2Do-List

Overview

2Do-List Application is a Java Base Application. It is Simple and faster to save a Task.

It is truly usable with great user experience.

With Todoist, you can keep track of everything – from simple errands like grocery shopping, to your most ambitious projects – so you can start getting things done and enjoy more peace-of-mind along the way.

Goals

- **Capture and organize tasks into your to-do list the moment they pop into your head**
- **Organize and plan your day**
- **Increase your productivity and decrease your stress levels.**

Features

- **Add Task**
- **Add notes to your tasks**
- **Nice UI**
- **Each User Have Separate task.**

System requirements

2Do-List is Java Application . For Making this App you need some requirement

Software:

- IntelliJ IDEA
- Java 1.8+
- MySql

Hardware requirement

- Min 2gb Ram
- 100Mb disk space

Language Known:

- Java
- FXML
- CSS
- SQL

Database:

- MySql

System Design Details

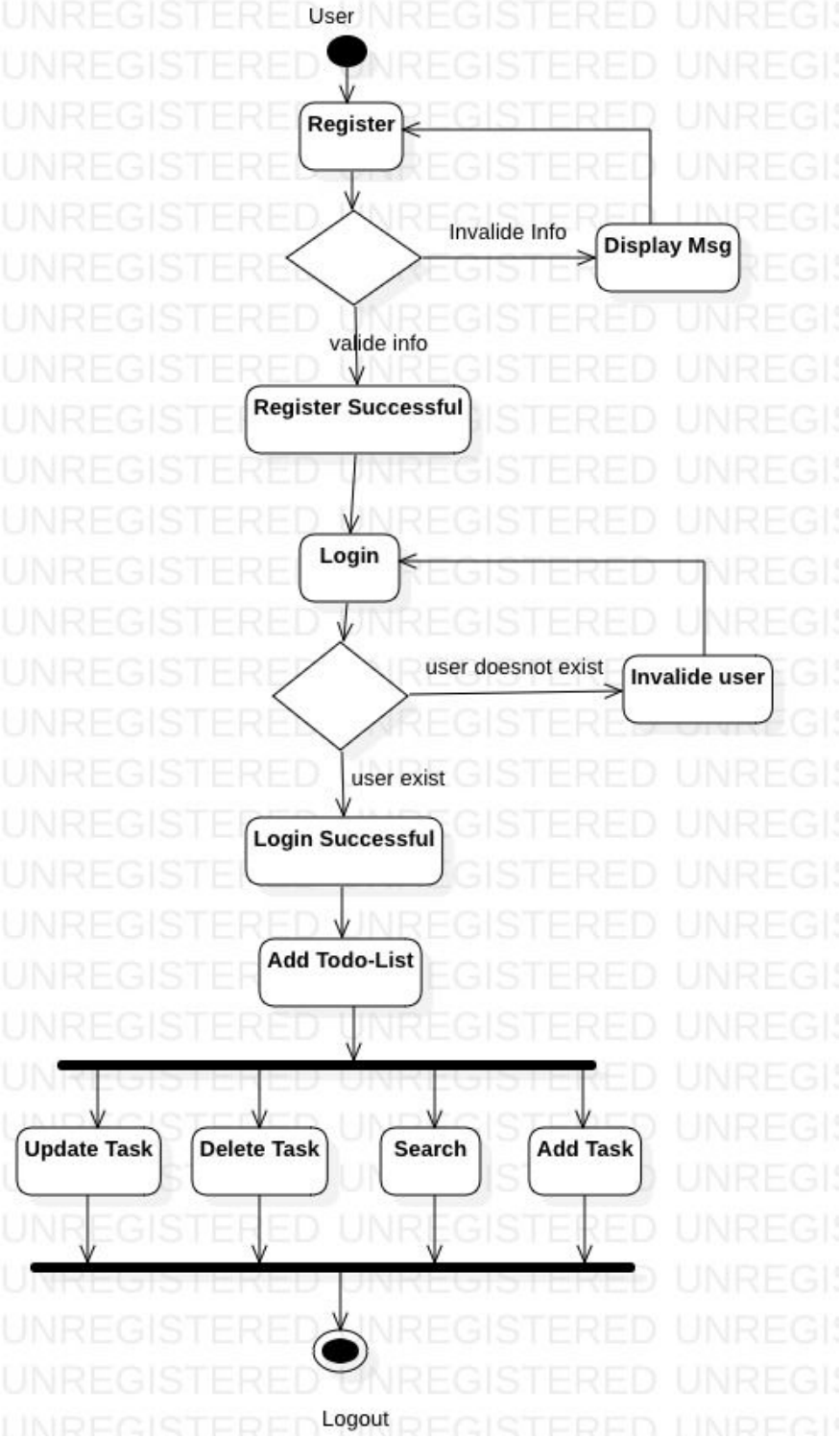
1)Classes

- Shake
- AddItemController
- AddItemFromController
- CellController
- ListController
- LoginController
- SignUpController
- UpdateTaskController
- Config
- Const
- DatabaseHandler
- Task
- User
- Main

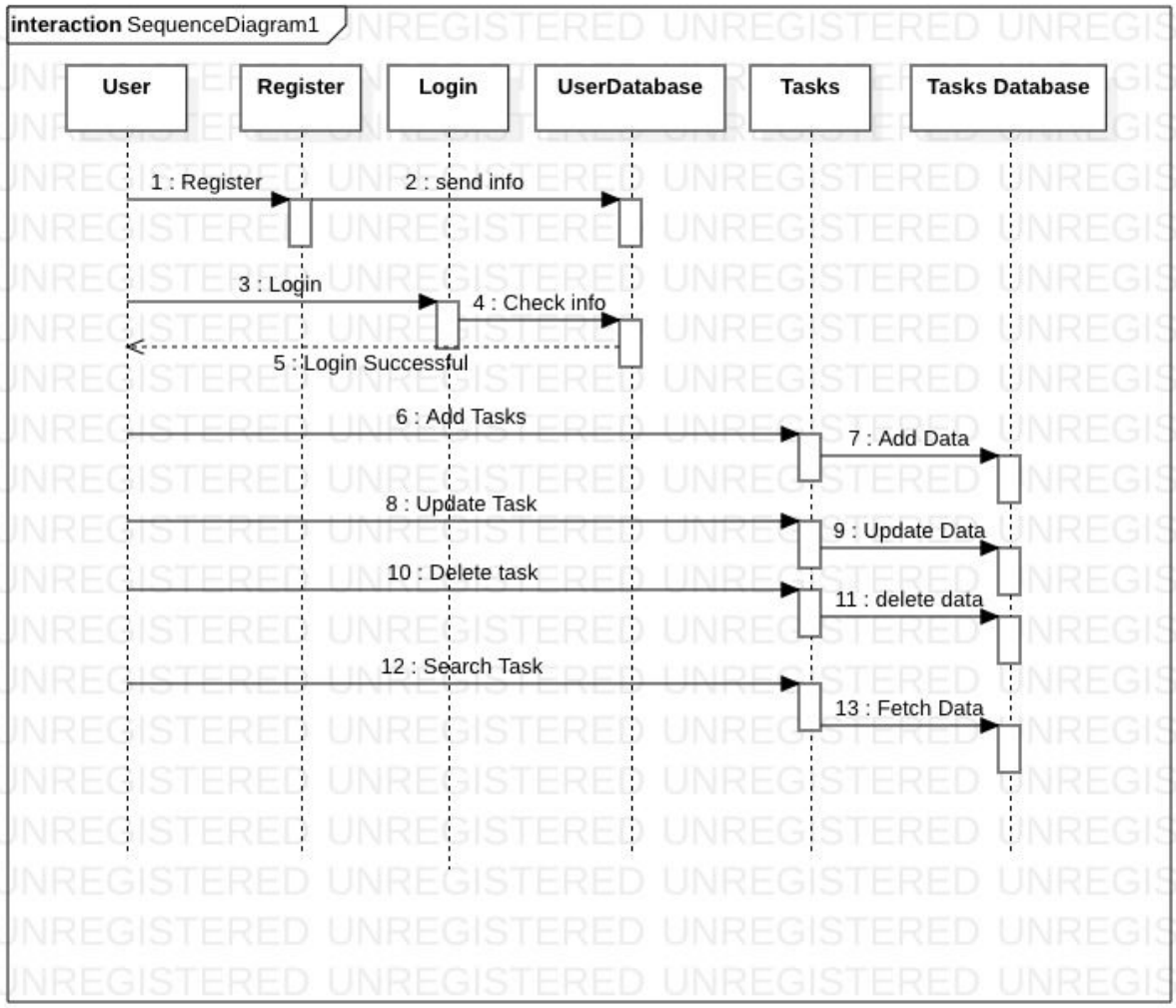
Diagrams

- ACTIVITY DIAGRAM
- SEQUENCE
- USE CASE
- CLASS
- Database Diagram

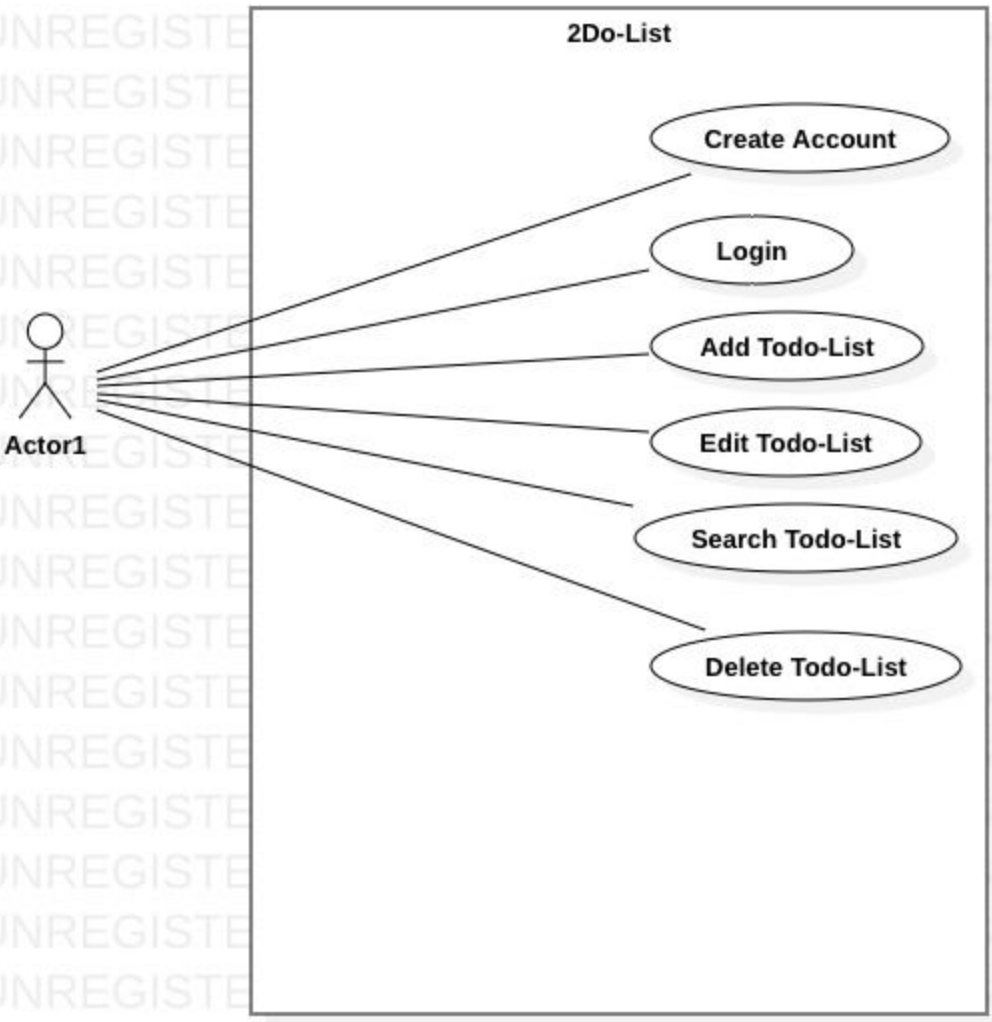
Activity Diagram



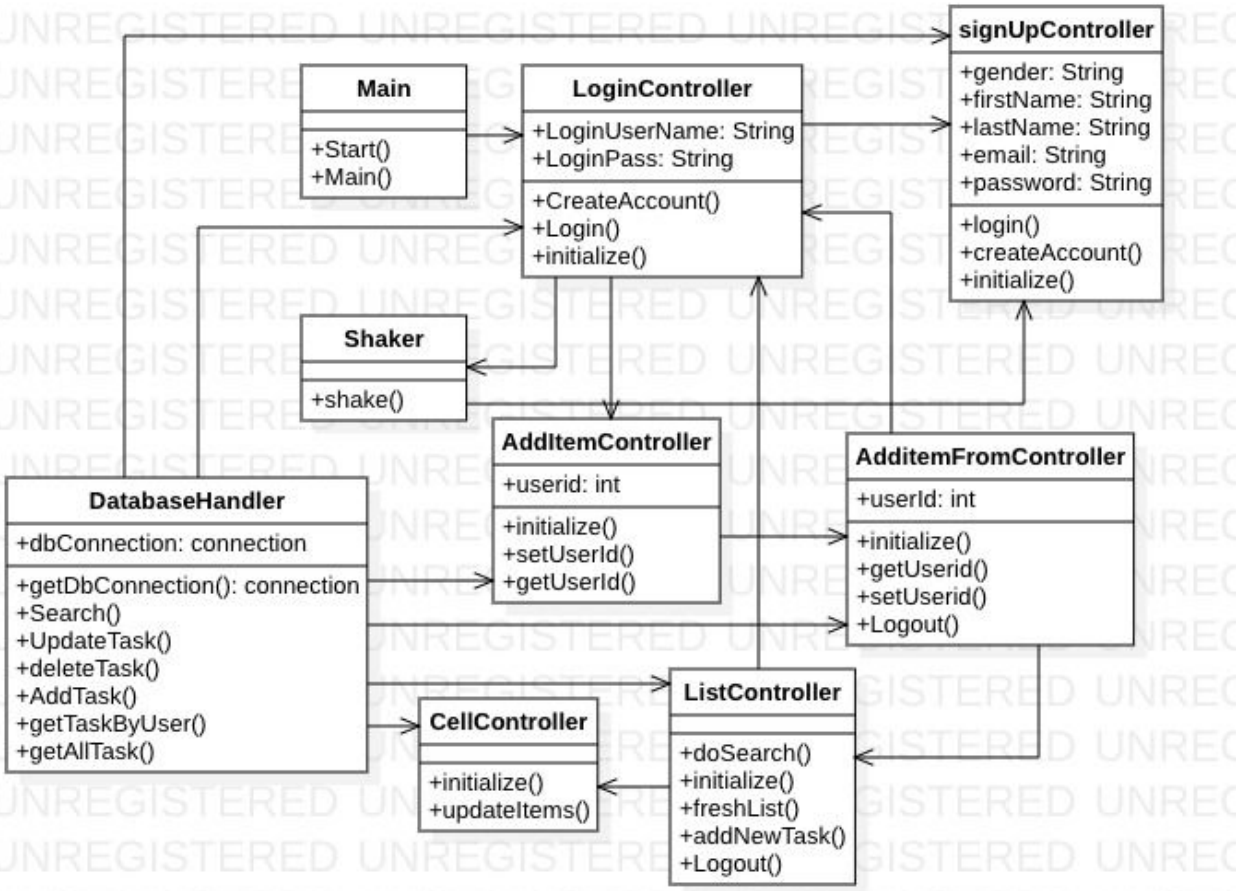
SEQUENCE DIAGRAM



USE CASE DIAGRAM



Class Diagram



Database Tables

Users

userid	firstname	lastname	username	password	location	gender
1	James	Bond	jamesb	password	England	male
2	Ana	Dennis	adenis	adonis8	Mozambique	female
...						
...						

Tasks

taskid	userid	datecreated	description
1	1	Bond	jamesb
2	2	Dennis	adenis
...			
...			

Code.

Main.Class

```
package sample;
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;
public class Main extends Application {
@Override
    public void start(Stage primaryStage) throws Exception{
        Parent root = FXMLLoader.load(getClass().getResource("view/login.fxml"));
        primaryStage.setTitle("Hello World");
        Scene scene=new Scene(root, 700, 400);
primaryStage.setScene(scene);
        scene.getStylesheets().addAll("sample/view/Style.css");
        primaryStage.setResizable(false);
        primaryStage.show();
    }
public static void main(String[] args) {
        launch(args);
    }
}
```

Shake.Class

```
package sample.animations;
import javafx.animation.TranslateTransition;
import javafx.scene.Node;
import javafx.util.Duration;
public class Shaker {
   private TranslateTransition translateTransition;
   public Shaker(Node node) {
      translateTransition =
            new TranslateTransition(Duration.millis(50), node);
      translateTransition.setFromX(0f);
      translateTransition.setByX(10f);
      translateTransition.setCycleCount(2);
      translateTransition.setAutoReverse(true);
}
   public void shake() {
      translateTransition.playFromStart();
   }
}
```

AddItemController.Class

```
package sample.controller;
import javafx.animation.FadeTransition;
import javafx.event.Event;
import javafx.event.EventDispatchChain;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.AnchorPane;
import javafx.util.Duration;
import sample.animations.Shaker;
import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
public class AddItemController {
 public static int userId;
@FXML
    private AnchorPane paneRoot;
    @FXML
    private ResourceBundle resources;
    @FXML
    private URL location;
    @FXML
    private ImageView addButton;

@FXML
    private Label notTaskLabel;
    @FXML
    void initialize() {
      addButton.addEventHandler(MouseEvent.MOUSE_CLICKED, event -> {
          Shaker buttonShaker = new Shaker(addButton);
```

```
        buttonShaker.shake();
        FadeTransition fadeTransition = new FadeTransition(Duration.millis(2000), addButton);
        FadeTransition labelTransition = new FadeTransition(Duration.millis(2000), notTaskLabel);
        //remove
        System.out.println("Added Clicked!");
        addButton.relocate(0, 20);
        notTaskLabel.relocate(0, 85);
        addButton.setOpacity(0);
        notTaskLabel.setOpacity(0);
        fadeTransition.setFromValue(1f);
        fadeTransition.setToValue(0f);
        fadeTransition.setCycleCount(1);
        fadeTransition.setAutoReverse(false);
        fadeTransition.play();
        labelTransition.setFromValue(1f);
        labelTransition.setToValue(0f);
        labelTransition.setCycleCount(1);
        labelTransition.setAutoReverse(false);
        labelTransition.play();
try {
            AnchorPane formPane =
                    FXMLLoader.load(getClass().getResource("/sample/view/addItemForm.fxml"));
            AddItemController.userId = getUserId();
            FadeTransition rootTransition = new FadeTransition(Duration.millis(1000), formPane);
            rootTransition.setFromValue(0f);
            rootTransition.setToValue(1f);
            rootTransition.setCycleCount(1);
            rootTransition.setAutoReverse(false);
            rootTransition.play();
```

```
            paneRoot.getChildren().setAll(formPane);
        } catch (IOException e) {
            e.printStackTrace();
        }
    });
  }
  public void setUserId(int userId) {
    this.userId = userId;
    System.out.println("User Id is " + this.userId);
  }
  public int getUserId(){
    return this.userId;
  }
}
```

AddItemFormController.Class

```
package sample.controller;
import com.jfoenix.controls.JFXButton;
import com.jfoenix.controls.JFXTextField;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;
import javafx.stage.Window;
import sample.Database.DatabaseHandler;
import sample.model.Task;
import java.io.IOException;
import java.sql.SQLException;
import java.util.Calendar;
public class AddItemFormController {
    private int userId;
private DatabaseHandler databaseHandler;
    @FXML
    private JFXTextField taskField;
    @FXML
    private JFXTextField descriptionField;
    @FXML
    private JFXButton saveTaskButton;
    @FXML
    private AnchorPane rootPane;
    @FXML
```

```
private Label successLabel;
@FXML
private JFXButton todosButton;
@FXML
private JFXButton taskLogoutButton;
@FXML
void initialize() {
   taskLogoutButton.setOnAction(event -> {
      taskLogoutButton.getScene().getWindow().hide();
      FXMLLoader loader = new FXMLLoader();
      loader.setLocation(getClass().getResource("/sample/view/login.fxml"));
      try {
         loader.setRoot(loader.getRoot());
         loader.load();
      } catch (IOException e) {
         e.printStackTrace();
      }
      Parent root = loader.getRoot();
      Stage stage = new Stage();
      stage.setScene(new Scene(root));
      stage.showAndWait();
   });
   databaseHandler = new DatabaseHandler();
   Task task = new Task();
   saveTaskButton.setOnAction(event -> {
      Calendar calendar = Calendar.getInstance();
      java.sql.Timestamp timestamp =
            new java.sql.Timestamp(calendar.getTimeInMillis());
      String taskText = taskField.getText().trim();
```

```
String taskDescription = descriptionField.getText().trim();
        if (!taskText.equals("") || !taskDescription.equals("")) {
            System.out.println("User Id: " + AddItemController.userId);
            task.setUserId(AddItemController.userId);
            task.setDatecreated(timestamp);
            task.setDescription(taskDescription);
            task.setTask(taskText);
            databaseHandler.insertTask(task);
            successLabel.setVisible(true);
            todosButton.setVisible(true);
            int taskNumber = 0;
            try {
                taskNumber = databaseHandler.getAllTasks(AddItemController.userId);
            } catch (SQLException e) {
                e.printStackTrace();
            } catch (ClassNotFoundException e) {
                e.printStackTrace();
            }
            todosButton.setText("My 2Do's: " + "(" + taskNumber + ")");
            taskField.setText("");
            descriptionField.setText("");
            todosButton.setOnAction(event1 -> {
                //send users to the list screen
                try {
                    AnchorPane pane = FXMLLoader.load(getClass().getResource("/sample/view/list.fxml"));
                    // pane.setMaxSize(700,400);
                    rootPane.getChildren().setAll(pane);
                    rootPane.setMaxSize(700, 400);
                } catch (IOException e) {
```

```
                    e.printStackTrace();
                }
            });
            System.out.println("Task Added Successfully!");
        } else {
            System.out.println("Nothing added!");
        }
    });
  }
  public int getUserId() {
    System.out.println("from getUserId() " + userId);
    return userId;
  }
  public void setUserId(int userId) {
    this.userId = userId;
    System.out.println("From setUserId " + this.userId);
  }
}
```

CellController.Class

```
package sample.controller;
import com.jfoenix.controls.JFXButton;
import com.jfoenix.controls.JFXDialog;
import com.jfoenix.controls.JFXListCell;
import com.jfoenix.controls.JFXTextField;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.geometry.Insets;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;
import javafx.util.Pair;
import sample.Database.DatabaseHandler;
import sample.model.Task;
import java.io.IOException;
import java.sql.SQLException;
import java.util.Calendar;
public class CellController extends JFXListCell<Task> {
    @FXML
    private AnchorPane rootAnchorPane;
    @FXML
    private ImageView iconImageView;
```

```
@FXML
private Label taskLabel;
@FXML
private ImageView deleteButton;
@FXML
public ImageView listUpdateButton;
private FXMLLoader fxmlLoader;
private DatabaseHandler databaseHandler;
@FXML
void initialize() throws SQLException {
}
@Override
public void updateItem(Task myTask, boolean empty) {
   databaseHandler = new DatabaseHandler(); //main change
   super.updateItem(myTask, empty);
   if (empty || myTask == null) {
      setText(null);
      setGraphic(null);
   }else {
       if (fxmlLoader == null ) {
           fxmlLoader = new FXMLLoader(getClass()
                   .getResource("/sample/view/cell.fxml"));
           fxmlLoader.setController(this);

          try {
              fxmlLoader.load();
          } catch (IOException e) {
              e.printStackTrace();
```

```
    }
}
taskLabel.setText(myTask.getTask());
dateLabel.setText(myTask.getDatecreated().toString());
descriptionLabel.setText(myTask.getDescription());
int taskId = myTask.getTaskId();
listUpdateButton.setOnMouseClicked(event -> {
  FXMLLoader loader = new FXMLLoader();
  loader.setLocation(getClass().getResource("/sample/view/updateTaskForm.fxml"));
   try {
      loader.load();
   } catch (IOException e) {
      e.printStackTrace();
   }
   Parent root = loader.getRoot();
   Stage stage = new Stage();
   stage.setScene(new Scene(root));
   UpdateTaskController updateTaskController = loader.getController();
   updateTaskController.setTaskField(myTask.getTask());
   updateTaskController.setUpdateDescriptionField(myTask.getDescription());
   updateTaskController.updateTaskButton.setOnAction(event1 -> {
      Calendar calendar = Calendar.getInstance();
      java.sql.Timestamp timestamp =
            new java.sql.Timestamp(calendar.getTimeInMillis());

      try {
         System.out.println("taskid " + myTask.getTaskId());
         databaseHandler.updateTask(timestamp, updateTaskController.getDescription(),
               updateTaskController.getTask(), myTask.getTaskId());
```

```
 } catch (SQLException e) {
                    e.printStackTrace();
                } catch (ClassNotFoundException e) {
                    e.printStackTrace();
                }
            });
            stage.show();
        });
        deleteButton.setOnMouseClicked(event -> {
            try {
                databaseHandler.deleteTask(AddItemController.userId,taskId);
            } catch (SQLException e) {
                e.printStackTrace();
            } catch (ClassNotFoundException e) {
                e.printStackTrace();
            }
            getListView().getItems().remove(getItem());
        });
        setText(null);
        setGraphic(rootAnchorPane);
    }   }
}
```

ListController.Class

```
package sample.controller;
import com.jfoenix.controls.JFXButton;
import javafx.scene.Scene;
import javafx.scene.image.ImageView;
import javafx.stage.Stage;
import sample.Database.DatabaseHandler;
import sample.model.Task;
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Timestamp;
import java.util.Calendar;
public class ListController {
    @FXML
   public JFXButton listSaveTaskButton;
   private ObservableList<Task> tasks;
   private ObservableList<Task> refreshedTasks;
   private DatabaseHandler databaseHandler;
   @FXML
   private JFXButton listLogout;
   @FXML
   private JFXTextField search;
   @FXML
   private JFXButton searchBtn;
   @FXML
   void dosearch(ActionEvent event) throws SQLException {
      databaseHandler = new DatabaseHandler();
      String tast = search.getText().toString();
```

```
    ResultSet resultSet = databaseHandler.search(tast);
    tasks = FXCollections.observableArrayList();
    while (resultSet.next()) {
        Task task = new Task();
        task.setTaskId(resultSet.getInt("taskid"));
        task.setTask(resultSet.getString("task"));
        task.setDatecreated(resultSet.getTimestamp("datecreated"));
        task.setDescription(resultSet.getString("description"));
        tasks.addAll(task);
    }
    listTask.setItems(tasks);
    listTask.setCellFactory(CellController -> new CellController());
}
@FXML
void initialize() throws SQLException {
    listLogout.setOnAction(event -> {
        listLogout.getScene().getWindow().hide();
        FXMLLoader loader = new FXMLLoader();
        loader.setLocation(getClass().getResource("/sample/view/login.fxml"));
        try {
            loader.load();
        } catch (IOException e) {
            e.printStackTrace();
        }
        Parent root = loader.getRoot();
        Stage stage = new Stage();
        stage.setScene(new Scene(root));
        stage.show();
    });
```

```
    System.out.println("initialize called");
    tasks = FXCollections.observableArrayList();
    databaseHandler = new DatabaseHandler();
    ResultSet resultSet = databaseHandler.getTasksByUser(AddItemController.userId);
    while (resultSet.next()) {
        Task task = new Task();
        task.setTaskId(resultSet.getInt("taskid"));
        task.setTask(resultSet.getString("task"));
        task.setDatecreated(resultSet.getTimestamp("datecreated"));
        task.setDescription(resultSet.getString("description"));
        tasks.addAll(task); }
    listTask.setItems(tasks);
    listTask.setCellFactory(CellController -> new CellController());
    listRefreshButton.setOnMouseClicked(event -> {
        try {
            refreshList();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    });
    listSaveTaskButton.setOnAction(event -> {
        addNewTask();
    });
}
public void refreshList() throws SQLException {
    System.out.println("refreshList in ListCont called");
    refreshedTasks = FXCollections.observableArrayList();
    DatabaseHandler databaseHandler = new DatabaseHandler();
    ResultSet resultSet = databaseHandler.getTasksByUser(AddItemController.userId);
```

```
    while (resultSet.next()) {
        Task task = new Task();
        task.setTaskId(resultSet.getInt("taskid"));
        task.setDatecreated(resultSet.getTimestamp("datecreated"));
        task.setDescription(resultSet.getString("description"));
        refreshedTasks.addAll(task);
    }
    listTask.setItems(refreshedTasks);
    listTask.setCellFactory(CellController -> new CellController());
  }
  public void addNewTask() {
    if (!listTaskField.getText().equals("")
            || !listDescriptionField.getText().equals("")) {
        Task myNewTask = new Task();
        Calendar calendar = Calendar.getInstance();
        Timestamp timestamp =
                new Timestamp(calendar.getTimeInMillis());
         myNewTask.setTask(listTaskField.getText().trim());
         myNewTask.setDatecreated(timestamp);
        databaseHandler.insertTask(myNewTask);
        listTaskField.setText("");
        listDescriptionField.setText("");
        try {
            initialize();
        } catch (SQLException e) {
            e.printStackTrace();
        }    }
    }
}
```

LoginController.Class

```
package sample.controller;
import com.jfoenix.controls.JFXButton;
import com.jfoenix.controls.JFXPasswordField;
import com.jfoenix.controls.JFXTextField;
import javafx.animation.TranslateTransition;
import sample.Database.DatabaseHandler;
import sample.animations.Shaker;
import sample.model.User;
import java.io.IOException;
import java.net.URL;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ResourceBundle;
public class LoginController {
    private int userId;
    private int a = 0;
    @FXML
    private ResourceBundle resources;
    @FXML
    private URL location;
    @FXML
    private JFXTextField loginUsername;
    @FXML
    private JFXPasswordField loginPassword;
    @FXML
    private Text loginSignupButton;
    @FXML
    void Create_Account(MouseEvent event) {
```

```
        loginSignupButton.getScene().getWindow().hide();
        FXMLLoader loader = new FXMLLoader();
        loader.setLocation(getClass().getResource("/sample/view/signup.fxml"));
        try {
            loader.load();
        } catch (IOException e) {
            e.printStackTrace();
        }
        Parent root = loader.getRoot();
        Stage stage = new Stage();
        stage.setScene(new Scene(root));
        stage.showAndWait();
    }
    @FXML
    private JFXButton loginButton;
    private DatabaseHandler databaseHandler;
    @FXML
    void passClick(MouseEvent event) {
if (a == 2) {
            wrongInp();
            a = 1;
        }
    }
    @FXML
    void userclick(MouseEvent event) {
        if (a == 2) {
            wrongInp();
            a = 1;
        }
```

```
    }
    @FXML
    void initialize() {
        databaseHandler = new DatabaseHandler();
        loginButton.setOnAction(event -> {
            String loginText = loginUsername.getText().trim();
            String loginPwd = loginPassword.getText().trim();
            if (!loginText.equals("") && !loginPwd.equals("")){
                User user = new User();
                user.setUserName(loginText);
                user.setPassword(loginPwd);
                ResultSet userRow = databaseHandler.getUser(user);
                int counter = 0;
                try {
                    while (userRow.next()) {
                        counter++;
                        String name = userRow.getString("firstname");
                        userId = userRow.getInt("userid");
System.out.println("Welcome! " + name);
                        showAddItemScreen();
                    }
                    if (counter == 1) {
                        a = 1;
                    } else {
                        Shaker userNameShaker = new Shaker(loginUsername);
                        Shaker passwordShaker = new Shaker(loginPassword);
                        passwordShaker.shake();
                        userNameShaker.shake();
                        loginUsername.setStyle("-fx-text-fill: red;");
```

```
                loginPassword.setStyle("-fx-text-fill: red;");
                a = 2;
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    else {
        Shaker userNameShaker = new Shaker(loginUsername);
        Shaker passwordShaker = new Shaker(loginPassword);
        passwordShaker.shake();
        userNameShaker.shake();
        loginUsername.setStyle("-fx-border-color:  #ff1c15;");
        loginPassword.setStyle("-jfx-focus-color: #ff2a25;");
        wrongInp();
    }
  });
}
private void showAddItemScreen() {
  //Take users to AddItem screen
  loginSignupButton.getScene().getWindow().hide();
  FXMLLoader loader = new FXMLLoader();
  loader.setLocation(getClass().getResource("/sample/view/addItem.fxml"));
  try {
    loader.setRoot(loader.getRoot());
    loader.load();
  } catch (IOException e) {
    e.printStackTrace();
  }
```

```
        Parent root = loader.getRoot();
        Stage stage = new Stage();
        stage.setScene(new Scene(root));
        stage.setResizable(false);
        stage.sizeToScene();
        AddItemController addItemController = loader.getController();
        addItemController.setUserId(userId);
        stage.showAndWait();
    }
    private void wrongInp() {
        loginUsername.setStyle("-fx-text-fill: #2c2c2c;");
        loginPassword.setStyle("-fx-text-fill: #2c2c2c;");
        loginPassword.clear();
        loginUsername.clear();
    }
}
```

SignupController.Class

```
package sample.controller;
import com.jfoenix.controls.JFXButton;
import com.jfoenix.controls.JFXCheckBox;
import com.jfoenix.controls.JFXPasswordField;
import com.jfoenix.controls.JFXTextField;
import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.input.MouseEvent;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;
import sample.Database.DatabaseHandler;
import sample.animations.Shaker;
import sample.model.User;
public class SignupController {
    String gender=null;
    @FXML
    private ResourceBundle resources;
    @FXML
    private URL location;
    @FXML
    private JFXTextField singUpLastName;
    @FXML
```

```
private JFXTextField signUpFirstName;
@FXML
private JFXTextField signUpUsername;
@FXML
private AnchorPane bgReg;
@FXML
private JFXPasswordField signUpPassword;
@FXML
private JFXTextField signUpLocation;
@FXML
private JFXCheckBox singUpCheckBoxMale;
@FXML
private JFXCheckBox singUpCheckBoxFemale;
@FXML
private JFXButton signUpButton;
@FXML
private JFXButton loginSignupButton;
@FXML
void selectedFemale(MouseEvent event) {
   gender="Female";
}
@FXML
void selectedMale(MouseEvent event) {
   gender="Male";
}
@FXML
void alreadyhaveAccount(ActionEvent event) {
   loginSignupButton.getScene().getWindow().hide();
   FXMLLoader loader = new FXMLLoader();
```

```
        loader.setLocation(getClass().getResource("/sample/view/login.fxml"));
        try {
            loader.load();
        } catch (IOException e) {
            e.printStackTrace();
        }
        Parent root = loader.getRoot();
        Stage stage = new Stage();
        stage.setScene(new Scene(root));
        stage.show();
    }
    @FXML
    void initialize() {
        signUpButton.setOnAction(event -> {
            String name = signUpFirstName.getText();
            String lastName = singUpLastName.getText();
            String userName = signUpUsername.getText();
            String password = signUpPassword.getText();
            String location = signUpLocation.getText();
            String chkgender = gender;
            if (!name.trim().equals("") && !lastName.trim().equals("") &&
                    !userName.trim().equals("")&& !password.trim().equals("") && !location.trim().equals("") &&
!chkgender.equals("")) {
                createUser();
                signUpButton.getScene().getWindow().hide();
                FXMLLoader loader = new FXMLLoader();
                loader.setLocation(getClass().getResource("/sample/view/login.fxml"));
                try {
                    loader.load();
                } catch (IOException e) {
```

```
                e.printStackTrace();
            }
            Parent root = loader.getRoot();
            Stage stage = new Stage();
            stage.setScene(new Scene(root));
            stage.show();
        }
        else {
            Shaker firstNameShaker = new Shaker(signUpFirstName);
            Shaker lastNameShaker = new Shaker(singUpLastName);
            Shaker userNameShaker = new Shaker(signUpUsername);
            Shaker passwordShaker = new Shaker(signUpPassword);
            Shaker locationShaker = new Shaker(signUpLocation);
            locationShaker.shake();
            firstNameShaker.shake();
            lastNameShaker.shake();
            passwordShaker.shake();
            userNameShaker.shake();
            signUpPassword.clear();
            System.out.println("Fill form");
        }
    });
}
private void createUser() {
    DatabaseHandler databaseHandler = new DatabaseHandler();
    String name = signUpFirstName.getText();
    String lastName = singUpLastName.getText();
    String userName = signUpUsername.getText();
    String password = signUpPassword.getText();
```

```
        String location = signUpLocation.getText();
        String chkgender=gender;

        if (!name.trim().equals("") && !lastName.trim().equals("") &&
                !userName.trim().equals("") && !location.trim().equals("")) {
            System.out.println("not null");
            User user = new User(name, lastName, userName, password, location, chkgender);
            databaseHandler.signUpUser(user);
        }  else {
            signUpFirstName.clear();
            singUpLastName.clear();
            signUpUsername.clear();
            signUpPassword.clear();
            singUpCheckBoxFemale.setSelected(false);
            singUpCheckBoxMale.setSelected(false);
            signUpLocation.clear();
        }

    }
}
```

UpdateTaskController.Class

```
package sample.controller;

import com.jfoenix.controls.JFXButton;

import com.jfoenix.controls.JFXTextField;

import javafx.fxml.FXML;

public class UpdateTaskController {

    @FXML

    private JFXTextField updateTaskField;

    @FXML

    private JFXTextField updateDescriptionField;

    @FXML

    public JFXButton updateTaskButton;

    @FXML

    void initialize() {

    }

    public void setTaskField(String task) {

        this.updateTaskField.setText(task);

    }

    public String getTask() {

        return this.updateTaskField.getText().trim();

    }

    public void setUpdateDescriptionField(String description) {

        this.updateDescriptionField.setText(description);
```

```
    }

    public String getDescription() {

        return this.updateDescriptionField.getText().trim();

    }

}
```

DatabaseHandler.Class

```
package sample.Database;

import sample.model.Task;

import sample.model.User;

import java.sql.*;

public class DatabaseHandler extends Configs {

    Connection dbConnection;

    public Connection getDbConnection() throws ClassNotFoundException, SQLException {

        String connectionString = "jdbc:mysql://" + dbHost + ":"

                + dbPort + "/"

                + dbName;

        Class.forName("com.mysql.cj.jdbc.Driver");

        dbConnection = DriverManager.getConnection(connectionString, dbUser, dbPass);

        return dbConnection;

    }

    public ResultSet search(String task) {

        ResultSet resultTasks = null;

        String query = "SELECT * FROM " + Const.TASKS_TABLE + " WHERE "

                + Const.TASKS_TASK + "=? order by " + Const.TASKS_ID + " desc";

        try {

            PreparedStatement preparedStatement = getDbConnection().prepareStatement(query);

            preparedStatement.setString(1, task);

            resultTasks = preparedStatement.executeQuery();
```

```
 } catch (Exception e) {
            e.printStackTrace();
        }
        return resultTasks;
    }
    public void updateTask(Timestamp datecreated, String description, String task, int taskId) throws
SQLException, ClassNotFoundException {
        String query = "UPDATE tasks SET datecreated=?, description=?, task=? WHERE taskid=?";
        PreparedStatement preparedStatement = getDbConnection().prepareStatement(query);
        preparedStatement.setTimestamp(1, datecreated);
        preparedStatement.setString(2, description);
        preparedStatement.setString(3, task);
        // preparedStatement.setInt(4, userId);
        preparedStatement.setInt(4, taskId);
        preparedStatement.executeUpdate();
        preparedStatement.close();
    }
    //Delete Task
    public void deleteTask(int userId, int taskId) throws SQLException, ClassNotFoundException {
        String query = "DELETE FROM " + Const.TASKS_TABLE + " WHERE " +
                Const.USERS_ID + "=?" + " AND " + Const.TASKS_ID + "=?";
        PreparedStatement preparedStatement = getDbConnection().prepareStatement(query);
        preparedStatement.setInt(1, userId);
        preparedStatement.setInt(2, taskId);
```

```
preparedStatement.execute();
        preparedStatement.close();
    }
    //Write
    public void signUpUser(User user) {
     String insert = "INSERT INTO " + Const.USERS_TABLE + "(" + Const.USERS_FIRSTNAME
                + "," + Const.USERS_LASTNAME + "," + Const.USERS_USERNAME + ","
                + Const.USERS_PASSWORD + "," + Const.USERS_LOCATION + ","
                + Const.USERS_GENDER + ")" + "VALUES(?,?,?,?,?,?)";
        try {
            PreparedStatement preparedStatement = getDbConnection().prepareStatement(insert);
            preparedStatement.setString(1, user.getFirstName());
            preparedStatement.setString(2, user.getLastName());
            preparedStatement.setString(3, user.getUserName());
            preparedStatement.setString(4, user.getPassword());
            preparedStatement.setString(5, user.getLocation());
            preparedStatement.setString(6, user.getGender());
            preparedStatement.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    public ResultSet getTasksByUser(int userId) {
        ResultSet resultTasks = null;
```

```
    String query = "SELECT * FROM " + Const.TASKS_TABLE + " WHERE "
            + Const.USERS_ID + "=? order by " + Const.TASKS_ID + " desc";
    try {
        PreparedStatement preparedStatement = getDbConnection().prepareStatement(query);
        preparedStatement.setInt(1, userId);
        resultTasks = preparedStatement.executeQuery();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return resultTasks;
}
public ResultSet getUser(User user) {
    ResultSet resultSet = null;
    if (!user.getUserName().equals("") || !user.getPassword().equals("")) {
        String query = "SELECT * FROM " + Const.USERS_TABLE + " WHERE "
                + Const.USERS_USERNAME + "=?" + " AND " + Const.USERS_PASSWORD
                + "=?";
        // select all from users where username="paulo" and password="password"
        try {
            PreparedStatement preparedStatement = getDbConnection().prepareStatement(query);
            preparedStatement.setString(1, user.getUserName());
            preparedStatement.setString(2, user.getPassword());
            resultSet = preparedStatement.executeQuery();
        } catch (SQLException e) {
```

```
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    } else {
        System.out.println("Please enter your credentials");
    }
    return resultSet;   }
public int getAllTasks(int userId) throws SQLException, ClassNotFoundException {
    String query = "SELECT COUNT(*) FROM " + Const.TASKS_TABLE + " WHERE "
            + Const.USERS_ID + "=?  order by " + Const.TASKS_ID + " desc ";
    PreparedStatement preparedStatement = getDbConnection().prepareStatement(query);
    preparedStatement.setInt(1, userId);
    ResultSet resultSet = preparedStatement.executeQuery();
    while (resultSet.next()) {
        return resultSet.getInt(1);
    }
    return resultSet.getInt(1);
}
public void insertTask(Task task) {
    String insert = "INSERT INTO " + Const.TASKS_TABLE + "(" + Const.USERS_ID + ","
            + Const.TASKS_DATE + "," + Const.TASKS_DESCRIPTION + "," + Const.TASKS_TASK + ")"
            + "VALUES(?,?,?,?)";
    try {
```

```
            PreparedStatement preparedStatement = getDbConnection().prepareStatement(insert);
            System.out.println("From DBHandler UserId: " + task.getUserId());
            preparedStatement.setInt(1, task.getUserId());
            preparedStatement.setTimestamp(2, task.getDatecreated());
            preparedStatement.setString(3, task.getDescription());
            preparedStatement.setString(4, task.getTask());
            preparedStatement.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
        }
    } }
```

Configs.Class

```
package sample.Database;
public class Configs {
    protected String dbHost = "localhost";
    protected String dbPort = "3306";
    protected String dbUser = "root";
    protected String dbPass = "1234567890";
    protected String dbName = "todo";
 }
```

Const.Class

```
package sample.Database;

public class Const {

    public static final String USERS_TABLE = "users";

    public static final String TASKS_TABLE = "tasks";


    //USERS Table Column Names

    public static final String USERS_ID = "userid";

    public static final String USERS_FIRSTNAME = "firstname";

    public static final String USERS_LASTNAME = "lastname";

    public static final String USERS_PASSWORD = "password";

    public static final String USERS_USERNAME = "username";

    public static final String USERS_LOCATION = "location";

    public static final String USERS_GENDER = "gender";


    //TASKS Table Column Names

    public static final String TASKS_ID = "taskid";

    public static final String TASKS_DATE = "datecreated";

    public static final String TASKS_DESCRIPTION = "description";

    public static final String TASKS_TASK = "task";
```

Task.Class

```
package sample.model;

import java.sql.Timestamp;

public class Task {

    private int userId;

    private int taskId;

    private Timestamp datecreated;

    private String description;

    private String task;

    public Task() {

    }

    public Task(Timestamp datecreated, String description, String task, int userId) {

        this.datecreated = datecreated;

        this.description = description;

        this.task = task;

        this.userId = userId;

    }

    public Timestamp getDatecreated() {

        return datecreated;

    }

    public void setDatecreated(Timestamp datecreated) {

        this.datecreated = datecreated;

    }
```

```
 public String getDescription() {
        return description;
    }
    public void setDescription(String description) {
        this.description = description;
    }
    public String getTask() {
        return task;
    }
    public void setTask(String task) {
        this.task = task;
    }
    public int getUserId() {
        return this.userId;     }
    public void setUserId(int userId) {
        this.userId = userId;
    }
    public int getTaskId() {
        return taskId;
    }
    public void setTaskId(int taskId) {
        this.taskId = taskId;
    }
}
```

User.Class

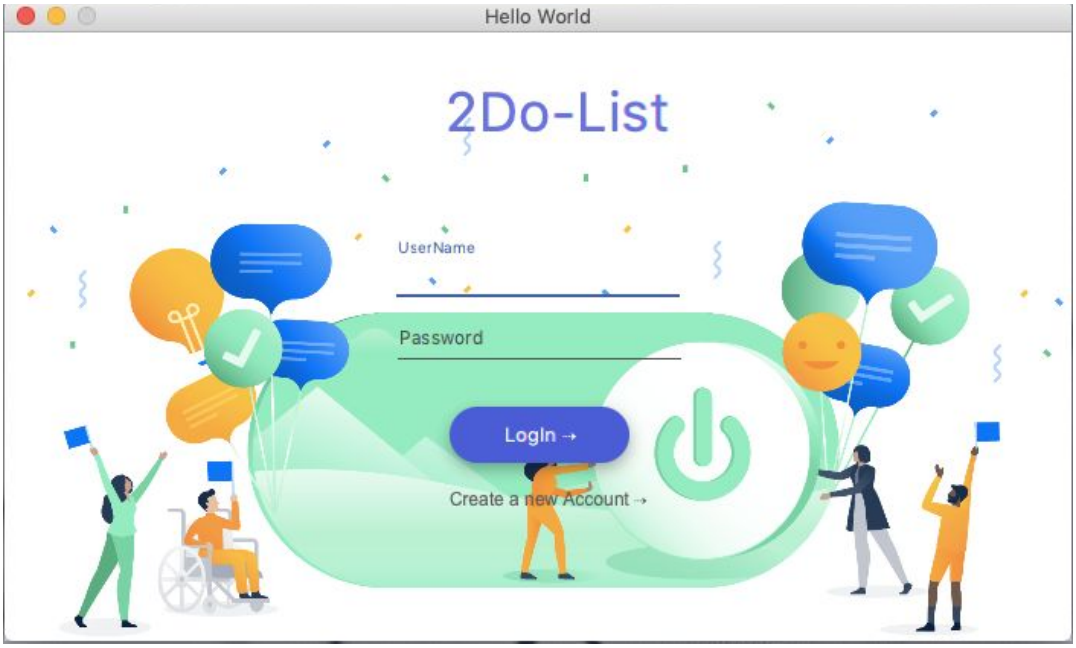
```
package sample.model;

public class User {

    private String firstName;

    private String lastName;

    private String userName;

    private String password;

    private String location;

    private String gender;

    public User() {

    }

    public User(String firstName, String lastName, String userName, String password, String location, String
gender) {

        this.firstName = firstName;

        this.lastName = lastName;

        this.userName = userName;

        this.password = password;

        this.location = location;

        this.gender = gender;

    }

    public String getFirstName() {

        return firstName;

    }

 public void setFirstName(String firstName) {
```

```
    this.firstName = firstName;
}
public String getLastName() {
    return lastName;
}
public void setLastName(String lastName) {
    this.lastName = lastName;
}
public String getUserName() {
    return userName;
}
public void setUserName(String userName) {
    this.userName = userName;
}
public String getPassword() {
    return password;
}
public void setPassword(String password) {
    this.password = password;
}
public String getLocation() {
    return location;
}
```

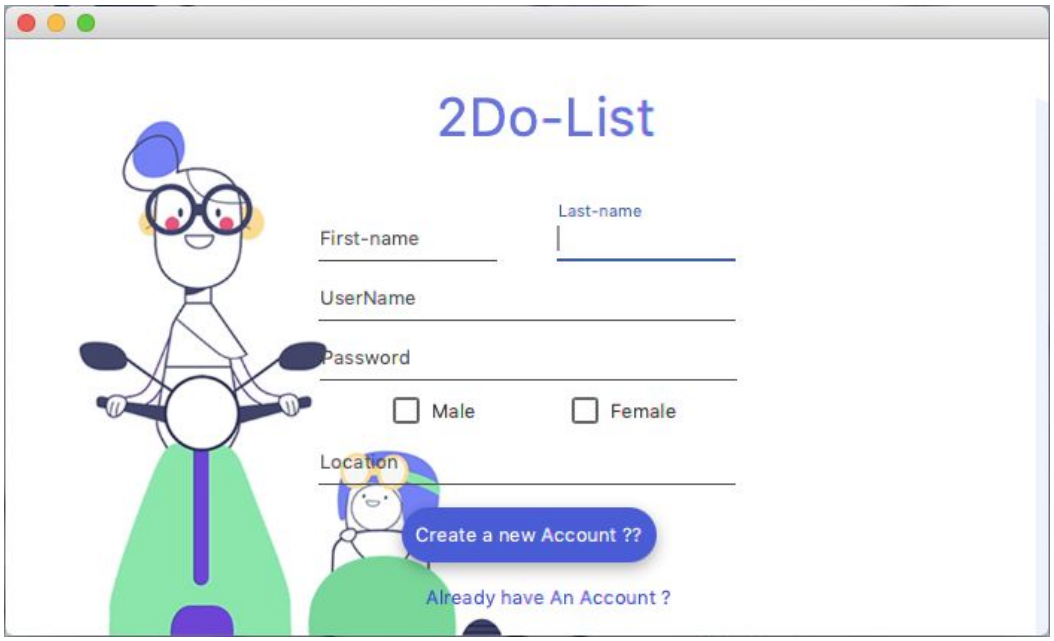
```
    public void setLocation(String location) {
        this.location = location;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
}
```

Output

Login Screen:

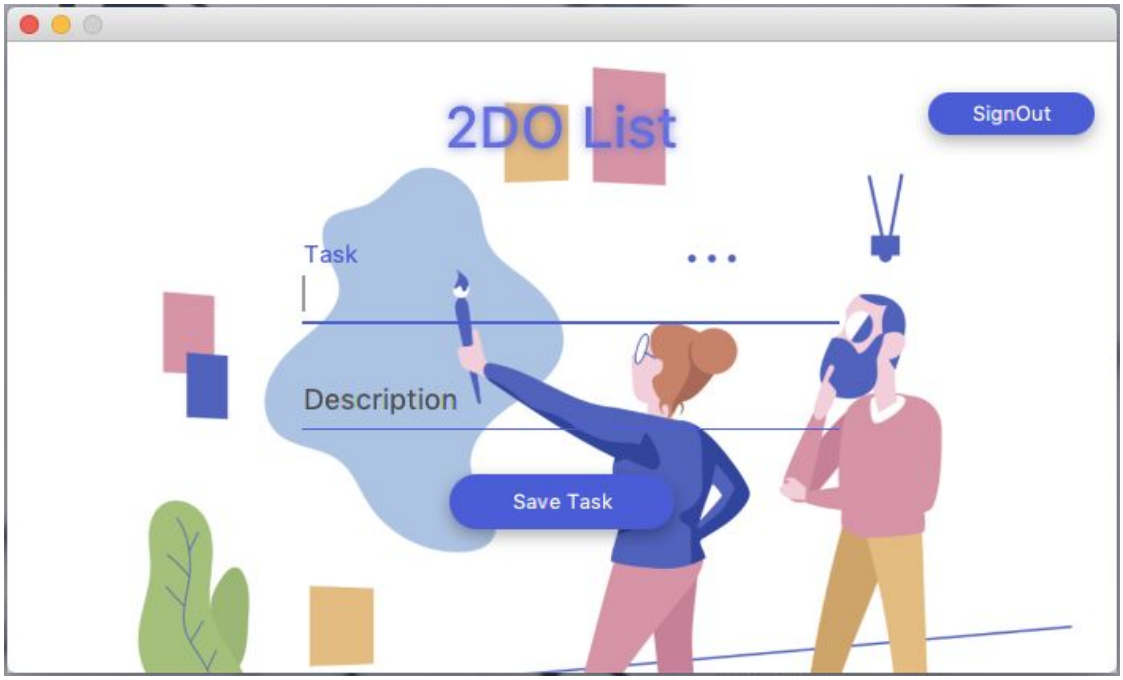


Registration Screen:

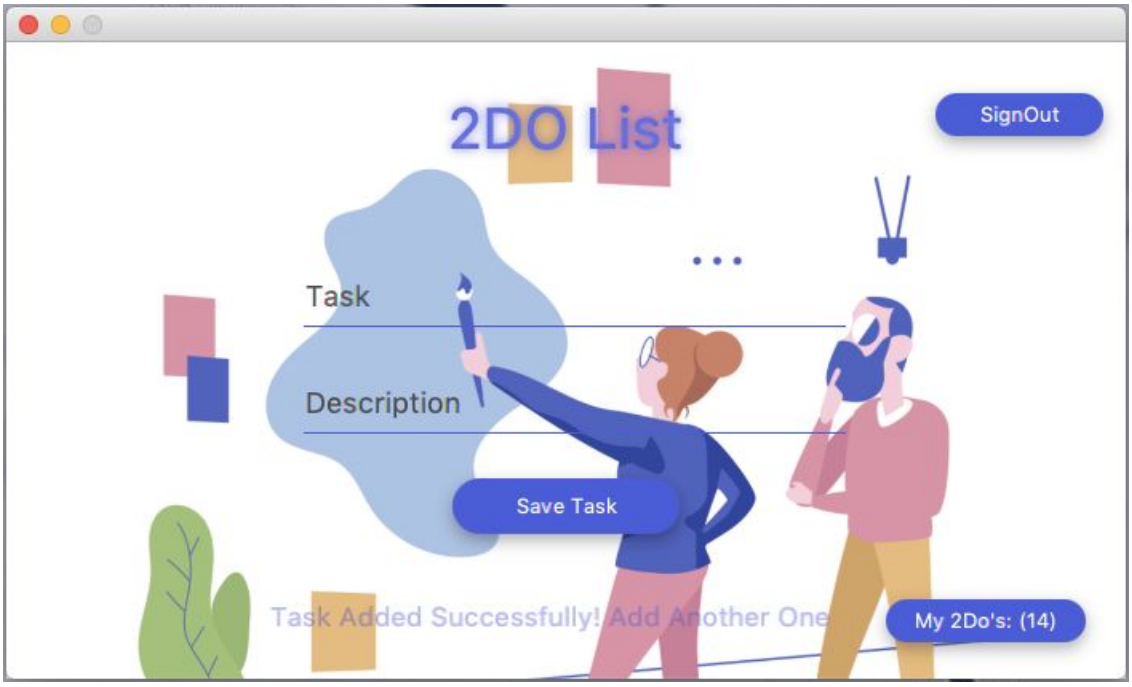


Welcome Screen:

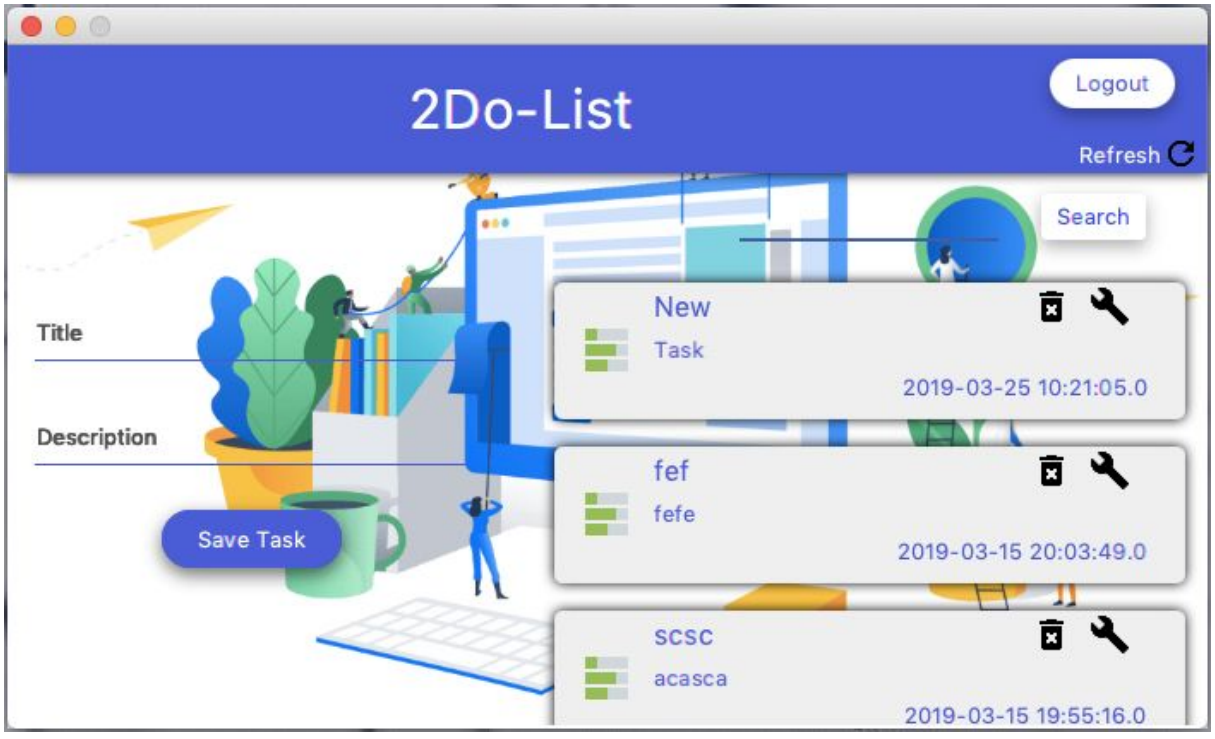
Add Task Screen



After Adding Task Screen:



List Of All Task Screen:



Search Result Screen: