5CCS2SEG COURSEWORK

TEAM Meerkat

Alexandru Enache, Rewaz Esmaeel, Julius Babenskas, Florence Anyakwo, Olufunke Sowole

---

Explanation of the Application:

This application is designed specifically for managers where they will be able to sign up and sign in. And then manage tasks, this application is a bit similar to the concept of Trello.

Once the manager signed in these functions will be available:

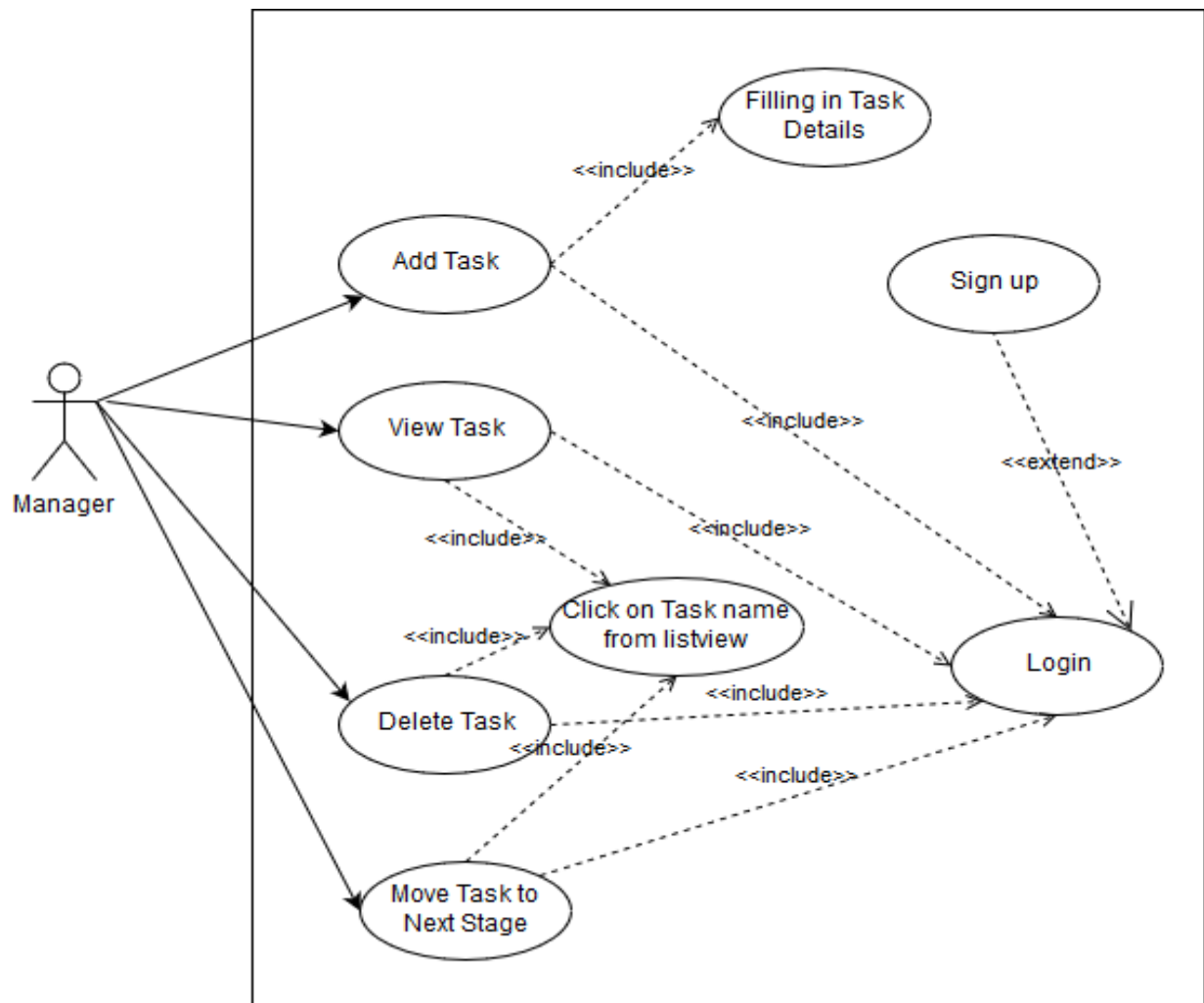- **Add Task:**

  By clicking on the ADD TASK button. A window will pop up where the managers needs to fill in **Task Details**.

- **View Task Details:**

  By clicking on the task name in the list view. A window will pop up where it has all the data of the task populated enabling the manager to edit task.

- **Edit Task:**

  By clicking on the task name in the list view. A window will pop up where the manager will be able to edit the Task Details.

- **Delete Task:**

  By clicking on the task name in the list view. A window will pop where the manager will be able to delete the Task Details by clicking on the Delete button.

- **Move Task To Next Stage:**

  Moving to next stage includes:

  > From *TO DO* to IN PROGRESS
  >
  > From IN PROGRESS to DONE

  By clicking on the task name in the list view. A window will pop where the manager needs to click on checkbox named 'move to next stage'.

Task Details:

- Name of task
- Priority
- Names of Employees
- Number of Employees
- Effort Estimate
- Start Date
- End Date

Sign up Details:

- Username
- Password
- Re-Enter Password

Sign in Details:

- Username
- Password

> Note that the Number of Employees is not required for the manager to insert but rather it is counted based on the number of employees are inserted into the "Names of Employees"
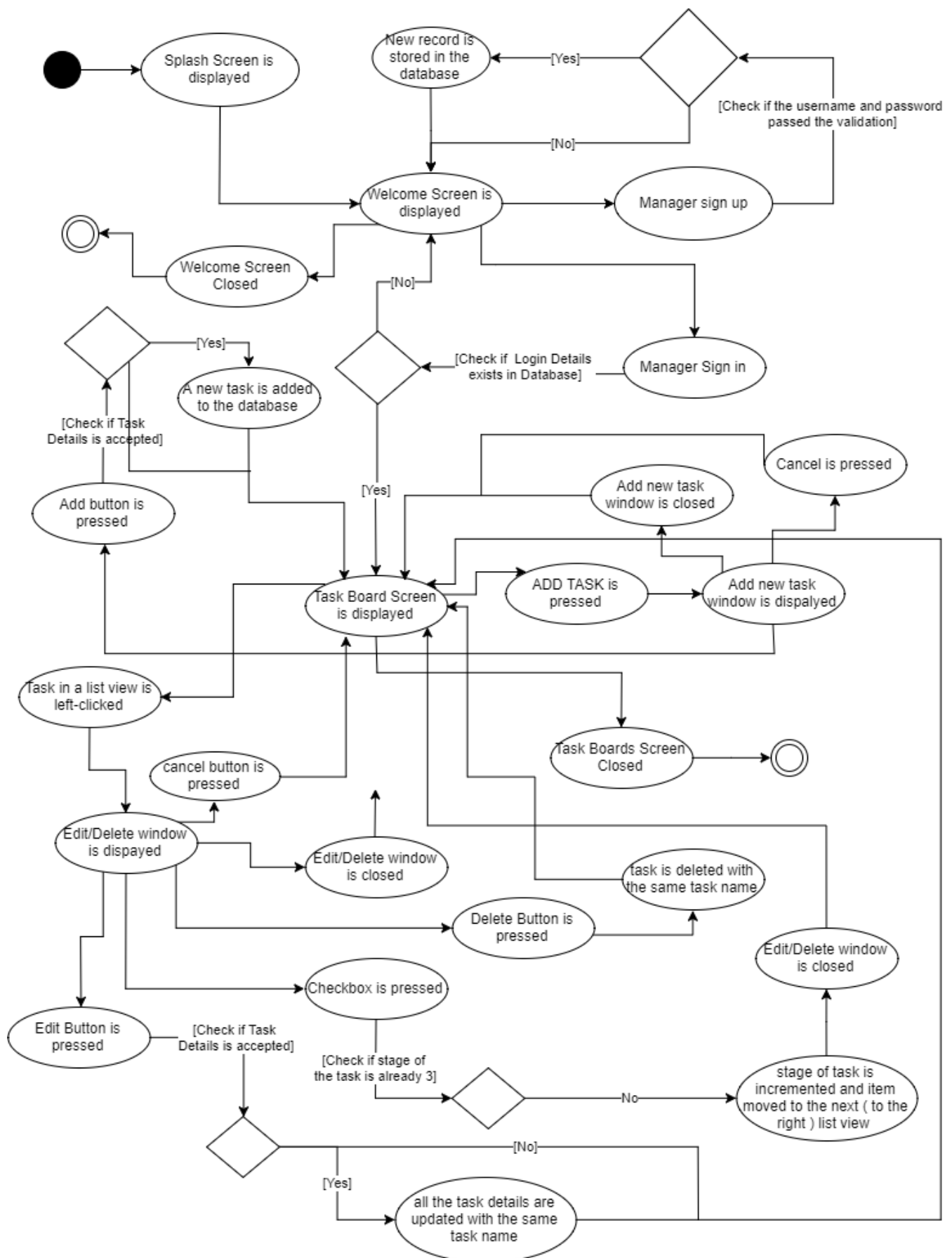
USE CASE DIAGRAM

As it can be seen from the diagram. Add Task/View Task/Delete Task/Move Task to Next Stage all include Login in so the manager can not do all these functions unless they are signed up and logined into the system.

The view Task/Delete Task/ Move Task to Next Stage required the manager to press on the task (item in the listview) first.

Activity Flow Diagram:

Class Diagram:

WelcomeController «Java Class» (controller)
- root: AnchorPane
- WelcomeController()
- initialize(URL,ResourceBundle):void
- loadSplashScreen():void

Main «Java Class» (driver)
- isSplashScreenLoaded: Boolean
- Main()
- main(String[]):void
- start(Stage):void

AddTaskModel «Java Class» (model)
- connection: Connection
- AddTaskModel()
- isDbConnected():boolean
- postTaskInDatabase(String,String,Date,Date,int):void

AddTaskController «Java Class» (main.java.controller)
- txtEffortEstimate: TextField
- txtNameOfTask: TextField
- cBoxPriority: ChoiceBox<String>
- txtNamesOfEmployees: TextField
- dateStart: DatePicker
- dateEnd: DatePicker
- btnCancel: Button
- priorityList: ObservableList<String>
- AddTaskController()
- initialize(URL,ResourceBundle):void
- cancel(ActionEvent):void
- addTask(ActionEvent):void

(-model 0..1 association)

LoginController «Java Class» (controller)
- errorMessage: Label
- txtPassword: PasswordField
- txtSignUpPassword: PasswordField
- txtReEnterPassword: PasswordField
- txtSignUpUsername: TextField
- txtUsername: TextField
- LoginController()
- initialize(URL,ResourceBundle):void
- login(ActionEvent):void
- register(ActionEvent):void

(+model 0..1 association)

LoginModel «Java Class» (model)
- connection: Connection
- LoginModel()
- isDbConnected():boolean
- isLogin(String,String):boolean
- isRegister(String,String):void

EditTaskModel «Java Class» (model)
- connection: Connection
- EditTaskModel()
- isDbConnected():boolean
- deleteTask(String):void
- updateTask(String,String,Date,Date,String,String):void
- goToNextStage(String,int):void

SqliteConnection «Java Class» (model)
- SqliteConnection()
- Connector():Connection

SqliteConnection «Java Class» (controller)
- SqliteConnection()
- Connector():Connection

EditTaskController «Java Class» (controller)
- lblNumberOfEmployees: Label
- txtEffortEstimate: TextField
- txtNameOfTask: TextField
- cBoxPriority: ChoiceBox<String>
- btnCancel: Button
- dateEnd: DatePicker
- dateStart: DatePicker
- txtNamesOfEmployees: TextField
- priorityList: ObservableList<String>
- cBoxIsDone: CheckBox
- EditTaskController()
- initialize(URL,ResourceBundle):void
- cancel(ActionEvent):void
- updateTask(ActionEvent):void
- deleteTask(ActionEvent):void
- goToNextStage(POJOTask):void
- initData(POJOTask):void
- countEmployees():void

(~model 0..1 association)

TasksManagementController «Java Class» (controller)
- listToDo: ListView<String>
- listInProgress: ListView<String>
- listDone: ListView<String>
- TasksManagementController()
- initialize(URL,ResourceBundle):void
- openAddPopUp():void
- openEditPopUp(POJOTask):void
- onClickItemToDo(MouseEvent):void
- onClickItemInProgress(MouseEvent):void
- onClickItemDone(MouseEvent):void
- populateListView(ListView<String>,int):void
- refreshListViews():void

(~model 0..1 association)

TasksManagementModel «Java Class» (model)
- connection: Connection
- TasksManagementModel()
- isDbConnected():boolean
- getTaskDetails(String):POJOTask
- populateListView(int):ObservableList<String>

POJOTask «Java Class» (model)
- taskName: String
- taskPriority: String
- nameOfEmployees: String
- startingDate: Date
- endingDate: Date
- stage: int
- effortEstimate: int
- POJOTask(String,String,String,Date,Date,int,int)
- getTaskName():String
- setTaskName(String):void
- getTaskPriority():String
- setTaskPriority(String):void
- getNameOfEmployees():String
- setNameOfEmployees(String):void
- getStartingDate():Date
- setStartingDate(Date):void
- getEndingDate():Date
- setEndingDate(Date):void
- getStage():int
- setStage(int):void
- getEffortEstimate():int
- setEffortEstimate(int):void

(~task 0..1 association)

SplashScreenController «Java Class» (controller)
- rootPane: StackPane
- SplashScreenController()
- initialize(URL,ResourceBundle):void