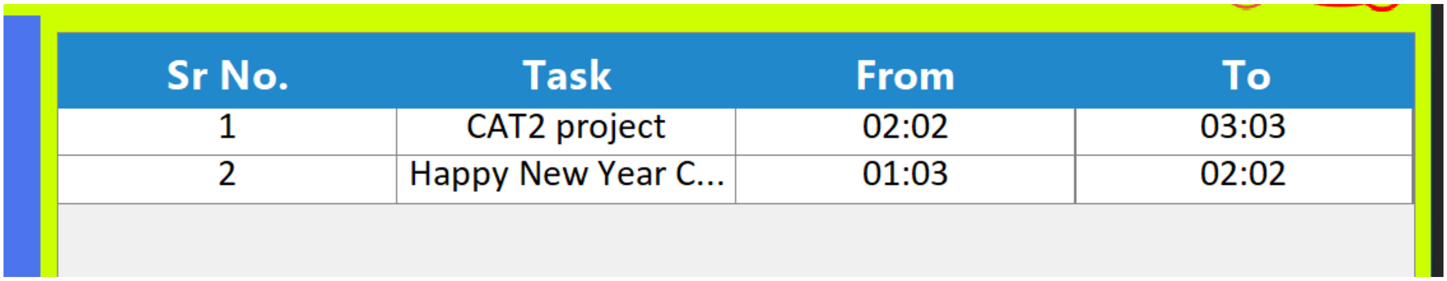
Sr. No :

Task :

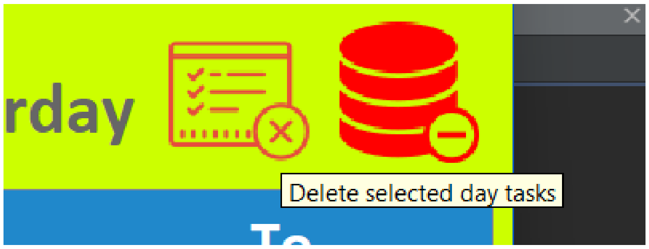
From : 00 Hr 00 Min

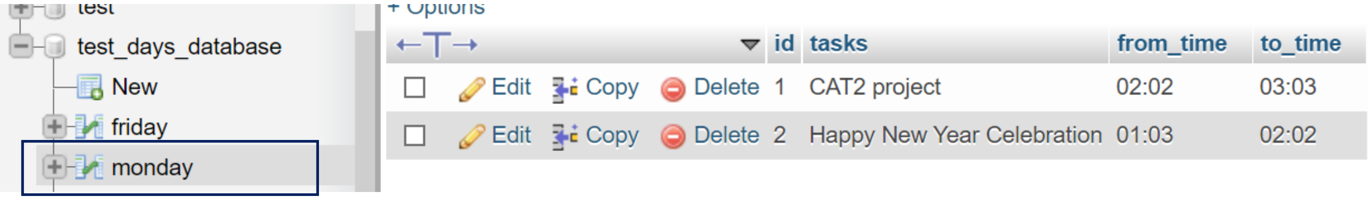
To : 00 Hr 00 Min

Sr No.	Task	From	To

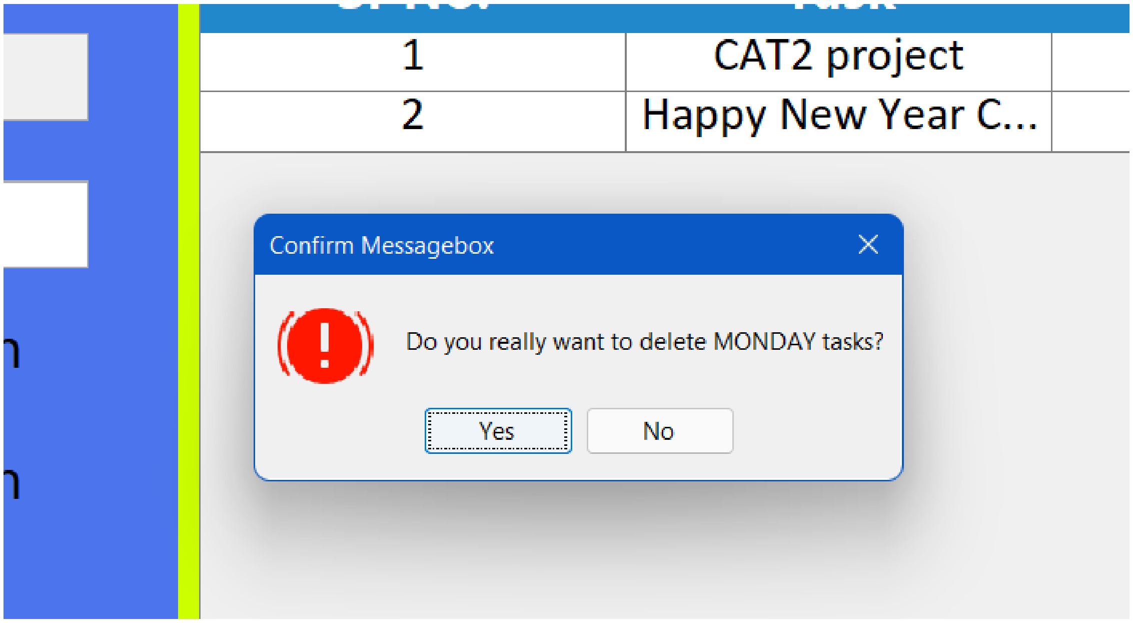


Sr No.	Task	From	To
1	CAT2 project	02:02	03:03
2	Happy New Year C...	01:03	02:02





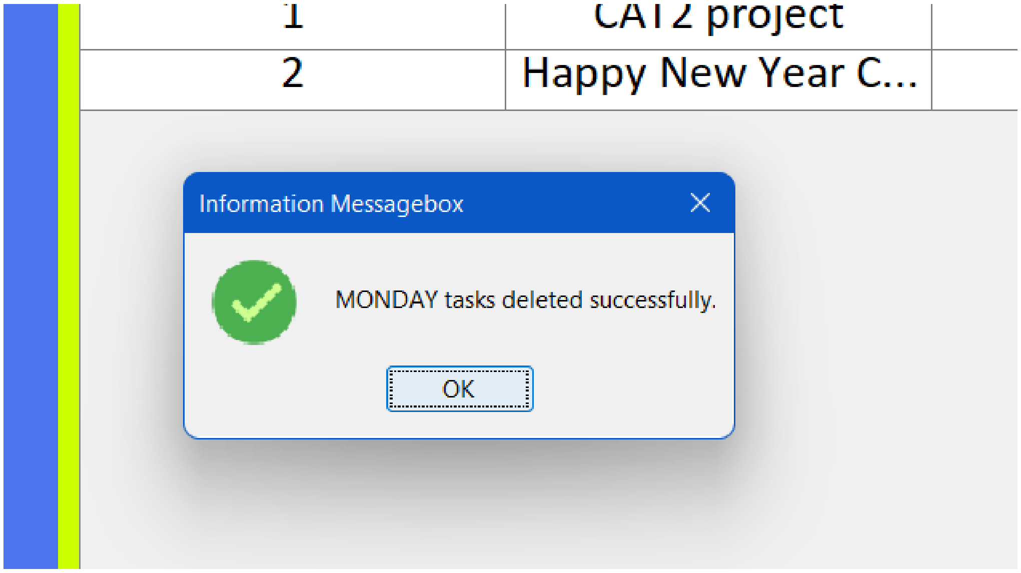
				id	tasks	from_time	to_time
□	Edit	Copy	Delete	1	CAT2 project	02:02	03:03
□	Edit	Copy	Delete	2	Happy New Year Celebration	01:03	02:02



1	CAT2 project
2	Happy New Year C...

Confirm Messagebox

Do you really want to delete MONDAY tasks?



1	CAT2 project
2	Happy New Year C...

Information Messagebox

MONDAY tasks deleted successfully.

CONCLUSION AND FUTURE ENHANCEMENTS:

THE ABOVE PROGRAM CAN BE ENHANCED BY ADDING LOGIN AUTHETICATION SYSTEM FOR INDIVIDUAL USERS,MORE UI/UX CAN BE ENHANCED,SMALL RECTIFICATIONS IN THE CONVENTION OF THE FEATURES CAN BE MADE TO IMPROVE THE PERFORMANCE OF THE APPLICATION BETTER.