Bilkent University

Department of Computer Engineering

CS 353 - Database Systems

Term Project - Final Report

Project Name: Tasks & Managers

Group No: 26

Group Members: Cihan Erkan
Merve Sağyatanlar
Çağla Sözen
Murat Tüver

Table of Contents

1. System Description

In this project, we have designed and implemented a project tracking system that aims to aid project management by representing the hierarchical structures of companies as well as the structure of projects. This system provides a user-friendly interface to track the progress of projects by structuring them into items such as boards, lists, cards. Cards and lists may be added, removed and updated in a simple manner using the user-interface.Although this state of the project does not fully provide the pledged functionalities, it provides the essentials.

2. E/R Model

2.1. Modifications

For clarification of the system structure and correction of the ER Diagram in the Design Phase, following modifications were made,

- *has* relation between **Team** and **Project** was removed because of its redundancy. For illustrating this relation, the aggregate relation **WorkOn** was used.
- *has* relation between **Card** and **Poll** was changed to be **1-to-many** as advised by our Teaching Assistant.
- *profession* attribute was removed from the **User** entity because of its redundancy for the system logic.
- *description* attribute was removed from the **Team** entity because of its redundancy for the system logic.

Furthermore,for correction of the schema with respect to the Final ER Diagram, following modifications were made,

- *WorkOn* schema was added to illustrate the aggregate relation as advised by our Teaching Assistant.
- *WorkerResponse* schema was added to illustrate the relation between Workers and Poll Response as advised by our Teaching Assistant.
- *Prereq* schema was added to illustrate the recursive relation between Cards.

The final states of the ER Diagram and Relational Schemas can be found in *Sections 2.2. and 3.*

2.2. Final E/R Diagram

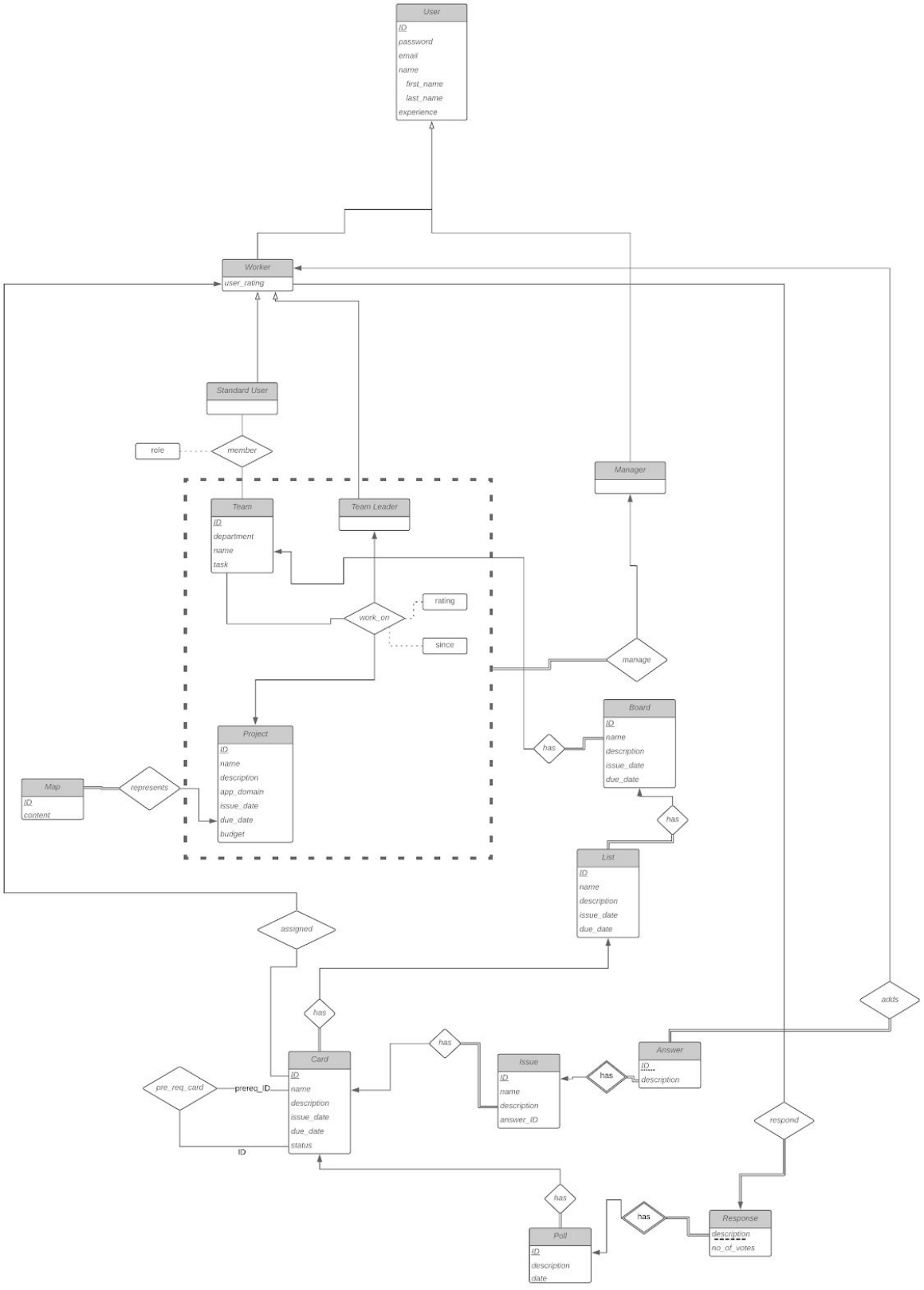


Figure 1: ER Diagram

3. Relational Schemas

3.1. User

Relational Model: User(id,email, password, first_name, last_name, experience)
Table Definition:

```
CREATE TABLE User(
        id INTEGER PRIMARY KEY AUTO_INCREMENT,
        email VARCHAR(50) NOT NULL UNIQUE,
        password VARCHAR(50) NOT NULL,
        first_name VARCHAR(50),
        last_name VARCHAR(50),
        experience INTEGER
);
```

3.2. Worker

Relational Model: Worker(ID, user_rating)
Table Definition:

```
CREATE TABLE Worker(
        ID INTEGER PRIMARY KEY,
        user_rating NUMERIC(1,0),
        FOREIGN KEY(ID) REFERENCES User(ID)
);
```

3.3. Standard User

Relational Model: StandardUser(ID)
Table Definition:

```
CREATE TABLE StandardUser(
        ID INTEGER PRIMARY KEY,
        FOREIGN KEY(ID) REFERENCES User(ID)
);
```

3.4. Manager

Relational Model: Manager(ID)
Table Definition:

```
CREATE TABLE Manager(
        ID INTEGER PRIMARY KEY,
        FOREIGN KEY(ID) REFERENCES User(ID)
);
```

3.5. Team Leader

Relational Model: TeamLeader(ID)
Table Definition:

```
CREATE TABLE TeamLeader(
        ID INTEGER PRIMARY KEY,
        FOREIGN KEY(ID) REFERENCES User(ID)
);
```

3.6. Team

Relational Model: Team(ID, department,task, name)
Table Definition:

```
CREATE TABLE Team(
        ID INTEGER PRIMARY KEY AUTO_INCREMENT,
        department VARCHAR(50),
        task VARCHAR(100),
        name VARCHAR(50) NOT NULL
);
```

3.7. Project

Relational Model: Project(ID, name, description, app_domain, issue_date, due_date, budget)
Table Definition:

```
CREATE TABLE Project(
        ID INTEGER PRIMARY KEY AUTO_INCREMENT,
        name VARCHAR(50) NOT NULL,
        description VARCHAR(100),
        app_domain VARCHAR(50),
        issue_date DATETIME,
        due_date DATETIME,
        budget INTEGER
);
```

3.8. Map

Relational Model: Map(ID, content, project_ID)
Table Definition:

```
CREATE TABLE Map(
```

```
        ID INTEGER PRIMARY KEY AUTO_INCREMENT,
        content VARCHAR(50) NOT NULL,
        project_ID INTEGER NOT NULL,
        FOREIGN KEY (project_ID) REFERENCES Project(ID)
);
```

3.9. Board

Relational Model: Board(ID, name, description, issue_date, due_date, team_ID)
Table Definition:

```
    CREATE TABLE Board(
        ID INTEGER PRIMARY KEY AUTO_INCREMENT,
        name VARCHAR(50) NOT NULL,
        description VARCHAR(100),
        issue_date DATETIME NOT NULL,
        due_date DATETIME,
        team_ID INTEGER NOT NULL,
        FOREIGN KEY (team_ID) REFERENCES Team(ID)
);
```

3.10. List

Relational Model: List(ID, name, description, issue_date, due_date, board_ID)
Table Definition:

```
    CREATE TABLE List(
        ID INTEGER PRIMARY KEY AUTO_INCREMENT,
        name VARCHAR(50) NOT NULL,
        description VARCHAR(100),
        issue_date DATETIME NOT NULL,
        due_date DATETIME,
        board_ID INTEGER NOT NULL,
        FOREIGN KEY (board_ID) REFERENCES Board(ID)
);
```

3.11. Card

Relational Model: Card(ID, name, description, issue_date, due_date, status, assigned_ID, list_ID)
Table Definition:

```
CREATE TABLE Card(
    ID INTEGER PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(50) NOT NULL,
    description VARCHAR(100),
    issue_date DATETIME NOT NULL,
```

```
    due_date DATETIME,
    assigned_ID INTEGER,
    list_ID INTEGER NOT NULL,
    status VARCHAR(15),
    FOREIGN KEY (list_ID) REFERENCES List(ID),
    FOREIGN KEY (assigned_ID) REFERENCES Worker(ID)
);
```

3.12. Issue

Relational Model: Issue(ID, name, description, card_ID)

Table Definition:

```
    CREATE TABLE Issue(
        ID INTEGER PRIMARY KEY AUTO_INCREMENT,
        name  VARCHAR(50) DEFAULT 0,
        description VARCHAR(50) NOT NULL,
        card_ID INTEGER NOT NULL,
        FOREIGN KEY (card_ID) REFERENCES Card(ID)
);
```

3.13. Answer

Relational Model: Answer(ID, issue_ID, description, user_ID)

Table Definition:

```
    CREATE TABLE Answer(
        ID INTEGER PRIMARY KEY AUTO_INCREMENT,
        description VARCHAR(50) NOT NULL,
        issue_ID INTEGER NOT NULL,
        user_ID INTEGER NOT NULL,
        FOREIGN KEY (issue_ID) REFERENCES Issue(ID),
        FOREIGN KEY (user_ID) REFERENCES Worker(ID)
);
```

3.14. Poll

Relational Model: Poll(ID, description, date, card_ID)

Table Definition:

```
    CREATE TABLE Poll(
        ID INTEGER PRIMARY KEY AUTO_INCREMENT,
        description VARCHAR(100) NOT NULL,
        date DATETIME NOT NULL,
```

```
        card_ID INTEGER NOT NULL,
        FOREIGN KEY (card_ID) REFERENCES Card(ID)
);
```

3.15. Response

Relational Model: Response(poll_ID, description, no_of_votes)
Table Definition:

```
CREATE TABLE Response(
        description VARCHAR(50) NOT NULL,
        no_of_votes INTEGER DEFAULT 0          ,
        poll_ID INTEGER NOT NULL,
        FOREIGN KEY (poll_ID) REFERENCES Poll(ID)
);
```

3.16. Prereq Cards

Relational Model: PrereqCards(ID, prereq_ID)
Table Definition:

```
CREATE TABLE PrereqCards(
        ID INTEGER NOT NULL,
        prereq_ID INTEGER NOT NULL,
        FOREIGN KEY (ID) REFERENCES Card(ID),
        FOREIGN KEY (prereq_ID) REFERENCES Card(ID)
);
```

3.17. Member

Relational Model: Member(member_ID, role, team_ID)
Table Definition:

```
CREATE TABLE Member(
        member_ID INTEGER NOT NULL,
        role VARCHAR(50),
        team_ID INTEGER NOT NULL,
        FOREIGN KEY (member_ID) REFERENCES StandardUser(ID),
        FOREIGN KEY (team_ID) REFERENCES Team(ID),
        PRIMARY KEY (member_ID, team_ID)
);
```

3.18. WorkerResponse

Relational Model: WorkerResponse(worker_ID, poll_ID, description)
Table Definition:

```
CREATE TABLE WorkerResponse(
        worker_ID INTEGER NOT NULL,
        poll_ID INTEGER NOT NULL,
        description VARCHAR(50),
        FOREIGN KEY (worker_ID) REFERENCES Worker(ID),
        FOREIGN KEY (poll_ID) REFERENCES Poll(ID),
        PRIMARY KEY (worker_ID, poll_ID)
);
```

3.19. WorkOn

Relational Model: WorkOn(team_ID, project_ID, leader_ID,manager_ID, rating, since)
Table Definition:

```
CREATE TABLE WorkOn(
        team_ID INTEGER NOT NULL,
        leader_ID INTEGER,
        project_ID INTEGER NOT NULL,
        manager_ID INTEGER NOT NULL,
        rating NUMERIC(1,0) DEFAULT 1,
        since DATETIME,
        FOREIGN KEY (team_ID) REFERENCES Team(ID),
        FOREIGN KEY (leader_ID) REFERENCES TeamLeader(ID),
        FOREIGN KEY (project_ID) REFERENCES Project(ID),
        FOREIGN KEY (manager_ID) REFERENCES Manager(ID),
        PRIMARY KEY( team_ID, leader_ID, project_ID),
        CHECK( rating < 5 AND 1 < rating )
);
```

4. Implementation Details

4.1. Environment, Framework and Languages

PHP was used as the main backend language to provide the interactions between the database and the website. The main front end language used to process the UI was HTML and CSS. We have also used Bootstrap for the UI of some pages. Since our website required a high UI interaction with the user and based all dynamic content to the user input, thus we used Javascript to dynamically create content based on user input. For the DBMS, we used mySQL (MariaDB). We've also outputted the latest version of the database to be able to rebuild it from scratch.

4.2. Problems and Solutions

4.2.1. Inconsistent Output

One of the major problems faced during the development was caused by the inconsistencies in outputs we received. First of all, the problem emerged as the group members used different browsers for testing the application. This problem was solved by agreeing on a common browser. This browser was chosen to be Mozilla Firefox. However, as we added new features to the application, some features did not work on Mozilla, so we decided to move on to Google Chrome regardless of its RAM usage.

4.2.1. Case-Sensitive Database

Another similar problem we faced was regarding the case sensitivity of MariaDB, whereas mySQL is case in sensitive. As some of the group members preferred to run the queries on their local SQL servers and some preferred to run on dijkstra, problems regarding only the inconsistent case-sensitivity emerged. We solved them by using the same database by all group members. Although this caused data inconsistency sometimes, we managed to overcome.

4.2.3. Lack of Data

Since this application is only at a demo level, there aren't users who can produce data for using in the database and for creating reports. For solving this problem, we manually created some garbage data during development, also for creating reports we've used http://filldb.info for generating plenty of dummy but useable data.

5. Advanced Database Features

5.1. Views

A project is a general entity in Tasks&Managers. For providing the required privileges and obeying the principle of least knowledge, we abstracted out details of Projects for some users. For practicality, we used a view as the following,

```
CREATE VIEW non_manager_project AS
SELECT id, name, description, issue_date, app_domain,due_date
FROM project
```

5.2. Secondary Indices

For efficient and faster searching, we used secondary indices. The attribute we chose to use as a secondary index is due_date. Because it is an advised practice and it also eases the filtering process of cards. We used the following queries for setting the secondary indices.

```
CREATE INDEX project_due_index USING BTREE ON project(due_date);

CREATE INDEX list_due_index USING BTREE ON list(due_date);

CREATE INDEX card_due_index USING BTREE ON card(due_date);
```

5.3. Reports

For getting interesting statistics about the usage of the system, we created the following reports.

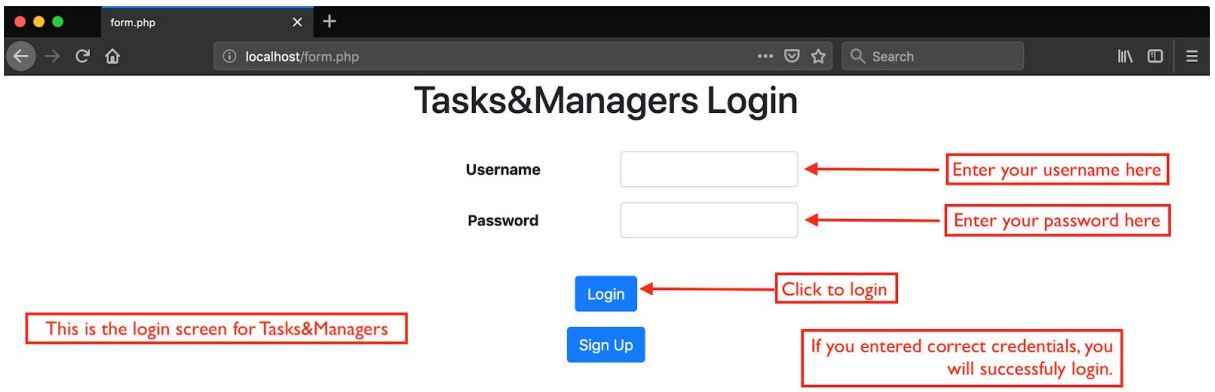
```
SELECT app_domain, sum(budget) as total_budget
FROM project
GROUP BY app_domain
ORDER BY total_budget DESC;
```

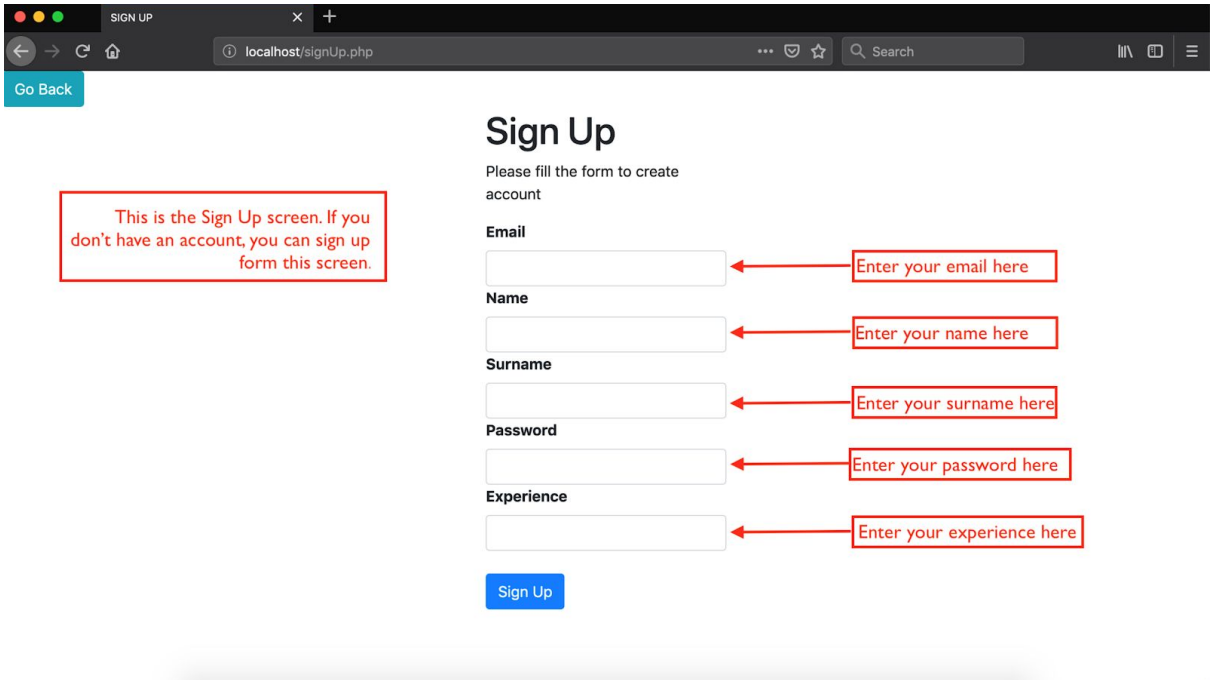
```
MariaDB [cagla_sozen]> SELECT app_domain, sum(budget) as total_budget
    -> FROM project
    -> GROUP BY app_domain
    -> ORDER BY total_budget DESC;
+------------------+--------------+
| app_domain       | total_budget |
+------------------+--------------+
| 123123           |       123123 |
| total domination |       100000 |
| dfsdf            |        23123 |
| graphics         |        14000 |
| 1232asdad        |         2333 |
| develop          |         1500 |
| adventure        |          123 |
| Innovation       |          100 |
| Good domain      |            6 |
| Super Heroes     |            2 |
+------------------+--------------+
```

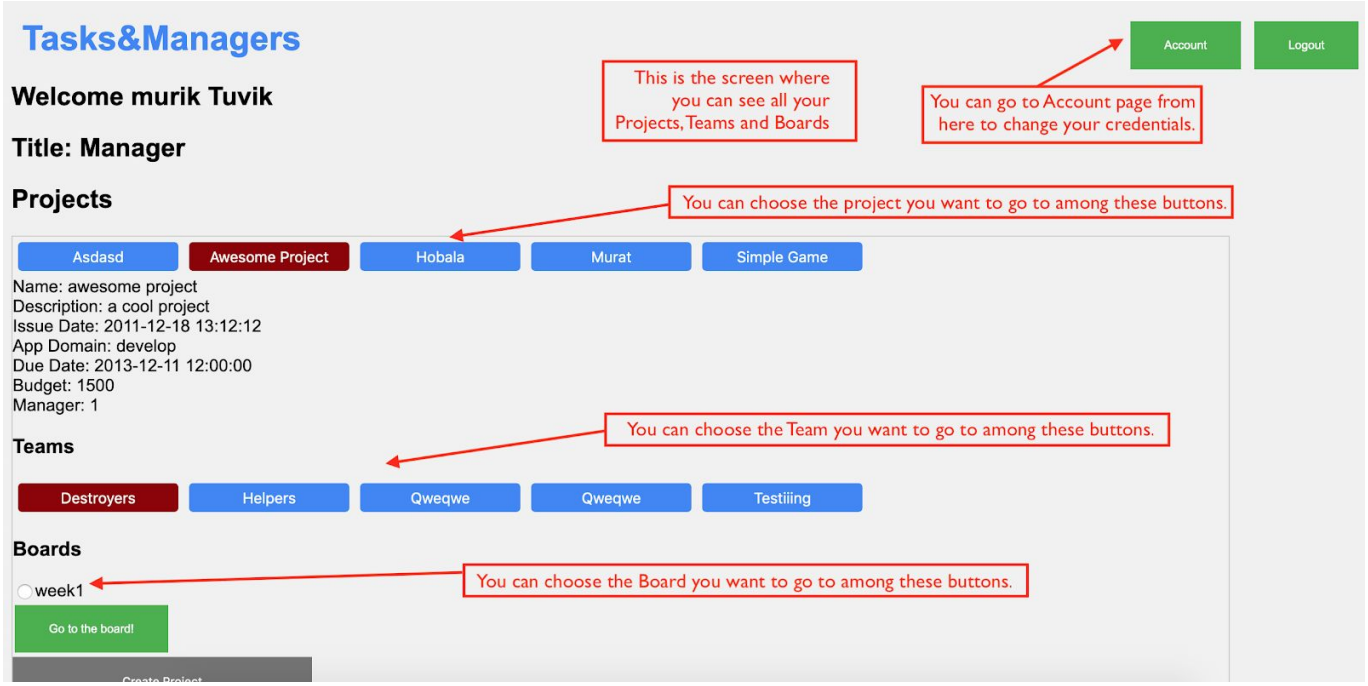
```
CREATE VIEW Popularity AS(
SELECT app_domain, count(ID) as total_count
FROM project
GROUP BY app_domain
ORDER BY total_count DESC
);
```

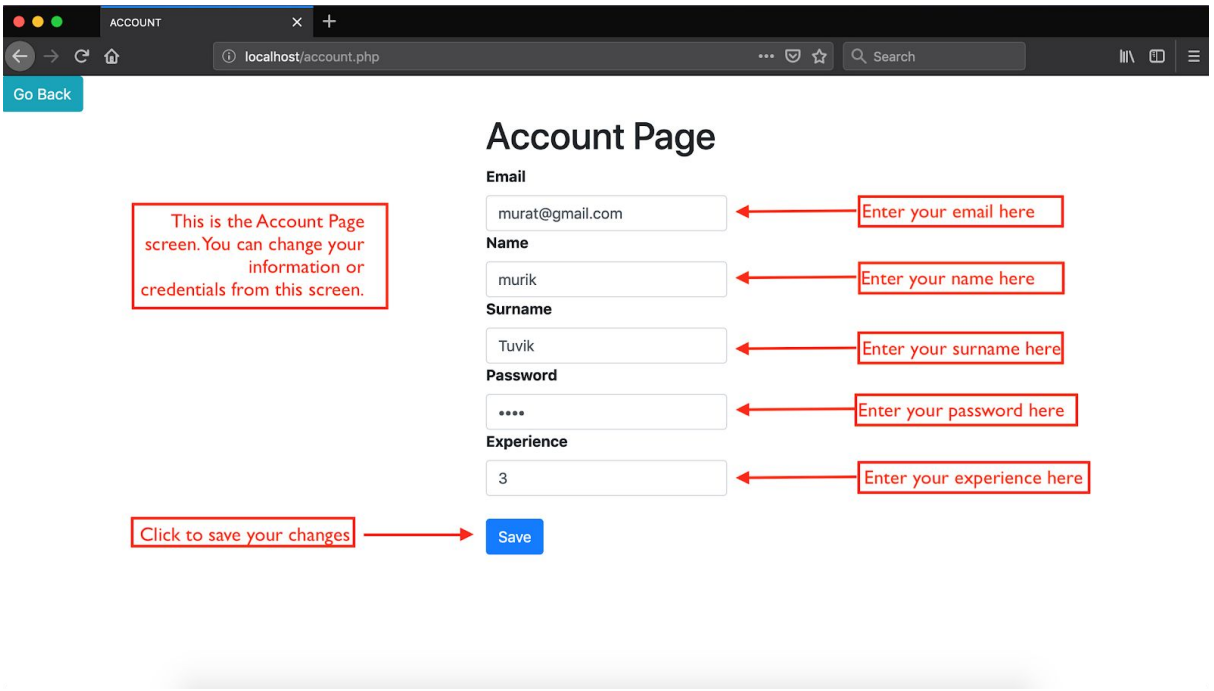
```
MariaDB [cagla_sozen]> CREATE VIEW Popularity AS(
    -> SELECT app_domain, count(ID) as total_count
    -> FROM project
    -> GROUP BY app_domain
    -> ORDER BY total_count DESC
    -> );
Query OK, 0 rows affected (0.02 sec)

MariaDB [cagla_sozen]> select * from Popularity;;
+------------------+-------------+
| app_domain       | total_count |
+------------------+-------------+
| Good domain      |           1 |
| graphics         |           1 |
| Innovation       |           1 |
| 123123           |           1 |
| dfsdf            |           1 |
| Super Heroes     |           1 |
| 1232asdad        |           1 |
| total domination |           1 |
| develop          |           1 |
| adventure        |           1 |
+------------------+-------------+
```

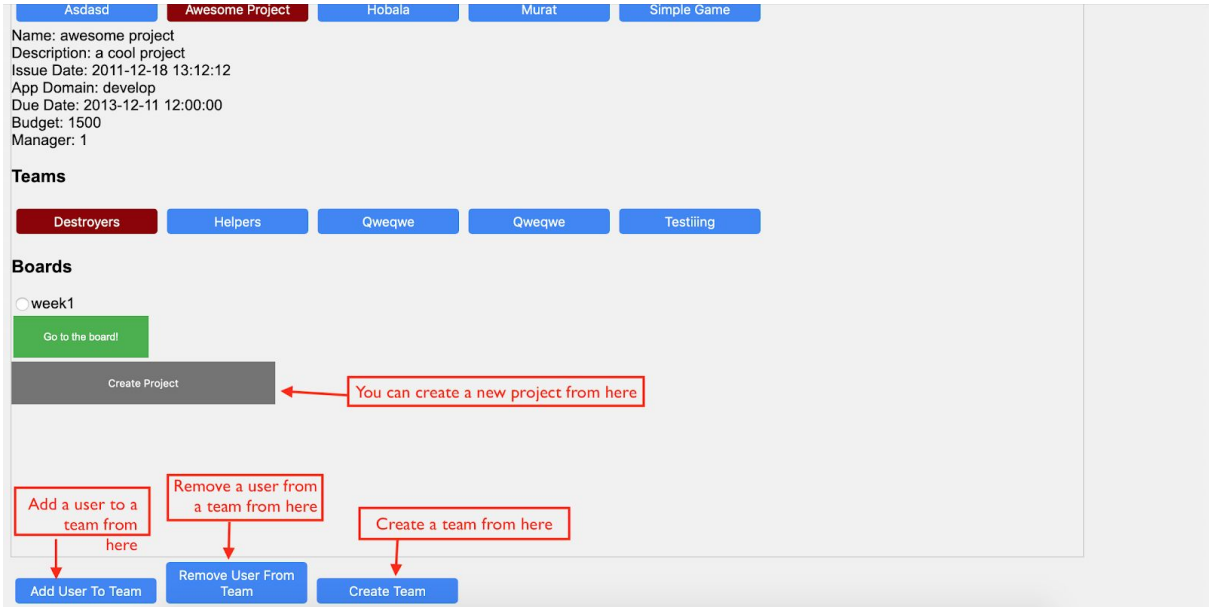
6. User Manual

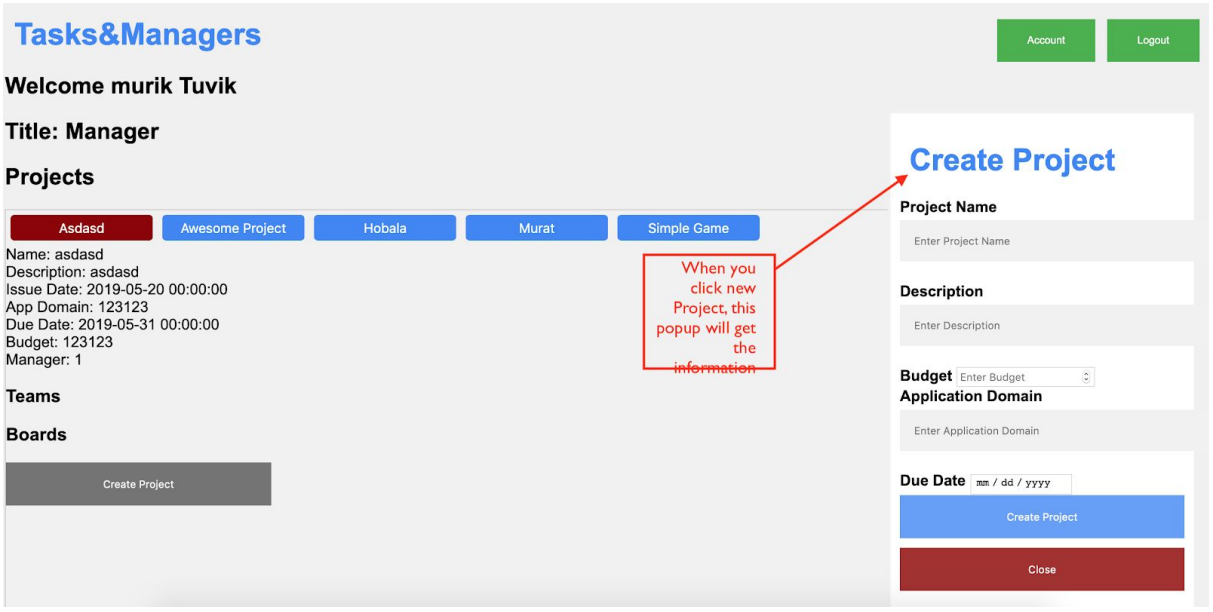


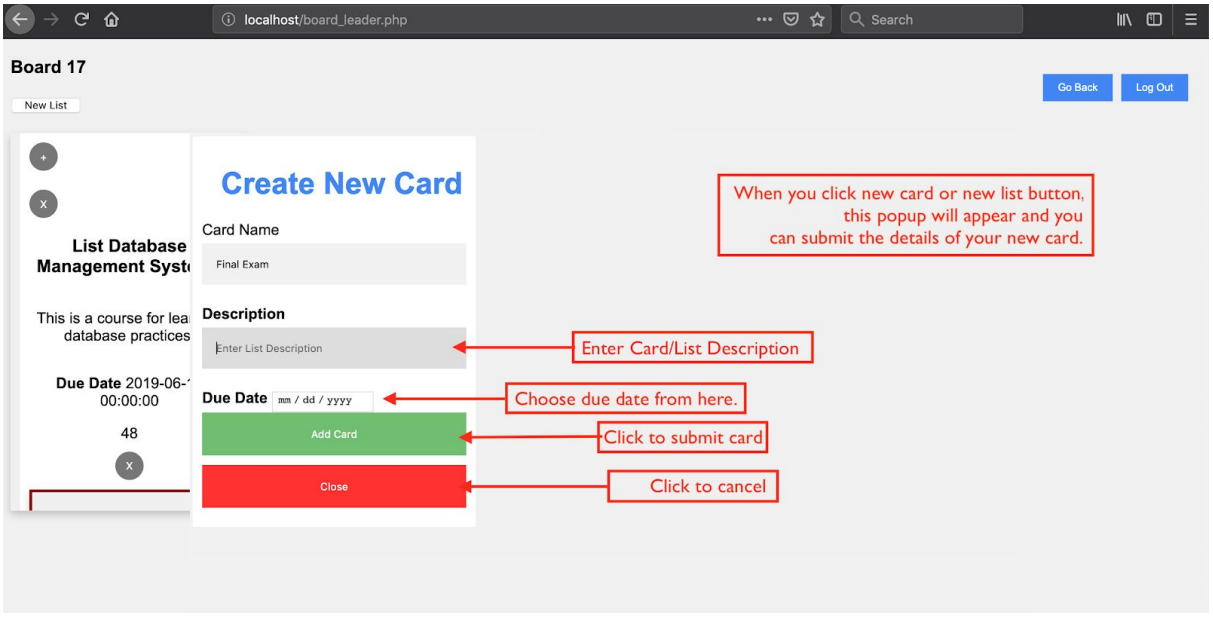


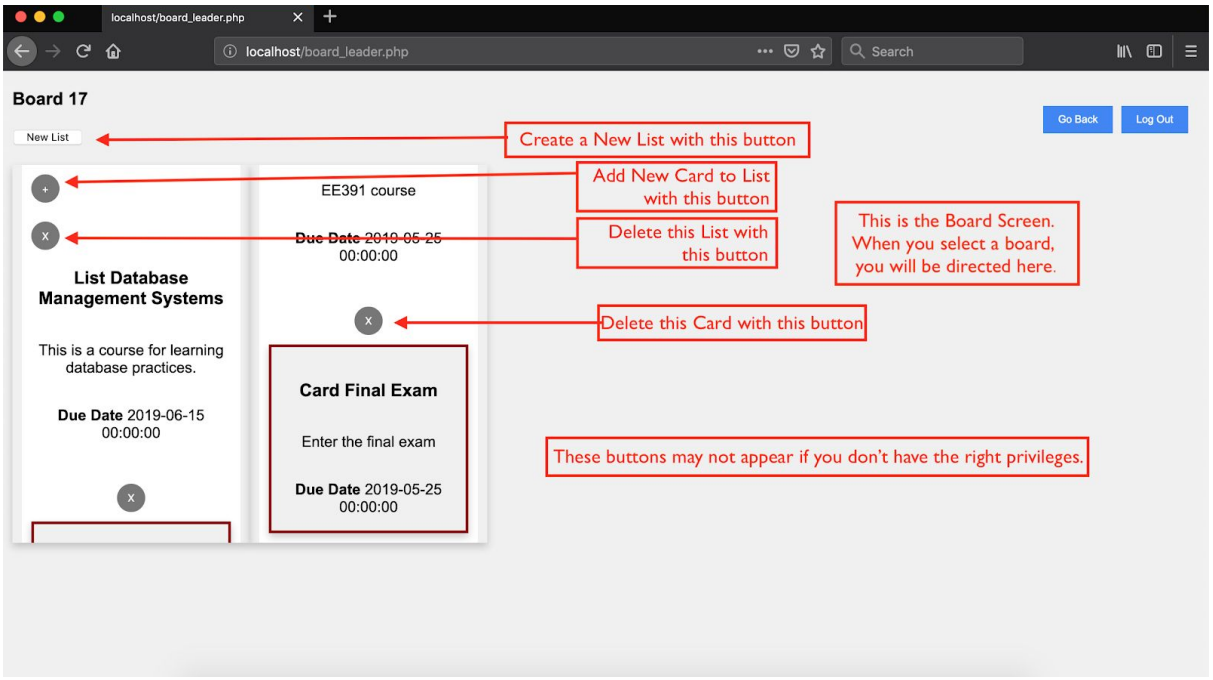

Tasks&Managers
Account
Logout
This is the screen where you can see all your Projects, Teams and Boards
You can go to Account page from here to change your credentials.
Welcome murik Tuvik
Title: Manager
Projects
You can choose the project you want to go to among these buttons.
Asdasd
Awesome Project
Hobala
Murat
Simple Game
Name: awesome project
Description: a cool project
Issue Date: 2011-12-18 13:12:12
App Domain: develop
Due Date: 2013-12-11 12:00:00
Budget: 1500
Manager: 1
Teams
You can choose the Team you want to go to among these buttons.
Destroyers
Helpers
Qweqwe
Qweqwe
Testiiing
Boards
week1
You can choose the Board you want to go to among these buttons.
Go to the board!

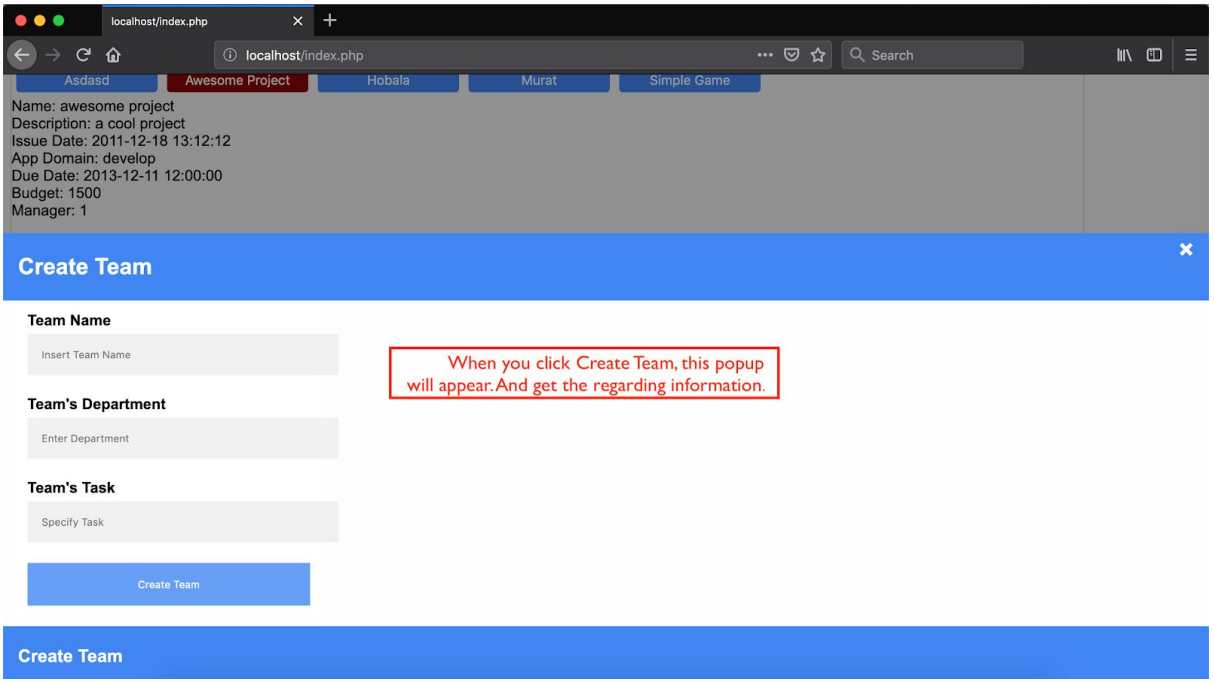

ACCOUNT
localhost/account.php
Search
Go Back
Account Page
This is the Account Page screen. You can change your information or credentials from this screen.
Email
murat@gmail.com
Enter your email here
Name
murik
Enter your name here
Surname
Tuvik
Enter your surname here
Password
••••
Enter your password here
Experience
3
Enter your experience here
Click to save your changes
Save

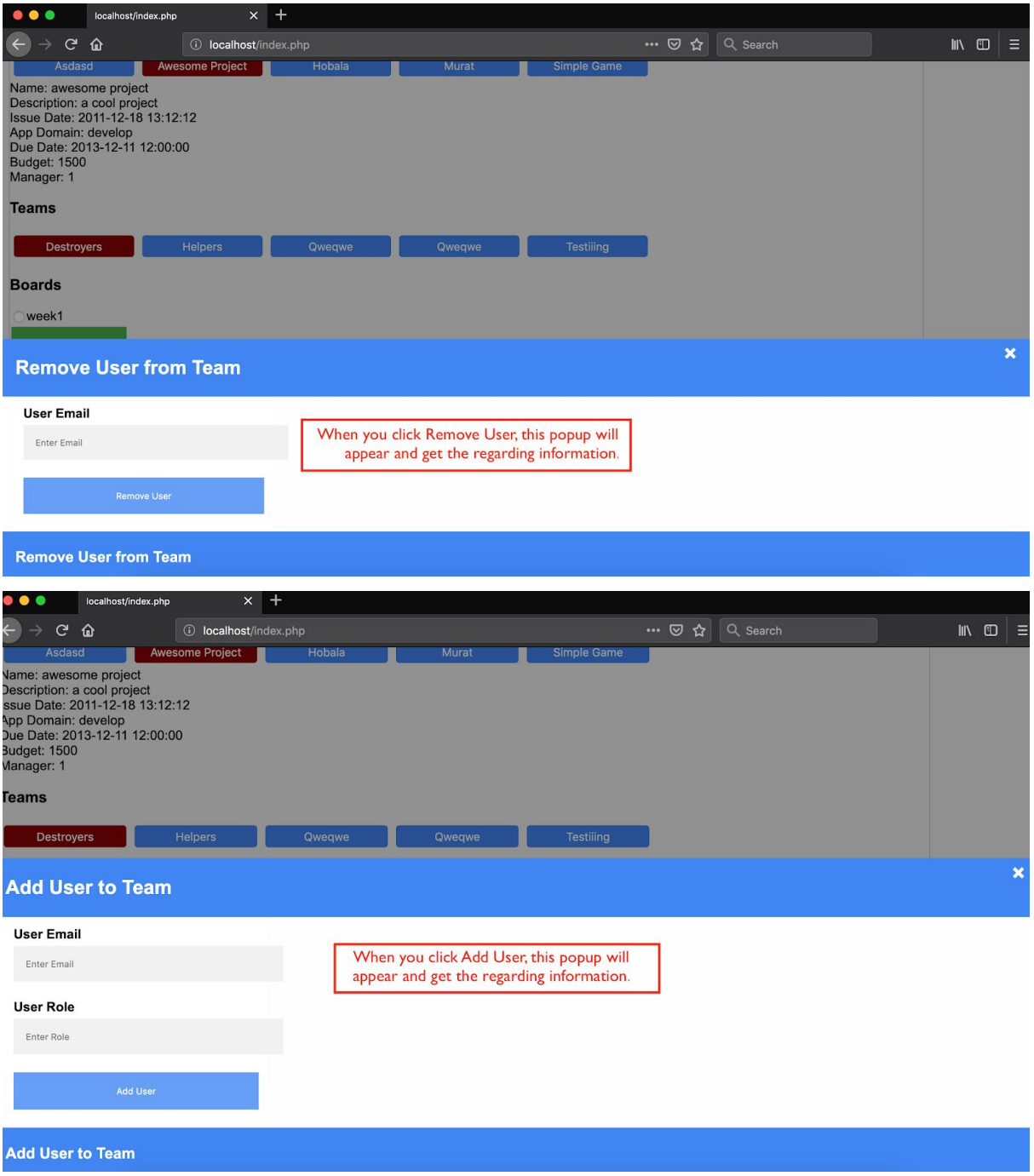

Asdasd
Awesome Project
Hobala
Murat
Simple Game
Name: awesome project
Description: a cool project
Issue Date: 2011-12-18 13:12:12
App Domain: develop
Due Date: 2013-12-11 12:00:00
Budget: 1500
Manager: 1
Teams
Destroyers
Helpers
Qweqwe
Qweqwe
Testiiing
Boards
week1
Go to the board!
Create Project
You can create a new project from here
Add a user to a team from here
Remove a user from a team from here
Create a team from here
Add User To Team
Remove User From Team
Create Team


Tasks&Managers
Account
Logout
Welcome murik Tuvik
Title: Manager
Projects
Asdasd
Awesome Project
Hobala
Murat
Simple Game
Name: asdasd
Description: asdasd
Issue Date: 2019-05-20 00:00:00
App Domain: 123123
Due Date: 2019-05-31 00:00:00
Budget: 123123
Manager: 1
Teams
Boards
Create Project
When you click new Project, this popup will get the information
Create Project
Project Name
Enter Project Name
Description
Enter Description
Budget
Enter Budget
Application Domain
Enter Application Domain
Due Date
mm / dd / yyyy
Create Project
Close


localhost/board_leader.php
Search
Board 17
New List
Go Back
Log Out
Create New Card
Card Name
Final Exam
Description
Enter List Description
Due Date mm / dd / yyyy
Add Card
Close
List Database Management Syst
This is a course for lea database practices
Due Date 2019-06-
00:00:00
48
When you click new card or new list button, this popup will appear and you can submit the details of your new card.
Enter Card/List Description
Choose due date from here.
Click to submit card
Click to cancel


localhost/board_leader.php
Search
Board 17
New List
Go Back
Log Out
Create a New List with this button
Add New Card to List with this button
Delete this List with this button
This is the Board Screen. When you select a board, you will be directed here.
Delete this Card with this button
EE391 course
Due Date 2019-05-25
00:00:00
List Database Management Systems
This is a course for learning database practices.
Due Date 2019-06-15
00:00:00
Card Final Exam
Enter the final exam
Due Date 2019-05-25
00:00:00
These buttons may not appear if you don't have the right privileges.


localhost/index.php
localhost/index.php
Search
Asdasd
Awesome Project
Hobala
Murat
Simple Game
Name: awesome project
Description: a cool project
Issue Date: 2011-12-18 13:12:12
App Domain: develop
Due Date: 2013-12-11 12:00:00
Budget: 1500
Manager: 1
Create Team
Team Name
Insert Team Name
Team's Department
Enter Department
Team's Task
Specify Task
Create Team
When you click Create Team, this popup will appear. And get the regarding information.
Create Team



7. Website

Project reports will be available on:

https://caglasozen.github.io/Tasks-Managers/