CMPE 321

ASSIGNMENT 3

ProjMan

A PROJECT MANAGEMENT APPLICATION

Cemal Burak Aygün

2014400072

2018 SPRING

Contents

1 INTRODUCTION

In this assignment, I implemented a **Project Management Application** with a web-based user interface, named **ProjMan**.

1.1 Overview of ProjMan

In ProjMan, login to the system is required. A person can log in to the system using his/her **username** and **password**. There are 2 user groups that can log in to ProjMan: **Admin**s and **ProjectManager**s. Both have a **first name**, **last name**, **username** and **password**. Admins are pre-defined in the system (database) and cannot be created, updated or deleted through the user interface. ProjectManagers can be created, updated and deleted by Admins through the user interface. Also, ProjectManagers can update their information through the user interface.

Besides the 2 user groups mentioned above, ProjMan also includes the following entities:

- **Project**: A project has a **name**, **start date** and **(estimated) number of task days**.
- **Task**: A task has a **name**, **start date** and **total (completion) days**.
- **Employee**: An employee has a **first name** and a **last name**.

A basic flow of the application is as follows: An Admin creates a Project and assigns some ProjectManagers to it. Then, a ProjectManager creates some Tasks for a Project assigned to him/her, and assigns some Employees (which are created by Admins) to those Tasks.

1.1.1 Authorization of Admins and ProjectManagers

Admins can:

- Create / Update / Delete a Project.
- View all the Projects in the system. (with filtering using status (complete/incomplete/all) and ProjectManager ID combinations)
- Create / Update / Delete a ProjectManager.

- View all the ProjectManagers in the system.
- Assign / Unassign a Project to a ProjectManager.
- View all the Tasks in the system.
- Delete a Task.
- Create / Update / Delete an Employee.
- View all the Employees in the system.

ProjectManagers can:

- View all the Projects (s)he is assigned to. (with filtering using status (complete/incomplete/all))
- Create / Update / Delete a Task that belongs to one of his/her Projects.
- View all the Tasks that belong to one of his/her Projects.
- Assign / Unassign an Employee to a Task that belong to one of his/her Projects..

1.1.2 Some Important Points to Remember

- When a Project is created, the ProjectManager, if any, who has the least Projects assigned to him/her is automatically assigned to the newly created Project. (See 'Trigger 2' under Database section)
- When a Task or an Employee is deleted, all the related TaskAssignments are also deleted. (See 'Trigger 1' under Database section)
- When a Project or a ProjectManager is deleted, all the related ProjectAssignments are also deleted.
- An Employee cannot be assigned more than one Task for the same date.
- Admins cannot create a Task and assign / unassign an Employee to a Task, but they can view or delete any Tasks.
- A ProjectManager can update or delete any Task that belongs to one of his/her Projects even if that Task is created by another ProjectManager of the same Project.

1.2 Technology Stack And Implementation Details

I implemented ProjMan using **Rails** (v5.1.4) web application framework on top of **Ruby** (v2.2.2). For the database, I utilized **MySQL** (v5.7.22). Also, I made use of **Bootstrap** (v3) for the user interface.

I mostly utilized the concepts and built-in functions of Rails instead of writing raw SQL. There are 2 **Stored Procedure**s and 2 **Trigger**s that are written in raw SQL. (See section 3.3)

Some constraints exist at the database-level whereas some are implemented at the Rails-level. You can see database-level constraints in db/schema.rb file and Rails-level constraints under the files stored in app/model directory.

Database-level constraints are:

- None of the fields of any relation can be **NULL**.
- **Total days** of a Task and **task days** of a Project have the default value of 0.
- No two users (Admin/ProjectManager) can have the same **username** (case-insensitive).
- Foreign-key constraints.

Rails-level constraint are:

- **First name** of an Admin / ProjectManager / Employee can contain only letters and space.
- **Last name** of an Admin / ProjectManager / Employee can contain only letters.
- **First name** and **last name** can be minimum 2 maximum 16 characters long.
- **Username** of an Admin / ProjectManager can contain only letters and/or digits. Also, it can be minimum 2 maximum 16 characters long.
- **Password** of an Admin / ProjectManager can be minimum 4 maximum 16 characters long.
- **Name** of a Task / Project can contain only letter, digit and space. Also, it can be minimum 4 maximum 256 characters long.

- An Employee cannot be assigned more than one Task for the same date.

NOTE: Some database-level constraints are repeated in Rails (not rewritten on this document) in order to provide users with convenient feedback messages in case of an constraint violation.

1.3 Initial Setup

Make sure you have **Ruby**, **Rails** and **MySQL** installed. (Note that this project was implemented using Ruby 2.2.2 and Rails 5.1.4. Other version may cause problems.) Put your database server configs and credentials into proj_man/config/database.yml file. While in the directory proj_man, run the following commands on a **Terminal** in the following order: (ignore the parenthesis)

1. bundle install (installs necessary packages (gems))
2. rails db:create (creates an empty database named **proj_man_db**)
3. rails db:schema:load (sets up database with relations, indexes, etc.)
4. rails proj_man:setup_sp_and_trigger (creates stored procedures and creates in the database)

1.3.1 Creating Admins

While in the directory proj_man, execute following command on a Terminal:

```
rails proj_man:setup_admin
```

This command creates an Admin with username=admin, password=1234, first_name=Admin, last_name=One.

You can create additional Admins using the following command:

```
rails 'proj_man:create_admin[<username>, <password>, <first_name>, <last_name>]'
```

This command creates an Admin with given parameters. For example, following command will create an Admin with username=admin2, password=1234, first_name=Admin, last_name=Two:

```
rails 'proj_man:create_admin[admin2, 1234, Admin, Two]'
```

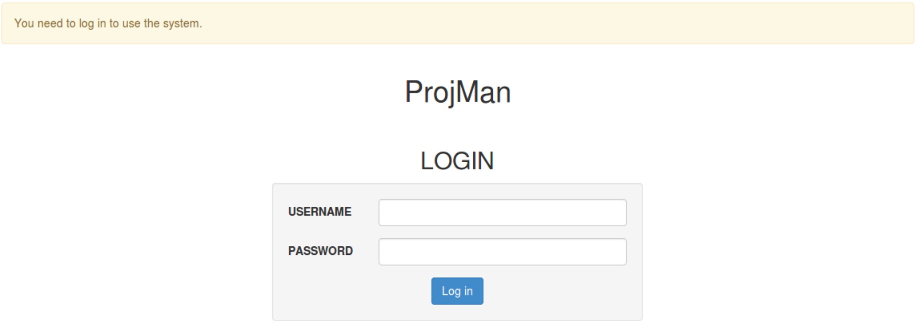
1.4 Starting the (Application) Server

While in the directory proj_man, run the following command on a **Terminal**: rails s

In Terminal, you will see the IP:PORT information on which ProjMan (server) now should be active. (In general, it is **0.0.0.0:3000**)

2 USER INTERFACE

When someone visits the application (s)he is required to log in.



You need to log in to use the system.

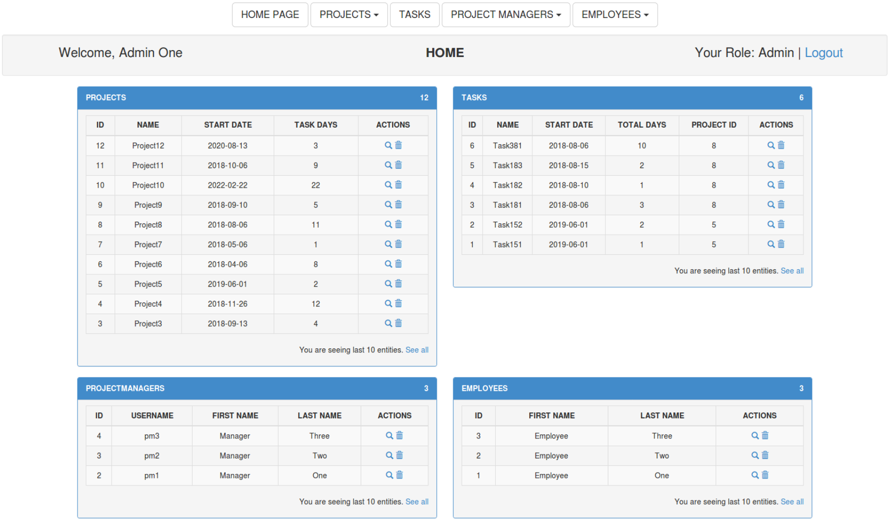
ProjMan

LOGIN

2.1 User Interface for Admins

2.1.1 Home Page

Admins can see the summary tables for the last 10 **Project**s, **Tasks**, **ProjectManager**s, and **Employee**s. They can go to the pages of the specific entities or delete them by clicking on the related icon under **ACTIONS** column.



Welcome, Admin One | **HOME** | Your Role: Admin | Logout

PROJECTS 12

ID	NAME	START DATE	TASK DAYS	ACTIONS
12	Project12	2020-08-13	3	🔍 🗑
11	Project11	2018-10-06	9	🔍 🗑
10	Project10	2022-02-22	22	🔍 🗑
9	Project9	2018-09-10	5	🔍 🗑
8	Project8	2018-08-06	11	🔍 🗑
7	Project7	2018-05-06	1	🔍 🗑
6	Project6	2018-04-06	8	🔍 🗑
5	Project5	2019-06-01	2	🔍 🗑
4	Project4	2018-11-26	12	🔍 🗑
3	Project3	2018-09-13	4	🔍 🗑

You are seeing last 10 entities. See all

TASKS 6

ID	NAME	START DATE	TOTAL DAYS	PROJECT ID	ACTIONS
6	Task381	2018-08-06	10	8	🔍 🗑
5	Task183	2018-08-15	2	8	🔍 🗑
4	Task182	2018-08-10	1	8	🔍 🗑
3	Task181	2018-08-06	3	8	🔍 🗑
2	Task152	2019-06-01	2	5	🔍 🗑
1	Task151	2019-06-01	1	5	🔍 🗑

You are seeing last 10 entities. See all

PROJECTMANAGERS 3

ID	USERNAME	FIRST NAME	LAST NAME	ACTIONS
4	pm3	Manager	Three	🔍 🗑
3	pm2	Manager	Two	🔍 🗑
2	pm1	Manager	One	🔍 🗑

You are seeing last 10 entities. See all

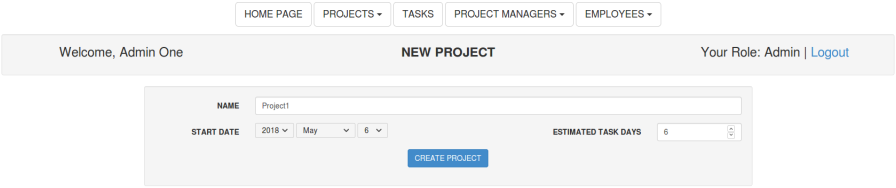
EMPLOYEES 3

ID	FIRST NAME	LAST NAME	ACTIONS
3	Employee	Three	🔍 🗑
2	Employee	Two	🔍 🗑
1	Employee	One	🔍 🗑

You are seeing last 10 entities. See all

2.1.2 Projects

2.1.2.1 Project Create Page Admins can create a Project. ("PROJECTS > Create a Project" on the menu).



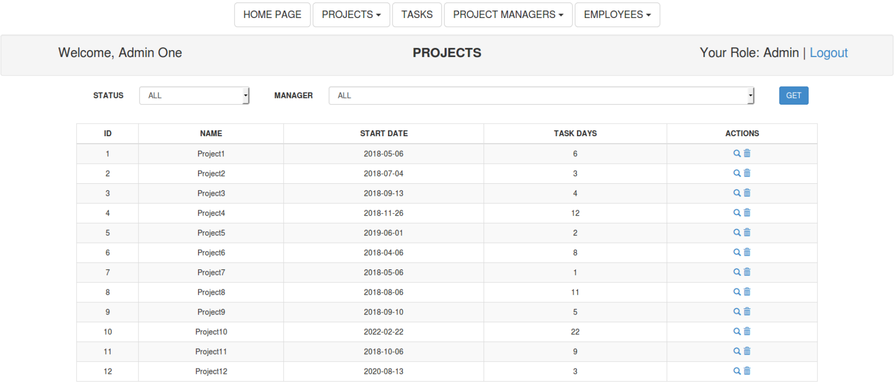
Welcome, Admin One | **NEW PROJECT** | Your Role: Admin | Logout

NAME: Project1

START DATE: 2018 May 6

ESTIMATED TASK DAYS: 6

2.1.2.2 All Projects Page Admins can view all the Projects in the system. ("PROJECTS > All Projects" on the menu) They can filter the Projects by status (complete/incomplete/all) and ProjectManager ID combinations. They can go to the pages of specific Projects or delete them by clicking on the related icon under **ACTIONS** column.

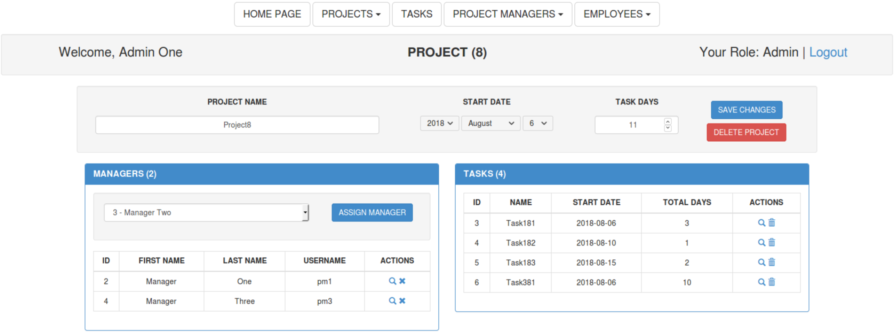


Welcome, Admin One — PROJECTS — Your Role: Admin | Logout

STATUS: ALL — MANAGER: ALL

ID	NAME	START DATE	TASK DAYS	ACTIONS
1	Project1	2018-05-06	6	🔍 🗑
2	Project2	2018-07-04	3	🔍 🗑
3	Project3	2018-09-13	4	🔍 🗑
4	Project4	2018-11-26	12	🔍 🗑
5	Project5	2019-06-01	2	🔍 🗑
6	Project6	2018-04-06	8	🔍 🗑
7	Project7	2018-05-06	1	🔍 🗑
8	Project8	2018-08-06	11	🔍 🗑
9	Project9	2018-09-10	5	🔍 🗑
10	Project10	2022-02-22	22	🔍 🗑
11	Project11	2018-10-06	9	🔍 🗑
12	Project12	2020-08-13	3	🔍 🗑

2.1.2.3 Project Page Admins can view the page of any Project in the system. On this page, they can see the related ProjectManagers and Tasks. They can also update or delete the Project and assign/unassign Project-Managers on this page.



Welcome, Admin One — PROJECT (8) — Your Role: Admin | Logout

PROJECT NAME: Project8 — START DATE: 2018 August 6 — TASK DAYS: 11

MANAGERS (2)

3 - Manager Two

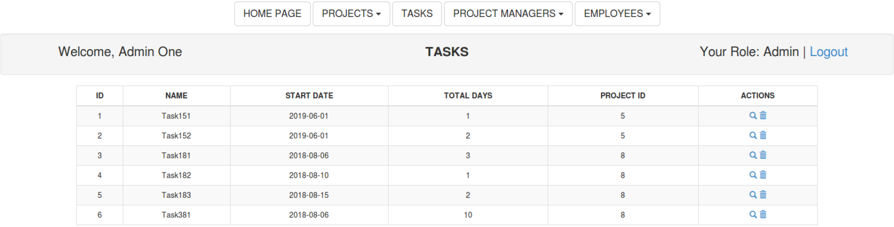
ID	FIRST NAME	LAST NAME	USERNAME	ACTIONS
2	Manager	One	pm1	🔍 ✖
4	Manager	Three	pm3	🔍 ✖

TASKS (4)

ID	NAME	START DATE	TOTAL DAYS	ACTIONS
3	Task181	2018-08-06	3	🔍 🗑
4	Task182	2018-08-10	1	🔍 🗑
5	Task183	2018-08-15	2	🔍 🗑
6	Task381	2018-08-06	10	🔍 🗑

2.1.3 Tasks

2.1.3.1 All Tasks Page Admins can view all the Tasks in the system. They can go to the pages of specific Tasks or delete them by clicking on the related icon under **ACTIONS** column.

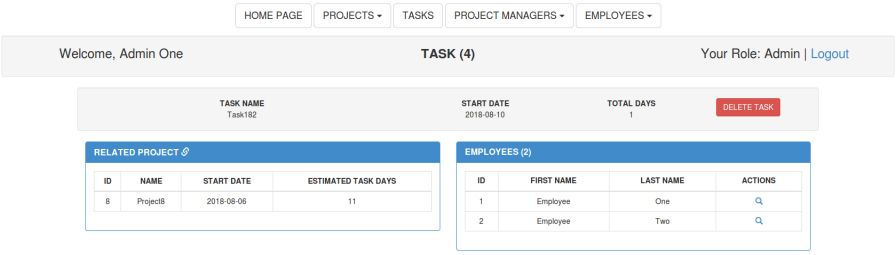


Welcome, Admin One — TASKS — Your Role: Admin | Logout

ID	NAME	START DATE	TOTAL DAYS	PROJECT ID	ACTIONS
1	Task151	2019-06-01	1	5	🔍 🗑
2	Task152	2019-06-01	2	5	🔍 🗑
3	Task181	2018-08-06	3	8	🔍 🗑
4	Task182	2018-08-10	1	8	🔍 🗑
5	Task183	2018-08-15	2	8	🔍 🗑
6	Task381	2018-08-06	10	8	🔍 🗑

2.1.3.2 Task Page

Admins can view the page of any Task in the system. On this page, they can see the related Project and Employees. They can also delete the Task on this page.



Welcome, Admin One | TASK (4) | Your Role: Admin | Logout

TASK NAME	START DATE	TOTAL DAYS
Task182	2018-08-10	1

RELATED PROJECT

ID	NAME	START DATE	ESTIMATED TASK DAYS
8	Project8	2018-08-06	11

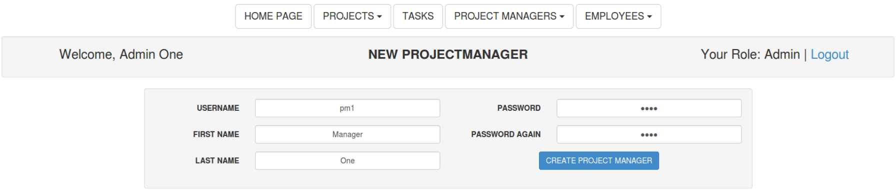
EMPLOYEES (2)

ID	FIRST NAME	LAST NAME	ACTIONS
1	Employee	One	🔍
2	Employee	Two	🔍

2.1.4 ProjectManagers

2.1.4.1 Create ProjectManager Page

Admins can create a ProjectManager. ("PROJECT MANAGERS > Create a Project Manager" on the menu).

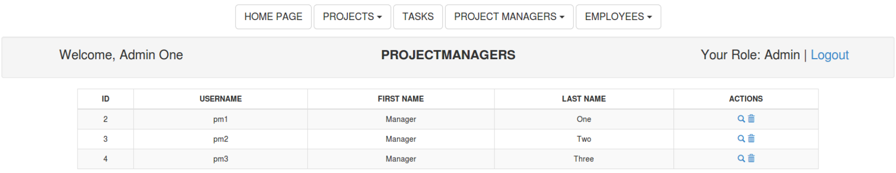


Welcome, Admin One | NEW PROJECTMANAGER | Your Role: Admin | Logout

USERNAME	pm1	PASSWORD	••••
FIRST NAME	Manager	PASSWORD AGAIN	••••
LAST NAME	One		

2.1.4.2 All ProjectManagers Page

Admins can view all the ProjectManagers in the system. ("PROJECT MANAGERS > All Project Managers" on the menu). They can go to the pages of specific ProjectManagers or delete them by clicking on the related icon under **ACTIONS** column.



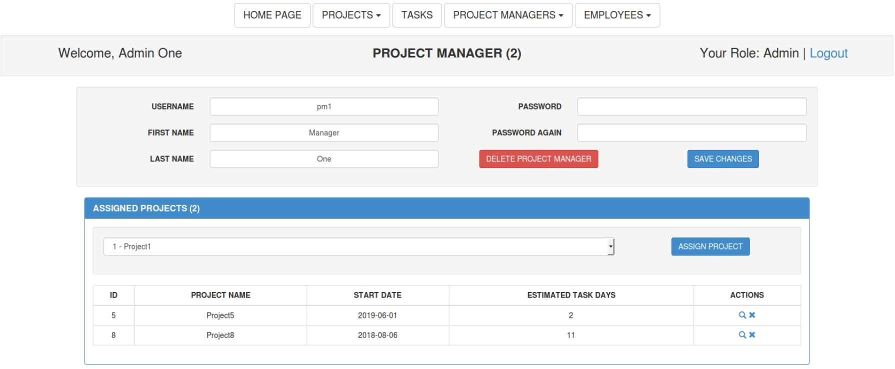
Welcome, Admin One | PROJECTMANAGERS | Your Role: Admin | Logout

ID	USERNAME	FIRST NAME	LAST NAME	ACTIONS
2	pm1	Manager	One	🔍 🗑
3	pm2	Manager	Two	🔍 🗑
4	pm3	Manager	Three	🔍 🗑

2.1.4.3 ProjectManager Page

Admins can view the page of any ProjectManager in the system. On this page, they can see the assigned Projects.

They can also update or delete the ProjectManager and assign/unassign Projects on this page.



Welcome, Admin One

PROJECT MANAGER (2)

Your Role: Admin | Logout

USERNAME	pm1	PASSWORD	
FIRST NAME	Manager	PASSWORD AGAIN	
LAST NAME	One		

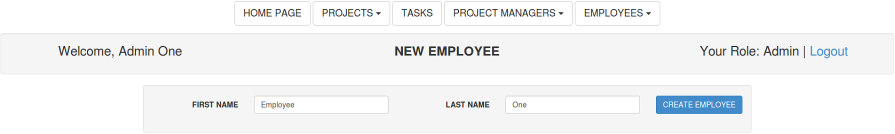
ASSIGNED PROJECTS (2)

1 - Project1

ID	PROJECT NAME	START DATE	ESTIMATED TASK DAYS	ACTIONS
5	Project5	2019-06-01	2	🔍 ✖
8	Project8	2018-08-06	11	🔍 ✖

2.1.5 Employees

2.1.5.1 Create Employee Page Admins can create an Employee. ("EMPLOYEES > Create an Employee" on the menu).



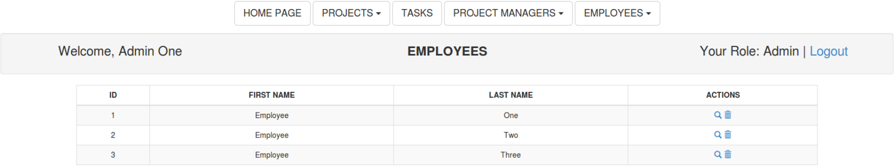
Welcome, Admin One

NEW EMPLOYEE

Your Role: Admin | Logout

FIRST NAME Employee LAST NAME One

2.1.5.2 All Employees Page Admins can view all the Employees in the system. ("EMPLOYEES > All Employees" on the menu). They can go to the pages of specific Employees or delete them by clicking on the related icon under **ACTIONS** column.



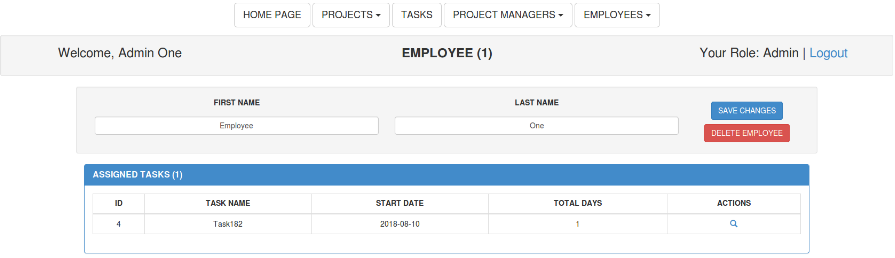
Welcome, Admin One

EMPLOYEES

Your Role: Admin | Logout

ID	FIRST NAME	LAST NAME	ACTIONS
1	Employee	One	🔍 🗑
2	Employee	Two	🔍 🗑
3	Employee	Three	🔍 🗑

2.1.5.3 Employee Page Admins can view the page of any Employee in the system. On this page, they can see the assigned Tasks. They can also update or delete the Employee on this page.



Welcome, Admin One | **EMPLOYEE (1)** | Your Role: Admin | Logout

FIRST NAME: Employee

LAST NAME: One

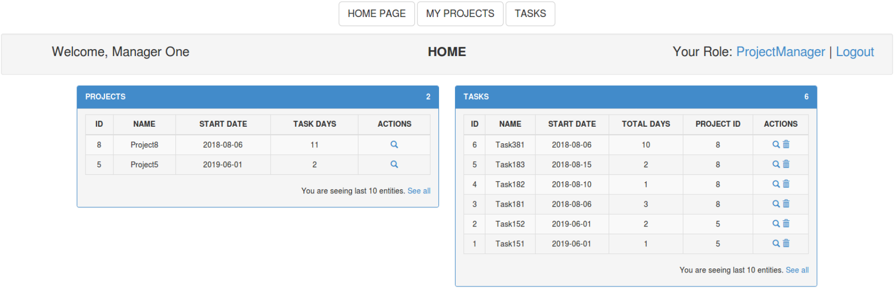
ASSIGNED TASKS (1)

ID	TASK NAME	START DATE	TOTAL DAYS	ACTIONS
4	Task182	2018-08-10	1	🔍

2.2 User Interface ProjectManagers

2.2.1 Home Page

ProjectManagers can see the summary tables for the last 10 **Project**s that are assigned to them and **Tasks** that belong to their assigned Projects. They can go to the pages of the specific entities or delete them (only Tasks) by clicking on the related icon under **ACTIONS** column.



Welcome, Manager One | **HOME** | Your Role: ProjectManager | Logout

PROJECTS 2

ID	NAME	START DATE	TASK DAYS	ACTIONS
8	Project8	2018-08-06	11	🔍
5	Project5	2019-06-01	2	🔍

You are seeing last 10 entities. See all

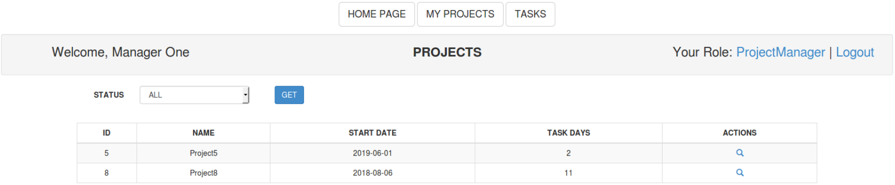
TASKS 6

ID	NAME	START DATE	TOTAL DAYS	PROJECT ID	ACTIONS
6	Task381	2018-08-06	10	8	🔍 🗑
5	Task183	2018-08-15	2	8	🔍 🗑
4	Task182	2018-08-10	1	8	🔍 🗑
3	Task181	2018-08-06	3	8	🔍 🗑
2	Task152	2019-06-01	2	5	🔍 🗑
1	Task151	2019-06-01	1	5	🔍 🗑

You are seeing last 10 entities. See all

2.2.2 Projects

2.2.2.1 All Projects Page ProjectManagers can view all the Projects that are assigned to them in the system. They can filter the Projects by status (complete/incomplete/all). They can go to the pages of specific Projects by clicking on the related icon under **ACTIONS** column.



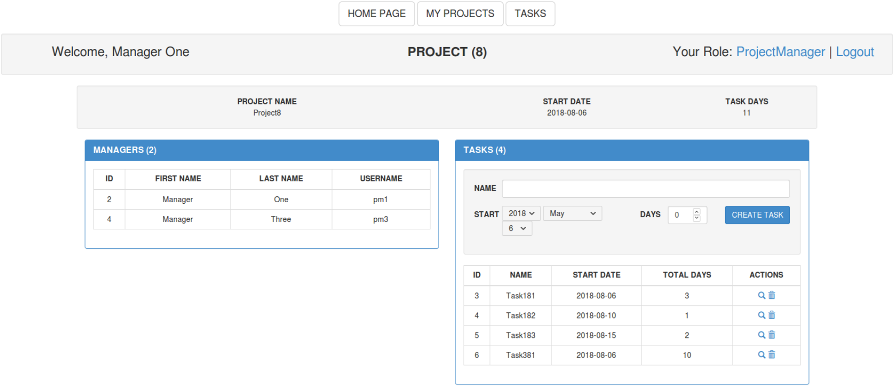
Welcome, Manager One | **PROJECTS** | Your Role: ProjectManager | Logout

STATUS: ALL

ID	NAME	START DATE	TASK DAYS	ACTIONS
5	Project5	2019-06-01	2	🔍
8	Project8	2018-08-06	11	🔍

2.2.2.2 Project Page

ProjectManagers can view the page of any Project that is assigned to them in the system. On this page, they can see the related ProjectManagers and Tasks. They can also create new Tasks for the Project on this page.



Welcome, Manager One

PROJECT (8)

Your Role: ProjectManager | Logout

PROJECT NAME	START DATE	TASK DAYS
Project8	2018-08-06	11

MANAGERS (2)

ID	FIRST NAME	LAST NAME	USERNAME
2	Manager	One	pm1
4	Manager	Three	pm3

TASKS (4)

NAME:

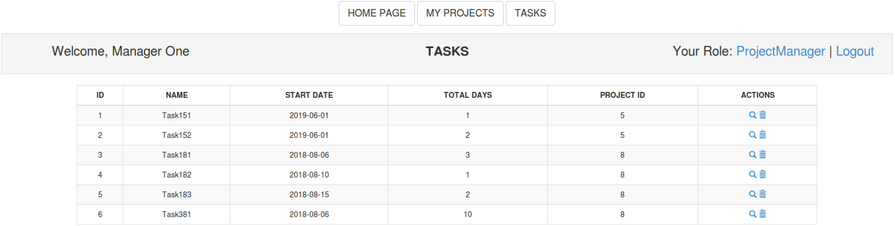
START: 2018 / May / 6 DAYS: 0

ID	NAME	START DATE	TOTAL DAYS	ACTIONS
3	Task181	2018-08-06	3	🔍 🗑
4	Task182	2018-08-10	1	🔍 🗑
5	Task183	2018-08-15	2	🔍 🗑
6	Task381	2018-08-06	10	🔍 🗑

2.2.3 Tasks

2.2.3.1 All Tasks Page

ProjectManagers can view all the Tasks that belong to any of their Projects in the system. They can go to the pages of specific Tasks or delete them by clicking on the related icon under **ACTIONS** column.



Welcome, Manager One

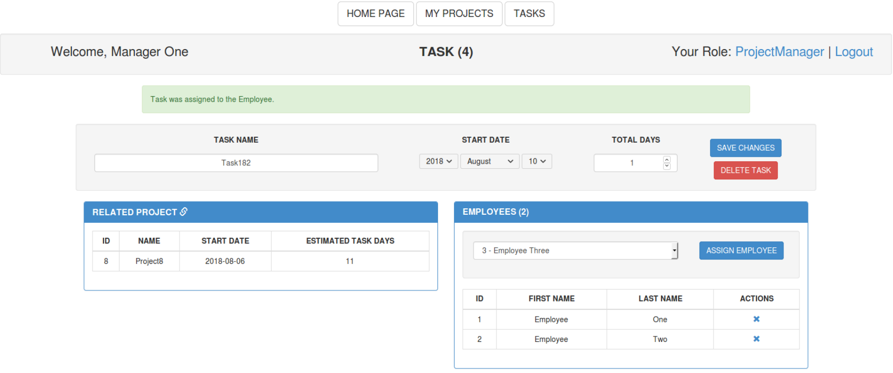
TASKS

Your Role: ProjectManager | Logout

ID	NAME	START DATE	TOTAL DAYS	PROJECT ID	ACTIONS
1	Task151	2019-06-01	1	5	🔍 🗑
2	Task152	2019-06-01	2	5	🔍 🗑
3	Task181	2018-08-06	3	8	🔍 🗑
4	Task182	2018-08-10	1	8	🔍 🗑
5	Task183	2018-08-15	2	8	🔍 🗑
6	Task381	2018-08-06	10	8	🔍 🗑

2.2.3.2 Task Page

ProjectManagers can view the page of any Task that belongs to any of their Projects in the system. On this page, they can see the related Project and Employees. They can also update or delete the Task and assign / unassign an Employee on this page. (On this screenshot, a success message is seen which is displayed after a successful Task creation.)



Welcome, Manager One

TASK (4)

Task was assigned to the Employee.

TASK NAME	START DATE	TOTAL DAYS
Task182	2018 August 10	1

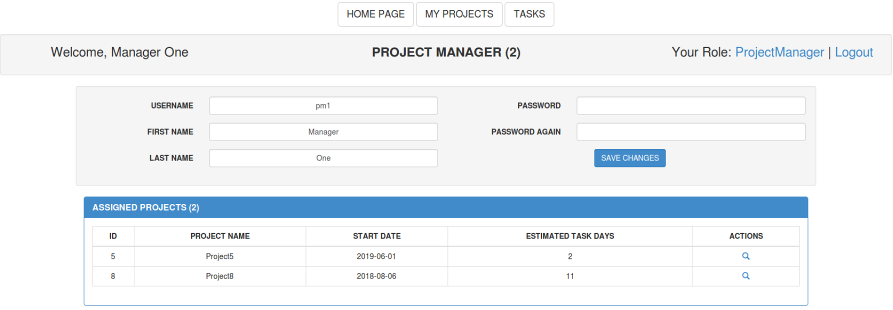
ID	NAME	START DATE	ESTIMATED TASK DAYS
8	Project8	2018-08-06	11

3 - Employee Three

ID	FIRST NAME	LAST NAME	ACTIONS
1	Employee	One	✖
2	Employee	Two	✖

2.2.4 ProjectManagers

2.2.4.1 ProjectManager Page ProjectManagers can view only their pages by clicking the "ProjectManager" text near the "Your Role:" in the header. On this page, they can see all the Projects assigned to them. Also, they can update their information.



Welcome, Manager One

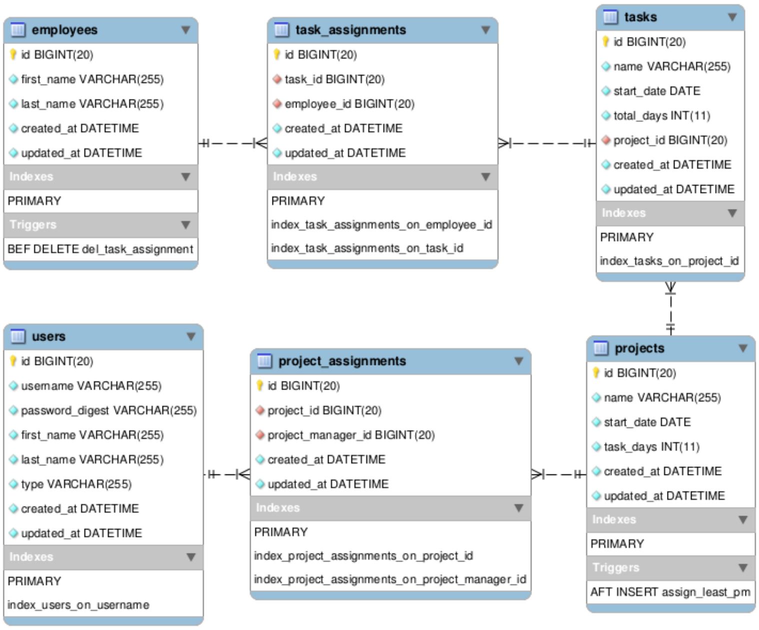
PROJECT MANAGER (2)

USERNAME	pm1	PASSWORD	
FIRST NAME	Manager	PASSWORD AGAIN	
LAST NAME	One		

ID	PROJECT NAME	START DATE	ESTIMATED TASK DAYS	ACTIONS
5	Project5	2019-06-01	2	🔍
8	Project8	2018-08-06	11	🔍

3 DATABASE

3.1 Relations and Diagram



In Rails convention, entities are represented by **Model**s. A model named MyModel has a corresponding table named my_models in the database. Every model has a primary key called id which is a unique integer. Also, every model has those 2 **DateTime** fields: created_at (stores the date and time the record is created) and updated_at (stores the date and time of last modification of the record). I went along with this convention.

3.1.1 Project

A Project belongs to many ProjectManagers (through ProjectAssignment model) and it has many Tasks.

- name: A brief description of the Project.
- start_date: Start date of the Project.
- task_days: Estimated task days of the Project.

3.1.2 Task

A Task belongs to many Employees (through TaskAssignment model) and it belongs to (only) one Project.

- name: A brief description of the Task.
- start_date: Start date of the Task.
- total_days: Total completion days of the Task.
- project_id: ID of the Project this Task belongs to.

3.1.3 User

A User Model encapsulates the 'Admin' and 'ProjectManager' Models. A ProjectManager has many Projects (through ProjectAssignment model).

- username: Username of the User.
- password_digest: Hash value of the password of the User.
- first_name: First name of the User.
- last_name: Last name of the User.
- type: Role of the User, either 'Admin' or 'ProjectManager'.

NOTE: In database, there are not such relations as 'Admin' and 'ProjectManager'. Both of these are represented in the relation 'User'. In Rails, there are Models called 'Admin' and 'ProjectManager'. Both of them inherits from the Model 'User'.

3.1.4 Employee

An Employee has many Tasks (through TaskAssignment model).

- first_name: First name of the Employee.
- last_name: Last name of the Employee.

3.1.5 TaskAssignment

This model represents an assignment of an Employee to a Task.

- task_id: ID of the Task this assignment is related.
- employee_id: ID of the Employee this assignment is related.

3.1.6 ProjectAssignment

This model represents an assignment of a ProjectManager to a Project.

- project_id: ID of the Project this assignment is related.
- project_manager_id: ID of the ProjectManager this assignment is related.

3.2 Index Files

There are index files for the following columns (fields) in the database:

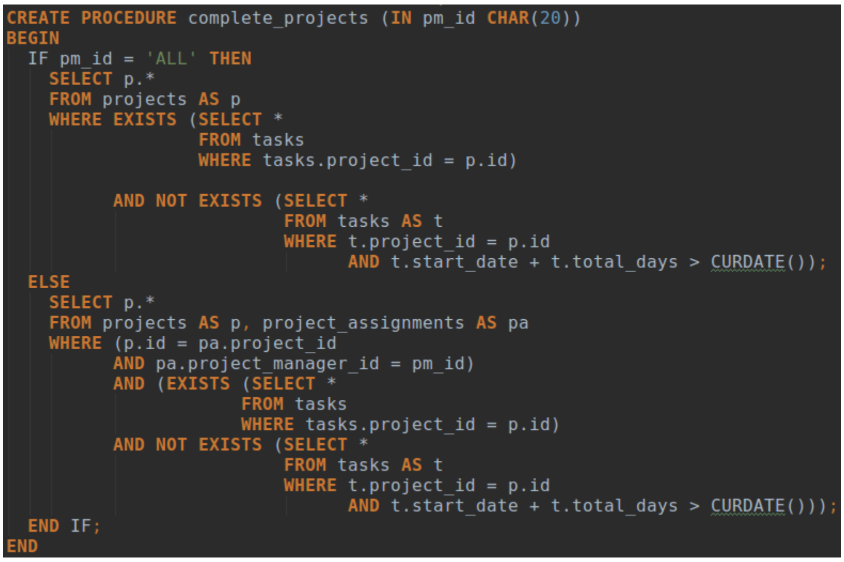
- **project_id** column of **project_assignments** table
- **project_manager_id** column of **project_assignments** table
- **task_id** column of **task_assignments** table
- **employee_id** column of **task_assignments** table
- **project_id** column of **tasks** table
- **username** column of **users** table

3.3 Stored Procedures and Triggers

Stored Procedures and Trigger are written in raw SQL in lib/tasks/proj_man.rake file.

3.3.1 Stored Procedures

3.3.1.1 SP 1: complete_projects(pm_id) This Stored Procedure takes a ProjectManager ID (or 'ALL') as the only argument and returns the complete Projects of that ProjectManager (or all the complete Projects in the system). A Projects is considered as 'complete' if there are at least 1 Task belongs to it and all the Tasks are past-dated.



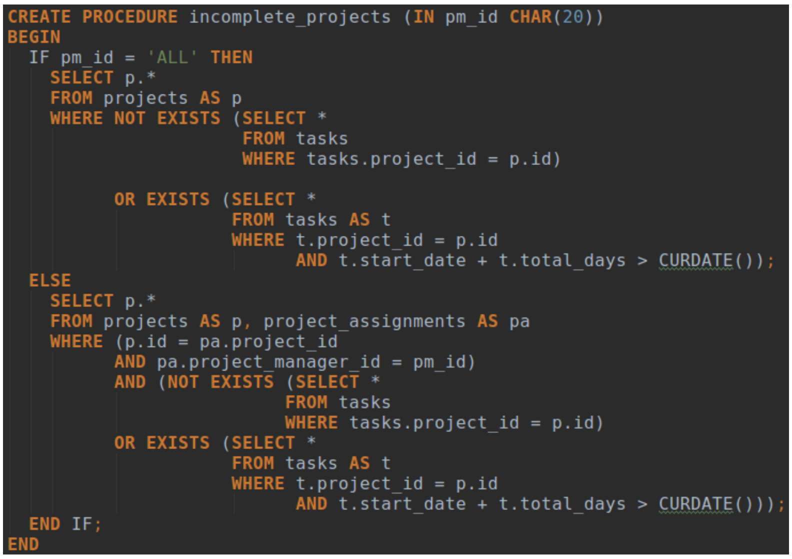
```
CREATE PROCEDURE complete_projects (IN pm_id CHAR(20))
BEGIN
  IF pm_id = 'ALL' THEN
    SELECT p.*
    FROM projects AS p
    WHERE EXISTS (SELECT *
                  FROM tasks
                  WHERE tasks.project_id = p.id)

          AND NOT EXISTS (SELECT *
                          FROM tasks AS t
                          WHERE t.project_id = p.id
                                AND t.start_date + t.total_days > CURDATE());
  ELSE
    SELECT p.*
    FROM projects AS p, project_assignments AS pa
    WHERE (p.id = pa.project_id
          AND pa.project_manager_id = pm_id)
          AND (EXISTS (SELECT *
                      FROM tasks
                      WHERE tasks.project_id = p.id)
          AND NOT EXISTS (SELECT *
                          FROM tasks AS t
                          WHERE t.project_id = p.id
                                AND t.start_date + t.total_days > CURDATE()));
  END IF;
END
```

IF-BRANCH in the SP, handles the parameter with value 'ALL'. **EXISTS CLAUSE** in **WHERE CLAUSE** checks if the Project has at least 1 Task and **NOT EXISTS CLAUSE** in **WHERE CLAUSE** checks if all the Tasks are past-dated.

ELSE-BRANCH handles the parameters with the value of an ID of a ProjectManager. It works like **IF-BRANCH**. In addition, it limits the Projects to ones that assigned to the given ProjectManager (ID).

3.3.1.2 SP 2: incomplete_projects(pm_id) This Stored Procedure takes a ProjectManager ID (or 'ALL') as the only argument and returns the incomplete Projects of that ProjectManager (or all the incomplete Projects in the system). A Projects is considered as 'incomplete' if either there are no Task belongs to that Project or at least one of its Tasks is not past-dated yet.



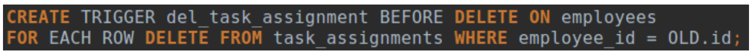
```
CREATE PROCEDURE incomplete_projects (IN pm_id CHAR(20))
BEGIN
  IF pm_id = 'ALL' THEN
    SELECT p.*
    FROM projects AS p
    WHERE NOT EXISTS (SELECT *
                      FROM tasks
                      WHERE tasks.project_id = p.id)

          OR EXISTS (SELECT *
                     FROM tasks AS t
                     WHERE t.project_id = p.id
                           AND t.start_date + t.total_days > CURDATE());
  ELSE
    SELECT p.*
    FROM projects AS p, project_assignments AS pa
    WHERE (p.id = pa.project_id
          AND pa.project_manager_id = pm_id)
          AND (NOT EXISTS (SELECT *
                           FROM tasks
                           WHERE tasks.project_id = p.id)
          OR EXISTS (SELECT *
                     FROM tasks AS t
                     WHERE t.project_id = p.id
                           AND t.start_date + t.total_days > CURDATE()));
  END IF;
END
```

First, ProjectAssignments are grouped by project_manager_id. Then, Projects of every ProjectManager is counted. Finally, results are sorted in ascending order and the first ProjectManager is selected.

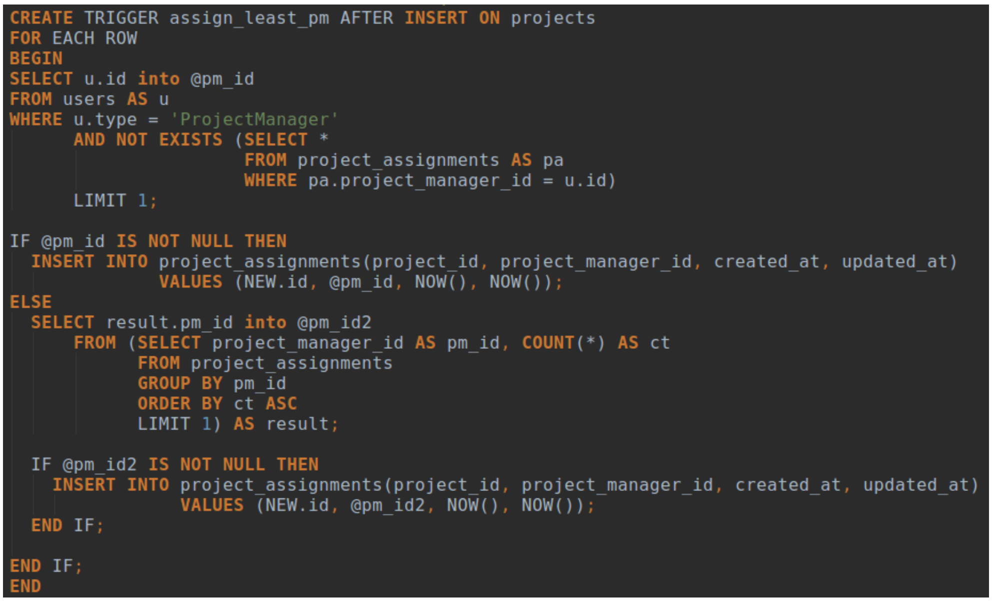
3.3.2 Triggers

3.3.2.1 Trigger 1: del_task_assignment This Trigger executes when an Employee is deleted from the database and deletes the related TaskAssignment from the database.



```
CREATE TRIGGER del_task_assignment BEFORE DELETE ON employees
FOR EACH ROW DELETE FROM task_assignments WHERE employee_id = OLD.id;
```

3.3.2.2 Trigger 2: assign_least_pm This Trigger executes when a Project is inserted (created) into the database. It finds the Employee who has the least Projects assigned to him/her and assigns the newly created Project to that ProjectManager by creating a ProjectAssignment record.



```sql
CREATE TRIGGER assign_least_pm AFTER INSERT ON projects
FOR EACH ROW
BEGIN
SELECT u.id into @pm_id
FROM users AS u
WHERE u.type = 'ProjectManager'
      AND NOT EXISTS (SELECT *
                      FROM project_assignments AS pa
                      WHERE pa.project_manager_id = u.id)
      LIMIT 1;

IF @pm_id IS NOT NULL THEN
  INSERT INTO project_assignments(project_id, project_manager_id, created_at, updated_at)
              VALUES (NEW.id, @pm_id, NOW(), NOW());
ELSE
  SELECT result.pm_id into @pm_id2
      FROM (SELECT project_manager_id AS pm_id, COUNT(*) AS ct
            FROM project_assignments
            GROUP BY pm_id
            ORDER BY ct ASC
            LIMIT 1) AS result;

  IF @pm_id2 IS NOT NULL THEN
    INSERT INTO project_assignments(project_id, project_manager_id, created_at, updated_at)
                VALUES (NEW.id, @pm_id2, NOW(), NOW());
  END IF;

END IF;
END
```

First, checks if there are any ProjectManagers that has zero Projects. If so, select him/her. If not, all the ProjectAssignments are grouped by ProjectManager (ID). Then, elements of eachs group is counted. Then, results are sorted in ascending order and the first ProjectManager (ID) is selected. Finally, if a ProjectManager is found, a new ProjectAssignment is created which assigns the newly created Project to the selected ProjectManager.

4 CONCLUSION

In this assignment, I implemented a **Project Management Application** with a web-based user interface, named **ProjMan**. ProjMan allows 2 distinct user groups, Admins and ProjectManagers, to log in and use the system. Basic usage scenario of ProjMan is as follows: An Admin creates a Project and assigns it some ProjectManagers. Then, a ProjectManager creates some Tasks for a Project (s)he is assigned to and assignes some Employees (which are created by Admins) to those Tasks.

I utilized Rails (v5.1.4) web framework on top of Ruby (v2.2.2) to implement ProjMan. For the database, I used MySQL (v5.7.22). Also, I utilized Bootstrap (v3) for the user interface. I had some experience with all of these technologies/tools before this assignment. With this assignment, I faced with new challanges and enhanced my skills of these tools. Especially, writing raw SQL for Stored Procedures and Triggers increased my confidencse with SQL.

I tried to find the most convenient ways while implementing ProjMan. I think, I created a user-friendly and easy-to-use interface. It allows related actions to be performed on a single page. For example, when a ProjectManager visits a specific Project page, (s)he can see all the related Tasks on that page. Also, (s)he can add new Tasks to the Project (or delete existing Tasks from the Project) right on that page.