Gamification of Data

***Final Report***

**Senior Design Project - CIS 4951 - Fall 2019**

**Version No. 1.0**

Team Members:

Cameron Kozan

Zaid Alsafi

Alex Navarre

**Project Document Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version Number | Date | Revising Author | Description of Revision |
| 1.0 | 4/19/2019 | Zaid Alsafi, Cam Kozan, Alex Navarre | Document Created |

**Acknowledgments**

This document contains all information on the development of the project “Gamification of Big

Data” for Sargon Partners Inc.

|  |
| --- |
| **Table of Contents** |

[**1.0 Introduction**](#_83ik91ktkl1p) **7**

[**1.1 Goals and Objectives**](#_orgkxpxhw7vb) **7**

[**1.2 Statement of Scope**](#_ungjh4hatje0) **7**

[**1.3 Software Context**](#_rdgqjr5hrkau) **7**

[**1.4 Major Constraints**](#_he46pyfvih) **7**

[**2.0 Requirements**](#_6r77i7146eai) **7**

[**2.1 Functional Model and Description**](#_g0drvh5cbgfj) **7**

[**2.1.1 Function Point 1**](#_m9ppk8n80q0q) **7**

[**2.1.2 Function Point 2**](#_5bufzgsokmj4) **8**

[**2.1.3 Function Point 3**](#_cy34oqhm8yf) **8**

[**2.1.5 Function Point 4**](#_4ok9qrxqqv81) **8**

[**2.1.6 Function Point 5**](#_3chhdbkz7gqo) **8**

[**2.1.7 Function Point 6**](#_19d2vzl9ltv2) **9**

[**2.1.8 Function Point 7**](#_4z7t805ebbhv) **9**

[**2.1.9 Function Point 8**](#_ahsw5w9o14nl) **9**

[**2.1.10 Function Point 9**](#_obkupd8zdg9i) **9**

[**2.1.11 Function Point 10**](#_wqae0kljb0rt) **10**

[**3.0 Hardware/Software Design**](#_mftvunn53dp6) **10**

[**3.1 Internal Software Data Structure**](#_603luj9271tf) **10**

[**3.2 Global Data Structure**](#_cb7o4hmg8for) **12**

[**3.3 Temporary Data Structure**](#_o2vi9lrt3imc) **13**

[**3.4 Database Description**](#_fgr9jnoaqbk6) **13**

[**3.5 Hardware Architecture Diagram**](#_s4872fr978ho) **14**

[**4.0 Implementation Details**](#_a4gveaecrrq) **14**

[**4.1 Creating a new shard query**](#_j3lp9q9y4y92) **14**

[**4.2 Shard Manager query**](#_661zx1aa45x5) **22**

[**4.3 Front End Files**](#_1ax6rnujxzg6) **28**

[**4.4 Back End Files**](#_p609e9lya0c6) **34**

[**5.0 Testing/SQA**](#_lt00q2d7ql4r) **38**

[**5.1 Scope and intent of SQA activities**](#_qj8qi8q97e9j) **38**

[**5.2 SQA organizational role**](#_nx96034zbpcp) **38**

[**5.3 SQA Tasks**](#_ga2s9lv0alkz) **38**

[**5.3.1 Task Overview**](#_xs8lmbmpwck6) **38**

[**5.3.1.1 Description of SQA Task 1**](#_7gakxcpzsvkh) **38**

[**5.3.1.1.1 Work Products and Documentation for Task 1**](#_2wcamxdi2sch) **38**

[**5.3.2 Description of SQA Task 2**](#_g1eddvlnlbow) **38**

[**5.3.2.1 Work Products and Documentation for Task 2**](#_ejwrfi109rxq) **39**

[**5.3.3 Description of SQA Task 3**](#_mnr9jxt7fkxp) **39**

[**5.3.3.1 Work Products and Documentation for Task 3**](#_c1cbi9rod3br) **39**

[**5.4 Standards, Practices and Conventions (SPC)**](#_9qvnevxeofig) **39**

[**5.5 SQA Resources**](#_s563n8cpzmew) **39**

[**5.6 Reviews and Audits**](#_doj7355pi2av) **40**

[**5.6.1 Generic Review Guidelines**](#_s0aixd8g1cv5) **40**

[**5.6.1.1 Conductions a Review**](#_frino6xoc397) **40**

[**5.6.1.2 Roles and Responsibilities**](#_langnz1a4h5k) **40**

[**5.6.1.3 Review work products**](#_6ra529il3yc0) **41**

[**5.6.2 Formal Technical Reviews**](#_ng0dgmzbgy0d) **41**

[**5.6.2.1 Description of SPMP review**](#_9ahxudvyzt34) **41**

[**5.6.2.1.1 Description and Focus of the review**](#_qaifc022o8c1) **41**

[**5.6.2.1.2 Timing of the review**](#_ftvmk9tvh5vm) **41**

[**5.6.2.1.3 Work products produced**](#_146qxin8rud5) **41**

[**5.6.2.1.4 Review checklist**](#_2eb6sevr9h0h) **41**

[**5.6.3 Description of RMMM review**](#_g18ilaapq9md) **42**

[**5.6.3.1 Description and Focus of the review**](#_c0m00qsvc0gm) **42**

[**5.6.3.2 Timing of the review**](#_duzx937xhhcl) **42**

[**5.6.3.3 Work products produced**](#_v53n0osiag0b) **42**

[**5.6.3.4 Review checklist**](#_f142b7haqhc6) **42**

[**5.6.4 Description of Requirements review**](#_fhsbch5a9ubn) **43**

[**5.6.4.1 Description and Focus of the review**](#_ll5grk9koaa) **43**

[**5.6.4.2 Timing of the review**](#_vghngcsiznth) **43**

[**5.6.4.3 Work products produced**](#_28pya52y069f) **43**

[**5.6.4.4 Review checklist**](#_hg2362oxyyeo) **43**

[**5.6.5 Description of Architectural design review**](#_6jbcdijaux63) **44**

[**5.6.5.1 Description and Focus of the review**](#_e1hgdvkbbp63) **44**

[**5.6.5.2 Timing of the review**](#_qobc91u7wnha) **44**

[**5.6.5.3 Work products produced**](#_c7e5ks1qukt1) **44**

[**5.6.5.4 Review checklist**](#_fcnjyfnfu9p7) **45**

[**5.6.6 Description of Interface design review**](#_cug2wm3vabft) **45**

[**5.6.6.1 Description and Focus of the review**](#_nzigpleb1fgh) **45**

[**5.6.6.2 Timing of the review**](#_wzauokhbwbvf) **45**

[**5.6.6.3 Work products produced**](#_incwcp7w2wc) **45**

[**5.6.6.4 Review checklist**](#_gecxs35twxsq) **45**

[**5.6.7 Description of Code review**](#_w1o4zei0bpej) **46**

[**5.6.7.1 Description and Focus of the review**](#_g0ins6f1cijh) **46**

[**5.6.7.2 Timing of the review**](#_kanfcv47021d) **46**

[**5.6.7.3 Work products produced**](#_ereixkt4kyy0) **46**

[**5.6.7.4 Review checklist**](#_m2b07qn9teqd) **46**

[**5.6.8 Description of Test specification review**](#_p4wqgz28ui1e) **46**

[**5.6.8.1 Description and Focus of the review**](#_p5i6ea71b6su) **46**

[**5.6.8.2 Timing of the review**](#_uc5orz2vg0uf) **47**

[**5.6.8.3 Work products produced**](#_oy6qof19des5) **47**

[**5.6.8.4 Review checklist**](#_2dq3q1xt1c2o) **47**

[**5.6.9 SQA audits**](#_h6qccd6ppemp) **47**

[**5.7 Problem Reporting and Corrective Action**](#_3uayrkxx35rw) **47**

[**5.7.1 Reporting Mechanisms**](#_jxwrgmeo9kwt) **47**

[**5.7.2 Responsibilities**](#_f6yhp5j5ki8v) **48**

[**5.7.3 Data collection and evaluation**](#_nm82ojbc0g2j) **48**

[**5.7.4 Statistical SQA**](#_4lo56he9rmst) **48**

[**5.8 Software Process Improvement Activities**](#_azyxqjtbl0dw) **48**

[**5.8.1 Goal and Objectives of SPI**](#_ju1xs7ij6fu6) **48**

[**5.8.2 SPI tasks and responsibilities**](#_cvgsd46eigug) **48**

[**5.9 Software Configuration Management Overview**](#_ja4nvhb0guir) **49**

[**5.9.1 Overview**](#_u0ou7t9633pc) **49**

[**5.9.2 Approach**](#_xxlqcsm5m5d) **49**

[**5.9.3 Impact**](#_7nifktkifar2) **49**

[**5.10 SQA Tools, Techniques, Methods**](#_4di8q3ibc3j8) **50**

[5.11 Test Cases](#_pk4f3xc0isco) 50

[**6.0 Future Maintenance Suggestions**](#_g1tisua7yo8o) **53**

[**6.1 Updating the Service**](#_rodaj7yccohn) **53**

[6.2 Updating Base Challenges](#_8hhcy6xa7rvv) 53

[6.3 Updating Base Badges](#_d150n0dvldvw) 53

[**7.0 References & Bibliography**](#_dhgezg977xeb) **53**

[**Appendix A - User Manual**](#_souo7ivgs5ly) **53**

[**Getting Started**](#_wiuifa30yvw6) **53**

[**Creating a new shard**](#_iv02xgp22cay) **53**

[**Installing dependencies & running the front end**](#_9g26f8t8kvy2) **54**

[**Running the back end online**](#_bwvd1y9lwi8x) **54**

[**EC2 Information**](#_bgey7ncxowv2) **55**

[**New Shard Configuration**](#_gp4fky6aofwj) **55**

[**Finishing Notes**](#_ow56jrrb1kho) **55**

[**Appendix B - Program Listing**](#_hci0ty5ijl4b) **56**

[**Appendix C - Team Member Resumes**](#_bnmbwij8qry9) **62**

[**Appendix D - Project Plan & Log Book**](#_6vxlapwwpkhk) **65**

[**Appendix E - Project Demo Notes**](#_63k6u47dj6v) **68**

[**Appendix F - Final Presentation Slides**](#_mqusal76egr) **68**

# 1.0 Introduction

## 1.1 Goals and Objectives

The goal of this project is to develop a gamified web application. The gamified web application will allow for the workers to be more motivated along with increasing productivity. The end goal of this project is for a user to go into the web application see how they rank among their peers, look at achievements, along with how all their teams are doing as well.

## 1.2 Statement of Scope

The Gamification software will use the client’s API, Invisitag, which will allow us to see all of the teams’ assets. From their api, each asset contains a cost and a weight, which we have incorporated in our website. Our website will allow admins to edit the cost and the weight of each asset. The employees will each have a score per every time they check in their assets. This score is composed of the asset weight multiplied by if it’s stolen, lost or it’s their own asset. From there, the user will see a dashboard with all of the statistics, such as how they rank among their teammates, their score over time graph, along with many more metrics. The admins will assign employees to their respective teams, which then the employee’s performance will be based on the team performance.

## 1.3 Software Context

This software is going to be used by employees and managers of a company. Employees will be grouped into teams, where the employee’s score will be an aggregate score of their teams. The managers and employees will be able to visually see their progress and how they can improve their performance.

## 1.4 Major Constraints

* Web application front end, back end and service must all run on EC2
* Database must be hosted on AWS RDS

# 2.0 Requirements

## 2.1 Functional Model and Description

### 2.1.1 Function Point 1

**Description**: User attempts to logs in

**Use Case Name**: Log In

**Actors**: Employee, Admin

**Preconditions**: N/A

**Triggers**: Navigate to URL

**Scenario Description**: A user will be prompted to log into their account any time they try to access the website. Each account will be associated with either an Employee or an Admin account that determines their privilege levels inside of the application.

**Post Conditions**: If the user successfully enters their credentials, they will be sent to their home dashboard and be given access to the application. If the user fails to enter in matching credentials, an error message will be displayed and they will not be granted access to the application.

**Exceptions**: N/A

### 2.1.2 Function Point 2

**Description**: User forgets their password/Password recovery

**Use Case Name**: Forgotten Password

**Actors**: Employee, Admin

**Preconditions**: User must already have an account set up

**Triggers**: User clicks the ‘Forgot Password’ button

**Scenario Description**: If a user forgets their password, they can navigate to the web application URL and the login page will give them an option to reset their password. The user must click on the ‘Forgot Password’ button on the login screen and then the application will generate a new password and send it to the user via the registered email.

**Post Conditions**: If the user enters a valid email address, the application will randomly generate a new password for the user and send it to the registered email account.

**Exceptions**: N/A

### 2.1.3 Function Point 3

**Description**: Log Out

**Use Case Name**: Log Out

**Actors**: Employee, Admin

**Preconditions**:User must already have an account set up and is logged in to the website.

**Triggers**: User clicks on the “Log Out” button

**Scenario Description**: When the user logs into their account and has completed the tasks that user logged in for they are able to click on the “Log Out” button on the top bar of the webpage.

**Post Conditions**: User Logs out of their account

**Exceptions**: N/A

### 2.1.5 Function Point 4

**Description**: View Profile Page

**Use Case Name**: View Profile

**Actors**: Employee, Admin

**Preconditions**: Successfully login to an account

**Triggers**: When the user Logs into their account and clicks on the view profile on the side bar the system will retrieve all the data that the user has involving them.

**Scenario Description**: When the user logs into their account and clicks on the view profile the system will grab all the information that the account has if it has no information the website will inform the user it has no data.

**Post Conditions**:Viewing the accounts data that the system has.

**Exceptions**: N/A

### 2.1.6 Function Point 5

**Description**: User navigates to the ‘Main’ page of website

**Use Case Name**: Display Dashboard

**Actors**: Employee, Admin

**Preconditions**: Successful login to account

**Triggers**: Successful login to account, Navigate to ‘Main’ page

**Scenario Description**: The user will be taken to the main dashboard page. This can happen in two different instances. The first instance is when the user first logs in to their account, the application will default to the ‘Main’ dashboard page. The second instance is whenever the user navigates to the ‘Main’ page using the navigation bar.

**Post Conditions**: Display the proper dashboard based on account level

**Exceptions**: N/A

### 2.1.7 Function Point 6

**Description**: User navigates to the ‘Manage Challenge’ page

**Use Case Name**: Display Manage Challenges Page

**Actors**: Admin

**Preconditions**: Successful admin level account login

**Triggers**: Logging into the application with an admin level account

**Scenario Description**: A ‘Manage Challenge’ option will be displayed on the navigation bar if a user logs in to the application with an admin level account. The page allows the user to create, edit, and delete specific challenges. It also lets the user add or edit the rewards as well.

**Post Conditions**: The ‘Manage Challenge’ page will be displayed to the admin level account. An employee level account will see the ‘Manage Challenge’ option on their navigation bar.

**Exceptions**: N/A

### 2.1.8 Function Point 7

**Description**: Create Challenge Page

**Use Case Name**: Create Challenge

**Actors**: Admin

**Preconditions**:User Login and has the right of a Admin

**Triggers**: The User clicks on “Manage Challenges” from the side navigation bar.

**Scenario Description**:The User wants to create and manage the challenges that they have created. If they want to update any of their information they are allowed to change it when clicking on the challenge they want to update.

**Post Conditions**: A challenge has been updated/Created

**Exceptions**: N/A

### 2.1.9 Function Point 8

**Description**: Completing/assigning rewards for challenges

**Use Case Name**: Reward Challenge

**Actors**: Admin

**Preconditions**: Admin level account navigates to the ‘Manage Challenge’ page.

**Triggers**: Having at least one available challenge

**Scenario Description**: The admin level account can manually complete and assign rewards for users that have completed challenges. They have the ability to set a clear condition for user defined challenges and can alter/distribute the rewards associated with each challenge.

**Post Conditions**: The associated account will be awarded the chosen badge and experience points from their completed challenge.

**Exceptions**: N/A

### 2.1.10 Function Point 9

**Description**: User navigates to ‘Manage Team’ page

**Use Case Name**: Manage Team

**Actors**: Admin

**Preconditions**: Admin level account login.

**Triggers**: Admin level account navigates to the ‘Manage Challenge’ page.

**Scenario Description**: An admin level account can edit existing teams, add new teams, or delete teams.

**Post Conditions**: All changes will be made for all users and the database will be updated.

**Exceptions**: N/A

### 2.1.11 Function Point 10

**Description**: Manage Asset

**Use Case Name**: Manage Assets

**Actors**: Admin

**Preconditions**: Admin level account

**Triggers**: The user clicks on the manage assets button on the side bar.

**Scenario Description**: The user wants to edit a tool and change its value. By clicking on the page the user is able to edit and delete an asset including adding assets that are not in the tool list.

**Post Conditions**:An asset has been updated, deleted, or added.

**Exceptions**: N/A

**2.1.12 Function Point 11**

**Description**: User navigates to ‘Manage Employee’

**Use Case Name**: Manage Employee

**Actors**: Admin

**Preconditions**: Admin level account login.

**Triggers**: Admin level account navigates to the ‘Manage Employee’ page.

**Scenario Description**: An admin level account can edit existing employees, add new employee accounts, and deactivate employees. The user can edit the fields: First Name, Last Name, Email, Phone Number, Start Date, and Active.

**Post Conditions**: All changes will be saved and updated to the user.

**Exceptions**: N/A

**2.1.13 Function Point 12**

**Description**: User views notification drawer

**Use Case Name**: Display Notifications

**Actors**: Employee, Admin

**Preconditions**: Successful login to application

**Triggers**: Successful login to application

**Scenario Description**: The user has the five most recent notifications displayed to them when they click the bell icon on the top of the webpage. Notifications will also be displayed on the dashboard.

**Post Conditions**: Notification drawer will be opened and the notifications will be displayed to the user.

**Exceptions**: N/A

# 3.0 Hardware/Software Design

## 3.1 Internal Software Data Structure

* Data structures that are passed among components of the software are described.
  + **User**
  + The user table consists of the following attributes.
  + There will be two levels of security – User (Employee) and Admin (Owner/Manager).
    - **Id** (Primary Key) – The Id representing the user.
    - **CompanyId** (Foreign Key) – The Id representing the company the user works for.
    - **FirstName** – First name of the user.
    - **LastName** – Last name of the user.
    - **Email** – Work email for the user. This will be used as the username at login.
    - **Password** – Password for the user to login. This will be encrypted.
    - **Level** - Represents the user’s level
    - **Role** - Represents the user’s access role
    - **EXP** - How much EXP the user has earned, total
    - **Rank** - The current users’ ranking in the company
    - **StartDate** - Date that the user started working at the company
  + **Team**
  + The team table consists of the following attributes.
    - **Id** (Primary Id) – The Id representing the Team working on a task.
    - **CompanyId** (Foreign Key) – The Id representing the company the team works for.
    - **Name** – The name of the team.
  + **UserBadge**
  + The team table consists of the following attributes.
    - **Id** (Primary Id) – The Id representing the Team working on a task.
    - **UserId**(Foreign Key) – The Id representing the user who is being assigned a badge
    - **BadgeId**(Foreign Key) – The Id representing the badge that is being associated with a user
  + **Rank**
  + The ranking table consists of the following attributes.
    - **Id** (Primary Key) – The Id representing the rank of the team.
    - **AssetId** (Foreign Key) – The Id representing the asset that a team has.
    - **TeamId** (Foreign Key) – The Id representing the team the rank belongs to.
    - **Week** - The week that the asset was scanned
    - **Detected** - A boolean if the asset has been detected or not
    - **ExtraTag** - A boolean if the asset has been stolen or not
  + **Assets**
  + The assets table consists of the following attributes.
    - **Id** (Primary Key) – The Id representing the asset.
    - **Name** – The name of the asset.
    - **Weight** – The weight of the asset. This is what is used when calculating the actual score for a teams rank.
    - **Cost** – The cost of the asset. If no weight is specified by the admin, the cost by default will be used when calculating the actual score for a teams rank.
    - **RFID** - The RFID tag associated with the asset
  + **BadgeIcon**
  + The badge table consists of the following attributes.
    - **Id** (Primary Key) – The Id representing the badge.
    - **BadgeURL** - The URL of the icon for a badge
  + **Badges**
  + The badge table consists of the following attributes.
    - **Id** (Primary Key) – The Id representing the badge.
    - **Name** - The name of the badge
    - **Description** - The description of a badge
    - **BadgeIcon** - The icon of the badge
  + **Challenges**
  + The challenges table consists of the following attributes
    - **Id** (Primary Key) – The Id representing the challenge.
    - **Name** - The name of the challenge
    - **Description** - Description of the challenge and how to get it.
    - **AchievedTime** - Time the challenge was completed
    - **Reward** - Reward for completing the challenge
  + **Newsfeed**
  + The newsfeed table consists of the following attributes
    - **Id** (Primary Key) – The Id representing the news feed activity.
    - **EmployeeId** - The EmployeeId who completed the activity.
    - **Description** - Description of the activity and how to get it.
    - **AchievedTime** - Time the activity was completed
    - **Reward** - Reward for completing the challenge

## 3.2 Global Data Structure

* Data structures that are available to major portions of the architecture are described.
  + **Shard Manager**
  + The shard manager will be the only global data structure used.
    - **Company**
    - The company table consists of the following attributes.
      * **Id** (Primary Key) – The Id representing the company that uses the Invisi-tag software.
      * **ShardId** (Foreign Key) – The shard name that belongs to the company the user works for.
      * **Name** – Name of the company that uses the Invisi-tag software.
    - **Shard**
    - The shard consists of the following attributes.
      * **Id** (Primary Key) – The Id representing the shard for a specific company.
      * **Name** – The name of the shard.
  + **UserLogin**
  + The UserLogin table consists of the following attributes.
  + This table consists of the same attributes as User table, however this table will consist of all users for all companies.
    - **Id** (Primary Key) – The Id representing the user.
    - **CompanyId** (Foreign Key) – The Id representing the company the user works for.
    - **FirstName** – First name of the user.
    - **LastName** – Last name of the user.
    - **Email** – Work email for the user. This will be used as the username at login.
    - **Password** – Password for the user to login. This will be encrypted.
    - **Role** - Represents the user’s access role
    - **EXP** - How much EXP the user has earned, total
    - **Rank** - The current users’ ranking in the company
    - **StartDate** - Date that the user started working at the company

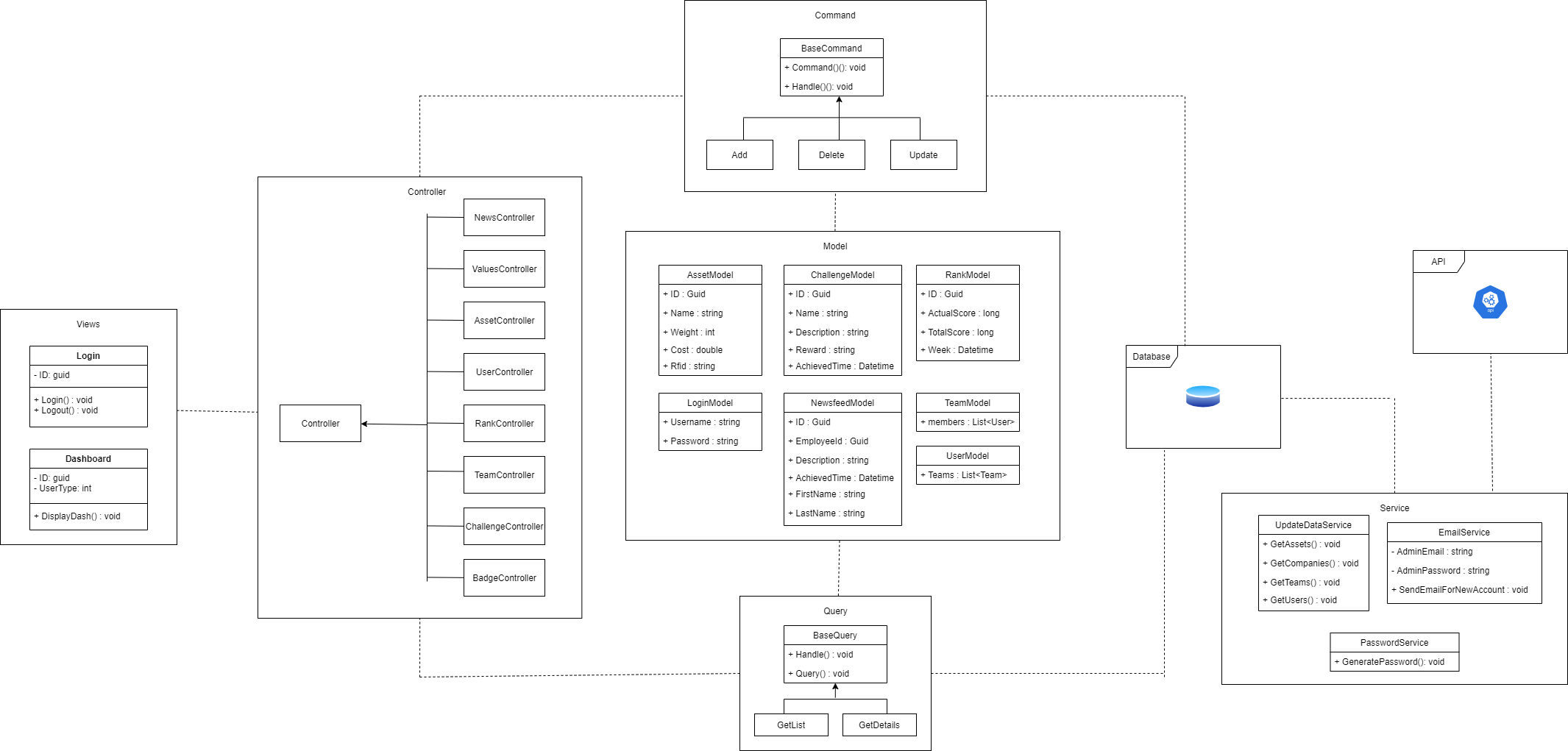
## 3.3 Temporary Data Structure

The only temporary data structures that are used are for session management. Sessions are managed with JWT tokens. They are created once someone logs in and deactivated once the back end shuts off.

## 3.4 Database Description

We have a few different databases. All of our databases are hosted in Amazon’s RDS. There is one main database which manages all of the others called ShardManager. ShardManager contains all of the shards, which are individual databases. These individual databases contain information about one specific company.

## 3.5 Hardware Architecture Diagram



# 4.0 Implementation Details

## 4.1 Creating a new shard query

CREATE TABLE [Asset](

[Id] [uniqueidentifier] NOT NULL,

[Name] [varchar](100) NOT NULL,

[Weight] [int] NOT NULL,

[Cost] [decimal](18, 2) NOT NULL,

[Rfid] [varchar](30) NOT NULL,

PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY],

UNIQUE NONCLUSTERED

(

[Rfid] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

/\*\*\*\*\*\* Object: Table [BadgeImages] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [BadgeImages](

[id] [uniqueidentifier] NOT NULL,

[badgeURL] [varchar](2048) NOT NULL,

CONSTRAINT [PK\_BadgeImages] PRIMARY KEY CLUSTERED

(

[id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0910', 'https://image.flaticon.com/icons/svg/1426/1426739.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0911', 'https://image.flaticon.com/icons/svg/1426/1426719.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0912', 'https://image.flaticon.com/icons/svg/1426/1426725.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0913', 'https://image.flaticon.com/icons/svg/1426/1426722.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0914', 'https://image.flaticon.com/icons/svg/1426/1426724.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0915', 'https://image.flaticon.com/icons/svg/1426/1426720.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0916', 'https://image.flaticon.com/icons/svg/1426/1426735.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0917', 'https://image.flaticon.com/icons/svg/1426/1426729.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0918', 'https://image.flaticon.com/icons/svg/1426/1426716.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d0919', 'https://image.flaticon.com/icons/svg/1426/1426717.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d091a', 'https://image.flaticon.com/icons/svg/1426/1426715.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'https://image.flaticon.com/icons/svg/1426/1426718.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d091c', 'https://image.flaticon.com/icons/svg/1426/1426714.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d091d','https://image.flaticon.com/icons/svg/1426/1426741.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d091e', 'https://image.flaticon.com/icons/svg/1426/1426726.svg');

INSERT INTO [BadgeImages] VALUES('886b5a1d-f64c-4c58-bfc4-998e5e9d091f', 'https://image.flaticon.com/icons/svg/1426/1426734.svg');

/\*\*\*\*\*\* Object: Table [Badges] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [Badges](

[ID] [uniqueidentifier] NOT NULL,

[BadgeIcon] [uniqueidentifier] NULL,

[Name] [varchar](255) NULL,

[Description] [varchar](255) NULL,

CONSTRAINT [PK\_Badges] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

INSERT INTO [Badges] VALUES('916b5a1d-f64c-4c58-bfc4-998e5e9d091b', '886b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'Time Worked - 6 Months', 'Work for a company for 6 months');

INSERT INTO [Badges] VALUES('926b5a1d-f64c-4c58-bfc4-998e5e9d091b', '886b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'Time Worked - 12 Months', 'Work for a company for 12 months');

INSERT INTO [Badges] VALUES('936b5a1d-f64c-4c58-bfc4-998e5e9d091b', '886b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'Time Worked - 18 Months', 'Work for a company for 18 months');

INSERT INTO [Badges] VALUES('946b5a1d-f64c-4c58-bfc4-998e5e9d091b', '886b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'Time Worked - 24 Months', 'Work for a company for 24 months');

INSERT INTO [Badges] VALUES('956b5a1d-f64c-4c58-bfc4-998e5e9d091b', '886b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'No lost tools - 7 days', 'Has not lost a tool in 7 days');

INSERT INTO [Badges] VALUES('966b5a1d-f64c-4c58-bfc4-998e5e9d091b', '886b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'No lost tools - 14 days', 'Has not lost a tool in 14 days');

INSERT INTO [Badges] VALUES('976b5a1d-f64c-4c58-bfc4-998e5e9d091b', '886b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'No lost tools - 30 days', 'Has not lost a tool in 30 days');

/\*\*\*\*\*\* Object: Table [BaseChallenge] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [BaseChallenge](

[id] [uniqueidentifier] NOT NULL,

[name] [nchar](255) NOT NULL,

[description] [nchar](255) NOT NULL,

[reward] [nchar](255) NOT NULL,

[userCreated] [bit] NOT NULL,

[difficulty] [int] NOT NULL,

[badge] [uniqueidentifier] NOT NULL

) ON [PRIMARY]

;

INSERT INTO [BaseChallenge] VALUES('016b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'Time Worked - 6 Months', 'Work for a company for 6 months', 'N/A', 0, 0, '916b5a1d-f64c-4c58-bfc4-998e5e9d091b');

INSERT INTO [BaseChallenge] VALUES('026b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'Time Worked - 12 Months', 'Work for a company for 12 months', 'N/A', 0, 0, '926b5a1d-f64c-4c58-bfc4-998e5e9d091b');

INSERT INTO [BaseChallenge] VALUES('036b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'Time Worked - 18 Months', 'Work for a company for 18 months', 'N/A', 0, 0, '936b5a1d-f64c-4c58-bfc4-998e5e9d091b');

INSERT INTO [BaseChallenge] VALUES('046b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'Time Worked - 24 Months', 'Work for a company for 24 months', 'N/A', 0, 0, '946b5a1d-f64c-4c58-bfc4-998e5e9d091b');

INSERT INTO [BaseChallenge] VALUES('056b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'No lost tools - 7 days', 'Has not lost a tool in 7 days', 'N/A', 0, 0, '956b5a1d-f64c-4c58-bfc4-998e5e9d091b');

INSERT INTO [BaseChallenge] VALUES('066b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'No lost tools - 14 days', 'Has not lost a tool in 14 days', 'N/A', 0, 0, '966b5a1d-f64c-4c58-bfc4-998e5e9d091b');

INSERT INTO [BaseChallenge] VALUES('076b5a1d-f64c-4c58-bfc4-998e5e9d091b', 'No lost tools - 30 days', 'Has not lost a tool in 30 days', 'N/A', 0, 0, '976b5a1d-f64c-4c58-bfc4-998e5e9d091b');

/\*\*\*\*\*\* Object: Table [Challenges] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [Challenges](

[ID] [uniqueidentifier] NOT NULL,

[Name] [nvarchar](255) NOT NULL,

[Description] [nvarchar](255) NOT NULL,

[AchievedTime] [datetime] NULL,

[Reward] [nvarchar](255) NULL,

[Badge] [uniqueidentifier] NULL,

[User] [uniqueidentifier] NULL,

[userCreated] [bit] NOT NULL,

[difficulty] [smallint] NOT NULL,

CONSTRAINT [PK\_Challenges] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

/\*\*\*\*\*\* Object: Table [MemberOf] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [MemberOf](

[TeamId] [uniqueidentifier] NOT NULL,

[UserId] [uniqueidentifier] NOT NULL,

PRIMARY KEY CLUSTERED

(

[TeamId] ASC,

[UserId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

/\*\*\*\*\*\* Object: Table [Newsfeed] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [Newsfeed](

[Id] [uniqueidentifier] NOT NULL,

[EmployeeId] [uniqueidentifier] NOT NULL,

[Description] [varchar](510) NULL,

[AchievedTime] [datetime] NOT NULL,

[Reward] [varchar](255) NULL,

[isNew] [bit] NOT NULL,

PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

/\*\*\*\*\*\* Object: Table [Rank] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [Rank](

[Id] [uniqueidentifier] NOT NULL,

[TeamId] [uniqueidentifier] NOT NULL,

[week] [datetime] NOT NULL,

[ActualScore] [bigint] NOT NULL,

[TotalScore] [bigint] NOT NULL,

[ActualWeightedScore] [bigint] NOT NULL,

[TotalWeightedScore] [bigint] NOT NULL,

PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

/\*\*\*\*\*\* Object: Table [Rank2] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [Rank2](

[Id] [uniqueidentifier] NOT NULL,

[TeamId] [uniqueidentifier] NOT NULL,

[AssetId] [uniqueidentifier] NOT NULL,

[Week] [datetime] NOT NULL,

[Detected] [bit] NOT NULL,

[ExtraTag] [bit] NOT NULL,

PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

/\*\*\*\*\*\* Object: Table [Team] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [Team](

[Id] [uniqueidentifier] NOT NULL,

[Name] [varchar](100) NOT NULL,

[IsActive] [bit] NOT NULL,

PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

;

/\*\*\*\*\*\* Object: Table [User] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [User](

[Id] [uniqueidentifier] NOT NULL,

[FirstName] [varchar](100) NOT NULL,

[LastName] [varchar](100) NOT NULL,

[Email] [varchar](100) NOT NULL,

[Image] [varchar](max) NULL,

[Role] [varchar](100) NOT NULL,

[IsActive] [bit] NOT NULL,

[level] [numeric](18, 0) NOT NULL,

[exp] [numeric](18, 0) NOT NULL,

[rank] [numeric](18, 0) NOT NULL,

[StartDate] [datetime] NOT NULL,

CONSTRAINT [PK\_\_User\_\_3214EC077FBDE21C] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

;

/\*\*\*\*\*\* Object: Table [UserBadges] Script Date: 4/14/2020 6:18:32 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

;

SET QUOTED\_IDENTIFIER ON

;

CREATE TABLE [UserBadges](

[Id] [uniqueidentifier] NULL,

[UserId] [uniqueidentifier] NULL,

[BadgeId] [uniqueidentifier] NULL

) ON [PRIMARY]

;

ALTER TABLE [Asset] ADD DEFAULT ((1)) FOR [Weight]

;

ALTER TABLE [Asset] ADD DEFAULT ((0.00)) FOR [Cost]

;

ALTER TABLE [Challenges] ADD CONSTRAINT [DF\_Challenges\_userCreated] DEFAULT ((1)) FOR [userCreated]

;

ALTER TABLE [Challenges] ADD CONSTRAINT [DF\_Challenges\_difficulty] DEFAULT ((0)) FOR [difficulty]

;

ALTER TABLE [Newsfeed] ADD CONSTRAINT [DF\_Newsfeed\_isNew] DEFAULT ((0)) FOR [isNew]

;

ALTER TABLE [Rank] ADD DEFAULT ((0)) FOR [ActualScore]

;

ALTER TABLE [Rank] ADD DEFAULT ((0)) FOR [TotalScore]

;

ALTER TABLE [Rank] ADD DEFAULT ((0)) FOR [ActualWeightedScore]

;

ALTER TABLE [Rank] ADD DEFAULT ((0)) FOR [TotalWeightedScore]

;

ALTER TABLE [Rank2] ADD DEFAULT ((0)) FOR [Detected]

;

ALTER TABLE [Rank2] ADD DEFAULT ((0)) FOR [ExtraTag]

;

ALTER TABLE [Team] ADD DEFAULT ((1)) FOR [IsActive]

;

ALTER TABLE [User] ADD CONSTRAINT [DF\_\_User\_\_IsActive\_\_398D8EEE] DEFAULT ((1)) FOR [IsActive]

;

ALTER TABLE [User] ADD CONSTRAINT [DF\_User\_level] DEFAULT ((0)) FOR [level]

;

ALTER TABLE [User] ADD CONSTRAINT [DF\_User\_exp] DEFAULT ((0)) FOR [exp]

;

ALTER TABLE [User] ADD CONSTRAINT [DF\_User\_rank] DEFAULT ((0)) FOR [rank]

;

ALTER TABLE [User] ADD CONSTRAINT [DF\_User\_StartDate] DEFAULT ('2020-10-02 00:00:00') FOR [StartDate]

;

ALTER TABLE [Challenges] WITH CHECK ADD CONSTRAINT [FK\_Challenges\_Badges] FOREIGN KEY([Badge])

REFERENCES [Badges] ([ID])

ON UPDATE CASCADE

;

ALTER TABLE [Challenges] CHECK CONSTRAINT [FK\_Challenges\_Badges]

;

ALTER TABLE [Challenges] WITH CHECK ADD CONSTRAINT [FK\_Challenges\_User] FOREIGN KEY([User])

REFERENCES [User] ([Id])

;

ALTER TABLE [Challenges] CHECK CONSTRAINT [FK\_Challenges\_User]

;

ALTER TABLE [MemberOf] WITH CHECK ADD FOREIGN KEY([TeamId])

REFERENCES [Team] ([Id])

;

ALTER TABLE [MemberOf] WITH CHECK ADD CONSTRAINT [FK\_\_MemberOf\_\_UserId\_\_4D94879B] FOREIGN KEY([UserId])

REFERENCES [User] ([Id])

;

ALTER TABLE [MemberOf] CHECK CONSTRAINT [FK\_\_MemberOf\_\_UserId\_\_4D94879B]

;

ALTER TABLE [Rank] WITH CHECK ADD FOREIGN KEY([TeamId])

REFERENCES [Team] ([Id])

;

ALTER TABLE [Rank2] WITH CHECK ADD FOREIGN KEY([AssetId])

REFERENCES [Asset] ([Id])

;

ALTER TABLE [Rank2] WITH CHECK ADD FOREIGN KEY([TeamId])

REFERENCES [Team] ([Id])

;

ALTER TABLE [UserBadges] WITH CHECK ADD CONSTRAINT [FK\_UserBadges\_Badges] FOREIGN KEY([BadgeId])

REFERENCES [Badges] ([ID])

;

ALTER TABLE [UserBadges] CHECK CONSTRAINT [FK\_UserBadges\_Badges]

;

ALTER TABLE [UserBadges] WITH CHECK ADD CONSTRAINT [FK\_UserBadges\_User] FOREIGN KEY([UserId])

REFERENCES [User] ([Id])

;

ALTER TABLE [UserBadges] CHECK CONSTRAINT [FK\_UserBadges\_User]

;

ALTER TABLE [Asset] WITH CHECK ADD CHECK (([Cost]>=(0.00) AND [Cost]<=(9999999999999999.99)))

;

ALTER TABLE [Asset] WITH CHECK ADD CHECK (([Weight]>=(1) AND [Weight]<=(10)))

;

ALTER TABLE [Rank] WITH CHECK ADD CHECK (([ActualScore]>=(0) AND [ActualScore]<=(9223372036854775807.)))

;

ALTER TABLE [Rank] WITH CHECK ADD CHECK (([ActualWeightedScore]>=(0) AND [ActualWeightedScore]<=(9223372036854775807.)))

;

ALTER TABLE [Rank] WITH CHECK ADD CHECK (([TotalScore]>=(0) AND [TotalScore]<=(9223372036854775807.)))

;

ALTER TABLE [Rank] WITH CHECK ADD CHECK (([TotalWeightedScore]>=(0) AND [TotalWeightedScore]<=(9223372036854775807.)))

;

## 4.2 Shard Manager query

Begin

CREATE TABLE [dbo].[Company](

[Id] [uniqueidentifier] NOT NULL,

[Name] [varchar](100) NOT NULL,

[ShardName] [varchar](100) NOT NULL UNIQUE,

[CreatedDate] [datetime] NOT NULL DEFAULT (getdate()),

primary key([Id]));

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUsers](

[Id] [nvarchar](450) NOT NULL,

[UserName] [nvarchar](256) NULL,

[NormalizedUserName] [nvarchar](256) NULL,

[Email] [nvarchar](256) NULL,

[NormalizedEmail] [nvarchar](256) NULL,

[EmailConfirmed] [bit] NOT NULL,

[PasswordHash] [nvarchar](max) NULL,

[SecurityStamp] [nvarchar](max) NULL,

[ConcurrencyStamp] [nvarchar](max) NULL,

[PhoneNumber] [nvarchar](max) NULL,

[PhoneNumberConfirmed] [bit] NOT NULL,

[TwoFactorEnabled] [bit] NOT NULL,

[LockoutEnd] [datetimeoffset](7) NULL,

[LockoutEnabled] [bit] NOT NULL,

[AccessFailedCount] [int] NOT NULL,

[CompanyId] [nvarchar](450) NULL,

CONSTRAINT [PK\_AspNetUsers] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUserTokens](

[UserId] [nvarchar](450) NOT NULL,

[LoginProvider] [nvarchar](450) NOT NULL,

[Name] [nvarchar](450) NOT NULL,

[Value] [nvarchar](max) NULL,

CONSTRAINT [PK\_AspNetUserTokens] PRIMARY KEY CLUSTERED

(

[UserId] ASC,

[LoginProvider] ASC,

[Name] ASC

)WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

ALTER TABLE [dbo].[AspNetUserTokens] WITH CHECK ADD CONSTRAINT [FK\_AspNetUserTokens\_AspNetUsers\_UserId] FOREIGN KEY([UserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserTokens] CHECK CONSTRAINT [FK\_AspNetUserTokens\_AspNetUsers\_UserId]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUserRoles](

[UserId] [nvarchar](450) NOT NULL,

[RoleId] [nvarchar](450) NOT NULL,

CONSTRAINT [PK\_AspNetUserRoles] PRIMARY KEY CLUSTERED

(

[UserId] ASC,

[RoleId] ASC

)WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[AspNetUserRoles] WITH CHECK ADD CONSTRAINT [FK\_AspNetUserRoles\_AspNetRoles\_RoleId] FOREIGN KEY([RoleId])

REFERENCES [dbo].[AspNetRoles] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserRoles] CHECK CONSTRAINT [FK\_AspNetUserRoles\_AspNetRoles\_RoleId]

GO

ALTER TABLE [dbo].[AspNetUserRoles] WITH CHECK ADD CONSTRAINT [FK\_AspNetUserRoles\_AspNetUsers\_UserId] FOREIGN KEY([UserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserRoles] CHECK CONSTRAINT [FK\_AspNetUserRoles\_AspNetUsers\_UserId]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUserLogins](

[LoginProvider] [nvarchar](450) NOT NULL,

[ProviderKey] [nvarchar](450) NOT NULL,

[ProviderDisplayName] [nvarchar](max) NULL,

[UserId] [nvarchar](450) NOT NULL,

CONSTRAINT [PK\_AspNetUserLogins] PRIMARY KEY CLUSTERED

(

[LoginProvider] ASC,

[ProviderKey] ASC

)WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

ALTER TABLE [dbo].[AspNetUserLogins] WITH CHECK ADD CONSTRAINT [FK\_AspNetUserLogins\_AspNetUsers\_UserId] FOREIGN KEY([UserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserLogins] CHECK CONSTRAINT [FK\_AspNetUserLogins\_AspNetUsers\_UserId]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUserClaims](

[Id] [int] IDENTITY(1,1) NOT NULL,

[UserId] [nvarchar](450) NOT NULL,

[ClaimType] [nvarchar](max) NULL,

[ClaimValue] [nvarchar](max) NULL,

CONSTRAINT [PK\_AspNetUserClaims] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

ALTER TABLE [dbo].[AspNetUserClaims] WITH CHECK ADD CONSTRAINT [FK\_AspNetUserClaims\_AspNetUsers\_UserId] FOREIGN KEY([UserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserClaims] CHECK CONSTRAINT [FK\_AspNetUserClaims\_AspNetUsers\_UserId]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetRoles](

[Id] [nvarchar](450) NOT NULL,

[Name] [nvarchar](256) NULL,

[NormalizedName] [nvarchar](256) NULL,

[ConcurrencyStamp] [nvarchar](max) NULL,

CONSTRAINT [PK\_AspNetRoles] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[\_\_EFMigrationsHistory](

[MigrationId] [nvarchar](150) NOT NULL,

[ProductVersion] [nvarchar](32) NOT NULL,

CONSTRAINT [PK\_\_\_EFMigrationsHistory] PRIMARY KEY CLUSTERED

(

[MigrationId] ASC

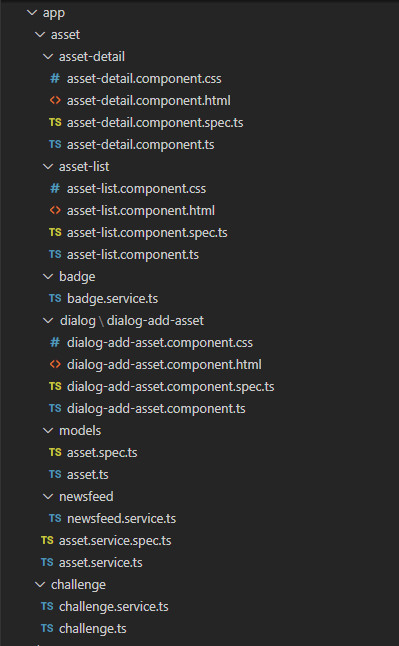
)WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF) ON [PRIMARY]

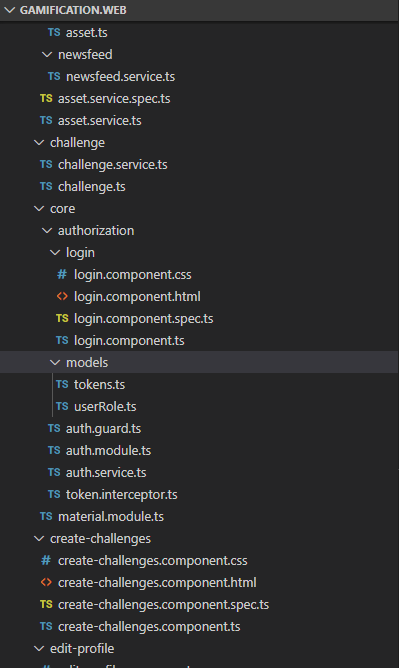
) ON [PRIMARY]

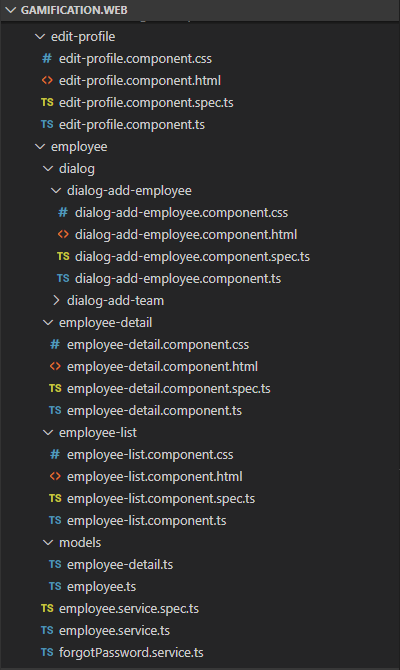
GO

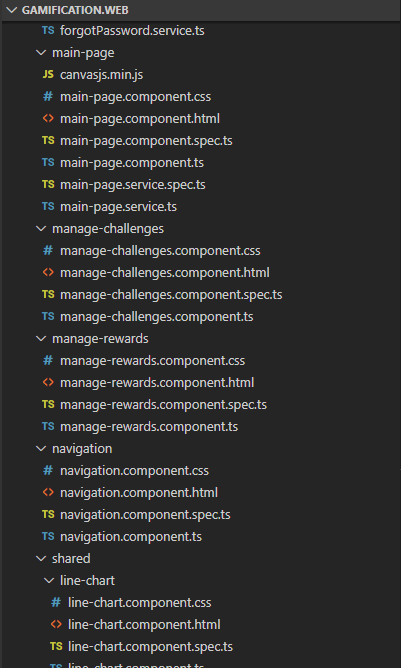
End

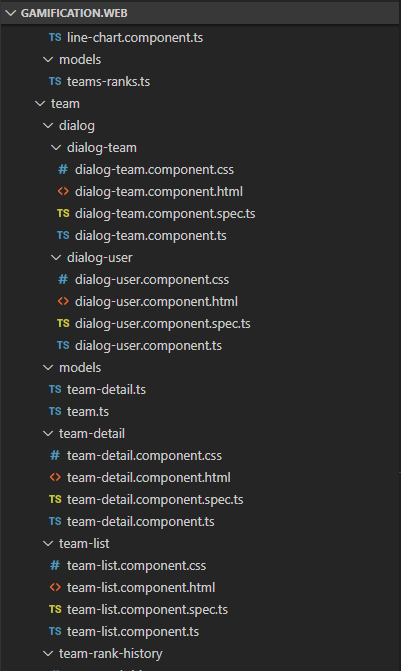
## 4.3 Front End Files

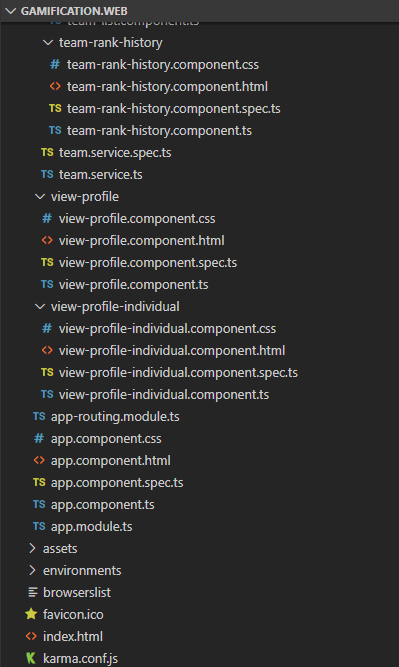






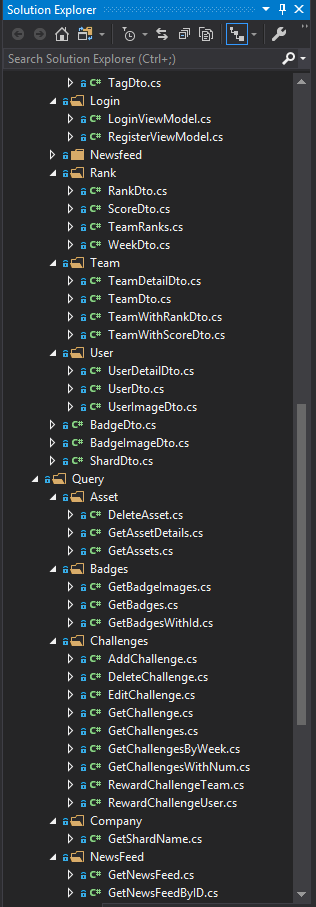


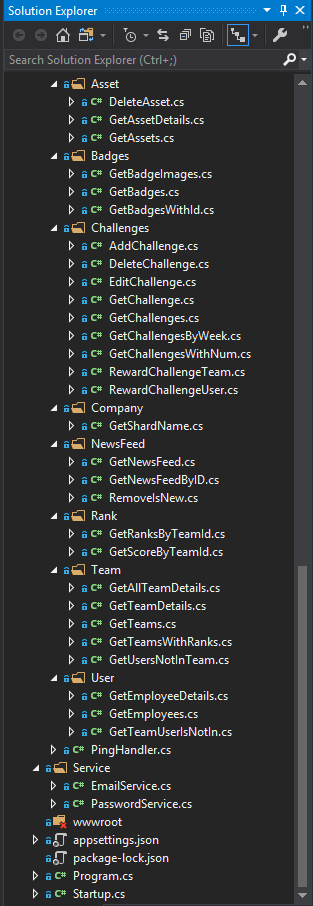




## 4.4 Back End Files

# 





# 5.0 Testing/SQA

## 5.1 Scope and intent of SQA activities

The team’s objective is to ensure that the product does not deviate far from the original gamification design specifications. If it is discovered that deviation has occurred, the team will notify each member to prevent future deviations and to correct the previous deviations. Also, the team will perform a walkthrough to analyze the product’s quality at any stage of development. Error detection and possible enhancements are also expressed to each team member.

## 5.2 SQA organizational role

The team organizational role is to review the product at specific times during product implementation. Upon reviewing, the team member duties will be to evaluate the software at its current development stage and recognize any defects in the subsequent stage (design or implementation). The team will directly interact with the customer in group discussions, discussing any errors or possible enhancements that have been identified. In addition, the team will ensure that the customer has not deviated in any way from the initial design specifications.

## 5.3 SQA Tasks

### 5.3.1 Task Overview

#### 5.3.1.1 Description of SQA Task 1

The team will meet with Sargon Partners every other week to adjust requirements if need be. Every week, the lead software engineer will match the software to the current design requirements agreed upon at meetings. This process will help ensure that the software development will be handled correctly. The weekly checks will consist of matching the requirements to individual segments of code and assessing their correctness.

##### 5.3.1.1.1 Work Products and Documentation for Task 1

As a result of Task 1, internal meeting minutes will be taken during meetings and reviewed at internal meetings. The meeting minutes serve as a solid baseline for our team meetings.

#### 5.3.2 Description of SQA Task 2

The design lead will develop initial mockups for the 8\* individual pages that the website is required to have. Two pages of mockups must be finished for each meeting with our clients to assess correctness and make potential changes to better fit the overall requirements. The lead will distribute the work evenly between group members and hold individual meetings to address small issues and keep the design unified.

##### 5.3.2.1 Work Products and Documentation for Task 2

As a result of Task 2, the rough drafts/mockups will be saved in our design requirements appendix.

#### 5.3.3 Description of SQA Task 3

Each group member will have to periodically go through the website to perform reliability testing. Individual use should help our team find the smaller, potentially lower priority, errors early on. Performing periodic testing throughout the life of our project will help ensure we can mitigate problems closer to when they arise. Each member will be responsible for checking new functionality, along with existing functions.

##### 5.3.3.1 Work Products and Documentation for Task 3

As a result of Task 3, a short report will be written for our client meetings. The reports will cover issues found, their priority, and resolution. The reports will be collected in a defect log.

## 5.4 Standards, Practices and Conventions (SPC)

Documentation Standards: Whenever applicable, documents will all follow the same format and layout. This layout was predetermined by our group before the project began.

Coding Standards: Standard AWS RDS naming conventions and format will be followed for the database. HTML5/CSS3 standards will be upheld for the website creation.

We will also be following C and Java best practices.

## 5.5 SQA Resources

* Eclipse IDE – Java testing and development
* Visual Studio Code – HTML testing and development
* Visual Studio Community – C# testing and development
* AWS RDS – database development and testing
* Microsoft Azure – database development and testing
* Figma – page design mockups
* Google Drive – collaborative document management

## 5.6 Reviews and Audits

### 5.6.1 Generic Review Guidelines

#### 5.6.1.1 Conductions a Review

The software will have scheduled reviews to detect any defects in the current prototype, and to determine any notable enhancements that should be implemented prior to the final product. The current prototype will be broken down into smaller subsets and each subset will be reviewed. The team will attend each review and critique each other’s part of the software to ensure that the maximum number of possible defects are accounted for. The group will meet on a weekly basis to discuss defects or enhancements. Each meeting will consist of no more than a couple of hours. The current prototype for both the user-interface and the back-end will be available at every meeting allowing defects to be explicitly pointed out to each group member. If the defect found could not be easily duplicated, the Development leader will take note of the defect. Any desired enhancements will be discussed to determine which are necessary and feasible. Any enhancement found to be too difficult or unnecessary for product completion will be noted by the Development leader.

#### 5.6.1.2 Roles and Responsibilities

The Development leader will oversee any formal technical reviews. Any defects or enhancements will be discussed and recorded by the Development leader. Each defect or enhancement will be given a priority rank, which will be recorded. Once the review is complete, the Development leader will make a summary of each defect or enhancement and distribute them to the appropriate member of the team.

Each member will be responsible for reviewing his own software module during module creation and upon module completion. Once each major software module is complete, it is the members duty to inform the Development leader that the module is ready for review.

#### 5.6.1.3 Review work products

The Development leader will keep a defect log. The defect log contains all defects and enhancements, as well as a priority rank for each. The following will also be noted in the defect log: (1) Priority level of the defect, (2) whether the defect or enhancement has been handled, (3) which software engineer oversaw the correction, and (4) what date the correction was completed.

### 5.6.2 Formal Technical Reviews

#### 5.6.2.1 Description of SPMP review

##### 5.6.2.1.1 Description and Focus of the review

The software project management plan review will focus on ensuring the overall scope of the project is correct and that any updates that need to occur will happen and be documented correctly will all parties signing off on any and all scope changes.

##### 5.6.2.1.2 Timing of the review

The document will be reviewed at the start of the second semester or when the client requests a change in the scope of the project.

##### 5.6.2.1.3 Work products produced

The SPMP will be updated and correct versioning will be observed by storing a copy of the old document for review purposes. This will be an additional document created to layout all changes that took place and why. This will keep all parties informed on the current scope.

##### 5.6.2.1.4 Review checklist

* Review old SPMP for possible updates
* Talk with clients about any changes to project scope and document all in separate document to track needed changes
* Version and update new document
* Get signoff by team lead and client

### 5.6.3 Description of RMMM review

#### 5.6.3.1 Description and Focus of the review

The RMMM review will focus on determining if the proposed risk management for the development of this software is within reason. The focus will be on if the risk management can handle a proposed problem accordingly. If the RMMM document is not managing each risk accordingly, the RMMM will be amended to correct the oversight.

#### 5.6.3.2 Timing of the review

The RMMM review will be held upon completion of the RMMM document. This should occur within the first few weeks of the software’s development. If any kind of updates are needed for the RMMM document all team members will inform each other, and when the document is updated it will be reviewed by the team.

#### 5.6.3.3 Work products produced

The team leader will create a summary report of the RMMM review. This includes any possible risks that have not been covered, and any risks that have been accounted for, but are not managed appropriately. Once a new risk is proposed, a discussion will take place on how to handle the risk if it were to occur. Any risks being managed inappropriately will also be discussed and an amendment to their management will be made to better handle the risk.

#### 5.6.3.4 Review checklist

* Have all risks been thoroughly covered in the document? If not, what is missing?
* Of the risks covered in the document, are there any that did not seem to be effectively covered? If yes, which one(s)?
* Of the risks covered in the document, are there any that did not seem to be appropriately managed? If yes, which one(s)?

### 5.6.4 Description of Requirements review

#### 5.6.4.1 Description and Focus of the review

The Requirements Specification review will work to analyze the proposed design of the software. The focus of this review will be to remove or discuss changes to any obvious design flaws. Once a design defect has been encountered, the Development team will discuss with the other team members for any ideas or suggestions about how to compensate for the design flaw.

#### 5.6.4.2 Timing of the review

The Requirements Specification review will be held upon completion of the Requirements Specification. This should occur within the first few weeks of the software’s development. To ensure that the software is conforming to the restrictions of the design, each team member will frequently conduct his own Requirements Specification review. If a team member determines that the current design of a module is flawed, it will be brought to the attention of the Design leader and an appropriate discussion to amend the problem will be held.

#### 5.6.4.3 Work products produced

The Document leader will create a summary report of the Requirements Specification Review. This includes any defects or enhancements that have been brought to attention. Once design defects have been identified, the Document team will discuss possible solutions with the team members. Each possible solution will be noted and reviewed again to determine if the solution will have an impact on the rest of the design. Once all obvious design defects have been handled, the Requirements Specification will be amended to account for the design changes.

#### 5.6.4.4 Review checklist

* Is the proposed design the best possible solution?
* Are there any obvious design flaws that have not been accounted for? If yes, what?
* Are there any necessary enhancements for the software?
* Is the proposed Requirements Specification within the time frame?

### 5.6.5 Description of Architectural design review

#### 5.6.5.1 Description and Focus of the review

The Architectural Design review will provide a basis for analysis of the proposed architectural design of the software. The focus of this review will be to assess the current design to ensure that data flow and data control are being handled appropriately. If a design flaw has been discovered, the team leader will discuss with the team members any ideas or suggestions about how to compensate for the architectural design flaw.

#### 5.6.5.2 Timing of the review

The Architectural Design review will be held upon completion of the Requirements Specification and the System Specification. This should occur within the first few weeks of the software’s development. This is necessary to ensure that the underlying architectural design of the software is sound and will not create problems for the software engineers or degrade the software’s performance in the future.

Each team member will frequently conduct his own Architectural Design review to ensure that the software is conforming to the restrictions of the architectural design. If the team member determines that the current architectural design of a module is flawed, it will be brought to the attention of the team leader and an appropriate discussion to amend the problem will be held.

#### 5.6.5.3 Work products produced

The team leader will create a summary report of the Architectural Design review. This includes any defects in the architectural design that has been brought to attention. Once architectural design defects have been identified, the team leader will discuss possible solutions with the team. Each possible solution will be noted and again reviewed to determine if the solution will have an impact on the rest of the design. Once all obvious architectural design defects have been handled, the Requirements Specification and System Specification will be amended to account for any architectural design changes.

#### 5.6.5.4 Review checklist

* Is the proposed architectural design the best possible solution?
* Are there any obvious architectural design flaws that have not been accounted for (i.e. slow data flow or control)? If yes, what?
* Are there any obvious changes you see would further enhance the software’s performance?
* Is the proposed architectural design complete? If not, what seems to be missing?
* Does the proposed architectural design seem possible within languages of choice? If not, why?

### 5.6.6 Description of Interface design review

#### 5.6.6.1 Description and Focus of the review

The Interface design review will be conducted first will team members responsible for the design, look and feel for the website. After the initial review, another review will take place with the clients to insure they are happy with the design. This allows for quick design with good feedback before full implementation.

#### 5.6.6.2 Timing of the review

The first part of the review will be completed by the team members before a meeting with the clients. The review is then completed at the client meeting which occurs on a weekly basis.

#### 5.6.6.3 Work products produced

The interface mockups created will be living documents until implementation for that specific interface. This will allow for quick changes to the designs.

#### 5.6.6.4 Review checklist

* Does the design include all needed functionality?
* Is it pleasing to look at?
* Is the color scheme consistent across designs?
* Is it possible to implement?

### 5.6.7 Description of Code review

#### 5.6.7.1 Description and Focus of the review

The code review is to mainly spread knowledge about a piece of functionality. In the code review teammates are allowed to ask questions, give insight on how code can be written better and even rewrite code altogether.

#### 5.6.7.2 Timing of the review

The reviews will happen every fortnight at our meetings on Wed. If a member is not able to come to the code review, they will be required to complete a code review on their own time, within a 24 hour period.

#### 5.6.7.3 Work products produced

The work products that will be produced from the code reviews are that we will be able to work on the next thing on the roadmap.

The code review will be the last thing that happens to a piece of code before it gets delivered. Once a piece of code’s code review is done, the person whom created the code will be put on a different piece of functionality.

#### 5.6.7.4 Review checklist

* Is the source code reliable ? If not, why?
* Is the source code efficient? If not, why?
* Is the source code highly modular?
* Is the source code easy to read?
* Is the source committed well?
* Are they using any bad practices?
* Is there a simpler way?
* Is the source code allow for easy maintainability?

### 5.6.8 Description of Test specification review

#### 5.6.8.1 Description and Focus of the review

The test specification review will be used to review and analyze our four step test strategy. The focus of this review will be to ensure we follow our individual test strategies. The review period will be conducted before each test stage.

#### 5.6.8.2 Timing of the review

The review period will take place before and after each stage of our test strategy: backend, frontend, integration, and regression testing. The initial review will be used to ensure our test cases are still valid after changes and alterations have been made. The post review will be to go over the test results.

#### 5.6.8.3 Work products produced

The work products produced from the Test specification review will be insight into how to improve our following test specification reviews for the upcoming test strategies.

#### 5.6.8.4 Review checklist

* Are the test specifications being met? If not, why?
* Is the backend being fully tested? If not, why?
* Is the frontend being fully tested? If not, why?
* Is the integration being fully tested? If not why?
* Is the regression testing addressing everything? If not, why?
* Is the test specification review helpful for further testing? If not, why?

### 5.6.9 SQA audits

We believe that there should be a “head” of SQA to oversee all of the SQA activities. We have designated that to the team lead. The head of the SQA will be in charge of making sure that version numbers are being updated, maintaining order during code reviews and automated tests are being created.

## 5.7 Problem Reporting and Corrective Action

### 5.7.1 Reporting Mechanisms

For all minor issues that occur in any FTR, that issue will be reported to every member on the team for feedback on how to fix the issue and proceed with the task at hand. If a large issue occurs the issue will be reported to all team members and the client in order to use all the resources available to fix the issue that arose.

### 5.7.2 Responsibilities

The responsibility to contact the team will be the member that completes the FTR audit. At that point the team will decide if the issue is large enough to escalate the issue to the client. The team lead will send an email to inform the client of any important information at that point.

### 5.7.3 Data collection and evaluation

All error/defect data will be collected at the time it occurs whether the team member deems it necessary or not. The data will be stored in the review document if one exists or a document will be created to store the data along with any information about how the error occurred.

### 5.7.4 Statistical SQA

If the error that occurs is recurring then a basic time series analysis will be performed to gain as much insight about the issue as possible. Bar charts may be used to show trends in the data.

## 5.8 Software Process Improvement Activities

### 5.8.1 Goal and Objectives of SPI

The goals and objectives of the SPI are to lower the frequency of software defects and to determine the underlying cause of the defects that occur. Furthermore, once the underlying causes have been identified, measures will be taken to determine the best course of action to eliminate the problem.

### 5.8.2 SPI tasks and responsibilities

Based on the Statistical information gathered, the Development leader will keep a tally of what errors or causes of errors occur most frequently. The more often a defect occurs from the same underlying cause, the more problematic an area will be considered. Depending on the nature of the cause and the individuals involved, two actions can take place: (1) the Development leader will investigate the Statistical SQA information and the defect log to determine if the problem area exists primarily for a single developer, or (2) if every developer experiences the same problem.

If the problem occurs mostly from one user, the developer will be informed of the frequency of the personal problem area. Most often, the developer has a better idea of how to handle his own implementation oversights.

If the problem occurs from all developer, development practices will be analyzed to determine the cause of the problem. The problem and possible solutions will be examined at a meeting to all the developers.

## 5.9 Software Configuration Management Overview

### 5.9.1 Overview

We will be creating a baseline for each artifact to maintain version control and to be able to view what changes have been made, along with seeing whom was responsible for each of these changes. There will be a project roadmap which will have all of the dates to which each configuration item shall be delivered. These dates have been agreed upon by both the team and the clients. If the dates wish to be changed,it MUST be agreed upon by both the team and client. Last but not least, we will have code reviews to make sure that the code we write is complete, consistent and is well written. In order for a code review to be deemed completed, it must be signed off by all team members. In the event of a member not being present during a code review, the code review will either be pushed back to a further date, or the code review will continue as planned, and the absent member will be required to review the code on their own time, in a timely manner.

### 5.9.2 Approach

All documents required by the Prof. will be baselined. In the event of a change in the document, the previous text that shall be replaced will be highlighted to ensure that the team is able to differentiate the previous version to the new one. The person editing the document will be the person in charge of making this change.

The code will be stored within a repository, where the files will have revision history, along with having the contributor’s name. The code reviews will happen during team meetings and be reviewed from the repository.

During the planning stages of the project, we will devise a roadmap with our clients require a signature from both parties to ensure that the road map will not change unless both parties agree. As previously stated, both clients will need to sign off on a change of delivery.

### 5.9.3 Impact

There’s a wide variety of positive and negative impacts. Some of the positive impacts is that it will hold people accountable. As a team, we will be required to stick to the delivery dates. The clients will also be accountable to give us what they want well in advance. Code reviews allow us, again, to hold people accountable. From code reviews, we will also be able to seek out potential bugs in the software, determine if there are any bad practices that we can rewrite and make sure we’re on the same page of how the code works and are able to ask questions if we are unsure.

Some negative impacts of using this are that time is a big factor within this project. We only have 8 months to totally complete this project. Creating road maps, doing code reviews along with doing version control on artifacts all cost time, which is very limited for us.

## 5.10 SQA Tools, Techniques, Methods

We will be using a number of tools in order to make sure our software is at the highest level possible. Due to the nature of our project, we will be using web testing tools. Some of the tools we will be using include Selenium Web Driver, Cypress.io, and possibly Test.ai.

Along with testing during, we will be creating automated tests that allows us to make sure any of the effort we have done in the past is not broken due to changes in the code. On top of making sure our code stays at its peak we will also be doing a combination of white box and black box testing.

Before anyone submits code to the project, they will have completed white box testing during development and directly after completion of a task. Once the white box testing is completed, the same user will complete black box testing on the said piece of code, the programmer will make sure all of the automated tests work properly, along with creating

new tests (if applicable).

## 5.11 Test Cases

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Expected Output | Actual Output | Pass/Fail | Date Fixed (or N/A) |
| 1 | User should be logged in | User Logged in with their personal information | Pass | N/A |
| 2 | Admin should be able to pull teams from the database | Admin is able to pull teams that have not been able to be pulled | Pass | N/A |
| 3 | Admin should be able to delete a team | Admin is able to delete any team | Pass | N/A |
| 4 | Admin should be able to added an employee to a team | Admin is able to add an employee to a team | Pass | N/A |
| 5 | Admin should be able to remove an employee from a team | Admin is able to delete an employee from a team | Pass | N/A |
| 6 | Admin ,should be able to create an employee account | Admin is able to create an employee account | Pass | N/A |
| 7 | Admin should be able to delete an employee account | Admin is able to delete an employee account | Pass | N/A |
| 8 | The user should now have the badge for the challenge they completed | Badge is displayed next to completed challenge | Pass | Implemented |
| 9 | The user should now be one level higher | User gained a level | Pass | Implemented |
| 10 | The user should be shown their own teams dashboard | User is shown their team dashboards | Pass | N/A |
| 11 | The user should be shown their own profile (Statistics) page | User is shown their own profile page | Pass | N/A |
| 12 | The admin should be able to access all of the team’s dashboards | The admin has access to all the team’s dashboard | Pass | N/A |
| 13 | The user should be able to see their peer’s profile | The user is able to see their peer’s profile | Pass | N/A |
| 14 | A user should be able to reset their password | A user gets an email with their new password | Pass | N/A |
| 15 | The user should be able to see their newly updated profile picture | User is able to see their newly updated profile picture | Pass | N/A |
| 16 | The user’s name should be changed | N/A | N/A | N/A |
| 17 | The users score should reflect the new team’s score, rather than the old | N/A | N/A | N/A |
| 18 | The users score should reflect the old team + the newly added teams score | N/A | N/A | N/A |
| 19 | The users score should reflect the single team they’re on | The users score does reflect the team they are on | Pass | N/A |
| 20 | The styling should remain good looking under compression | N/A | N/A | N/A |
| 21 | Pie Chart should contain colors: Red, Green, and Yellow | Red, Green, and Yellow | Pass | Implemented |
| 22 | Message displayed instead of empty pie chart | No data found message displayed | Pass | Implemented |
| 23 | Error message displayed under login prompt | Error message displayed under login prompt | Pass | Implemented |
| 24 | Error message displayed under login prompt | Error message displayed under login prompt | Pass | Implemented |
| 25 | Challenge created and displayed on Manage Challenge page | Challenge was created and displayed on the page | Pass | Implemented |
| 26 | Challenge created and displayed on Manage Challenge page | Challenge was created and displayed on the page | Pass | Implemented |
| 27 | Challenge created and displayed on Manage Challenge page | Challenge was created and displayed on the page | Pass | Implemented |
| 28 | Challenge created and displayed on Manage Challenge page | Challenge was created and displayed on the page | Pass | Implemented |
| 29 | Challenge created that has a customized reward | Challenge was created and has a user created reward | Pass | Implemented |
| 30 | Challenge should be completed for user and display badge | Challenge displayed with badge | Pass | Implemented |
| 31 | Challenge should be completed for user and display badge | Challenge displayed with badge | Pass | Implemented |
| 32 | Challenge deleted from Admin Challenges list | Challenge was deleted | Pass | Implemented |
| 33 | Challenge deleted from System Challenges list | Challenge was deleted | Pass | Implemented |

# 6.0 Future Maintenance Suggestions

## 6.1 Updating the Service

Whenever the shards get updated, make sure to update the back end service as well. If you do not, the front end will try to make API calls to the back end and the tables that it is looking for will not appear in that shard, which will result in an error.

## 6.2 Updating Base Challenges

Right now there is a set of 8 challenges that are loaded in the database. Down the road, you will want to add more challenges and change up challenges. Along with once you add new base challenges, you might want to update the challenges within the service to auto update.

## 6.3 Updating Base Badges

Right now there is a set of 8 badges that are loaded in the database. Down the road, you will want to add more badges into the database along with updating old badges, if they do not match the current style.

# 7.0 References & Bibliography

* Gathered free icons and saved them in our database from FlatIcon and used them for our badges.
  + <https://www.flaticon.com/packs/rewards-badges-4>
* Applied Icons from Font Awesome to use our web information
  + <https://fontawesome.com/icons?d=gallery>
* Updated our charts from canvas.js to google charts
  + <https://developers.google.com/chart>
* We had to learn AWS and RDS and had to make sure that we are not using a too small of a service but not something to big and we had to read a lot from aws website.
  + <https://aws.amazon.com/about-aws/>

# Appendix A - User Manual

## Getting Started

Clone the repository from Github by using the following command

git clone <https://github.com/Ckozan/Gamification2020.git>

Once you have the repository installed locally, you will want to run the service and the back end using the .sln files within the project directory.

## Creating a new shard

To create a new shard, you must do the following

* Start the back end
  + To run the back end, you must go into Gamification and click on the Gamification.sln. This will open up Visual studio. Once in visual studio, click run.
* Start the service
  + To run the service, you must go to the project and click on Gamification.Service.sln. Once in visual studio, click on run.

Once both of those services are up, you are ready to create a new shard. The Service uses a httppost endpoint to create a new shard.

The endpoint is: localhost:7071/api/shardCreator?name=<name>

I.e. Creating a new endpoint for myself would look like localhost:7071/api/shardCreator?name=Cam

Along with the URL, you must have a body along with the request. The body is setup like so:

{

"Xapikey": "F9E6BA00-071B-46CD-85A9-7D41C9923737",

"CompanyName": "Company Q",

"Email": "test@umich.edu",

"EmailPassword": "Yeet123"

}

Xapikey is the company ID in your system,

Company name is the name of the company

Email is the email to be associated with the admin login

Email password is the password to be used with the admin login

P.S. This service takes about 2 minutes to run, once you see the message:

Hello, <Name>

This will be the indication that the new shard has been created.

## Installing dependencies & running the front end

Go into Gamification/Gamification.Web within the console and type the command

npm install

This should install all required packages for the front end project

Once you’ve installed all the packages, type the command

ng serve OR ng serve --host 0.0.0.0

using ng server --host 0.0.0.0 will put the project up online

## Running the back end online

So, in order to set up the back end to be live on the internet (not localhost), you must configure the project to not do so.

Go into Program.cs and in the code, you should see:

public static IWebHost BuildWebHost(string[] args) =>

WebHost.CreateDefaultBuilder(args)

.UseStartup<Startup>()

.Build();

Change it to

public static IWebHost BuildWebHost(string[] args) =>

WebHost.CreateDefaultBuilder(args)

.UseStartup<Startup>()

.UseUrls("[http://\*:50547](about:blank)") //At the time of writing this, the current port we are using is 50547

.Build();

After you’ve done that, run it and it should be online!

## EC2 Information

In order to get the project up on EC2, you must create an elastic IP and associate it with a running EC2 Instance. Once you’ve done that, you also have to configure the security settings to allow the ports to be connected to.

My recommendation was to upgrade from t2.medium to t2.large to make the website smoother along with being able to handle all the back end, front end and the service running at the same time.

## New Shard Configuration

In order to get the new shard configured properly, you must do a few different things. The first of which is go to the Manage Teams, click on Add, which allows you to add a new team. Once there, add all of the teams you wish, and hit save.

After doing so, go to Manage Assets, click on add, and add all of the assets you wish to add to the company. After that, hit Save (P.S. Adding a bunch of new assets at a time may take a minute for them to all be added into the database.)

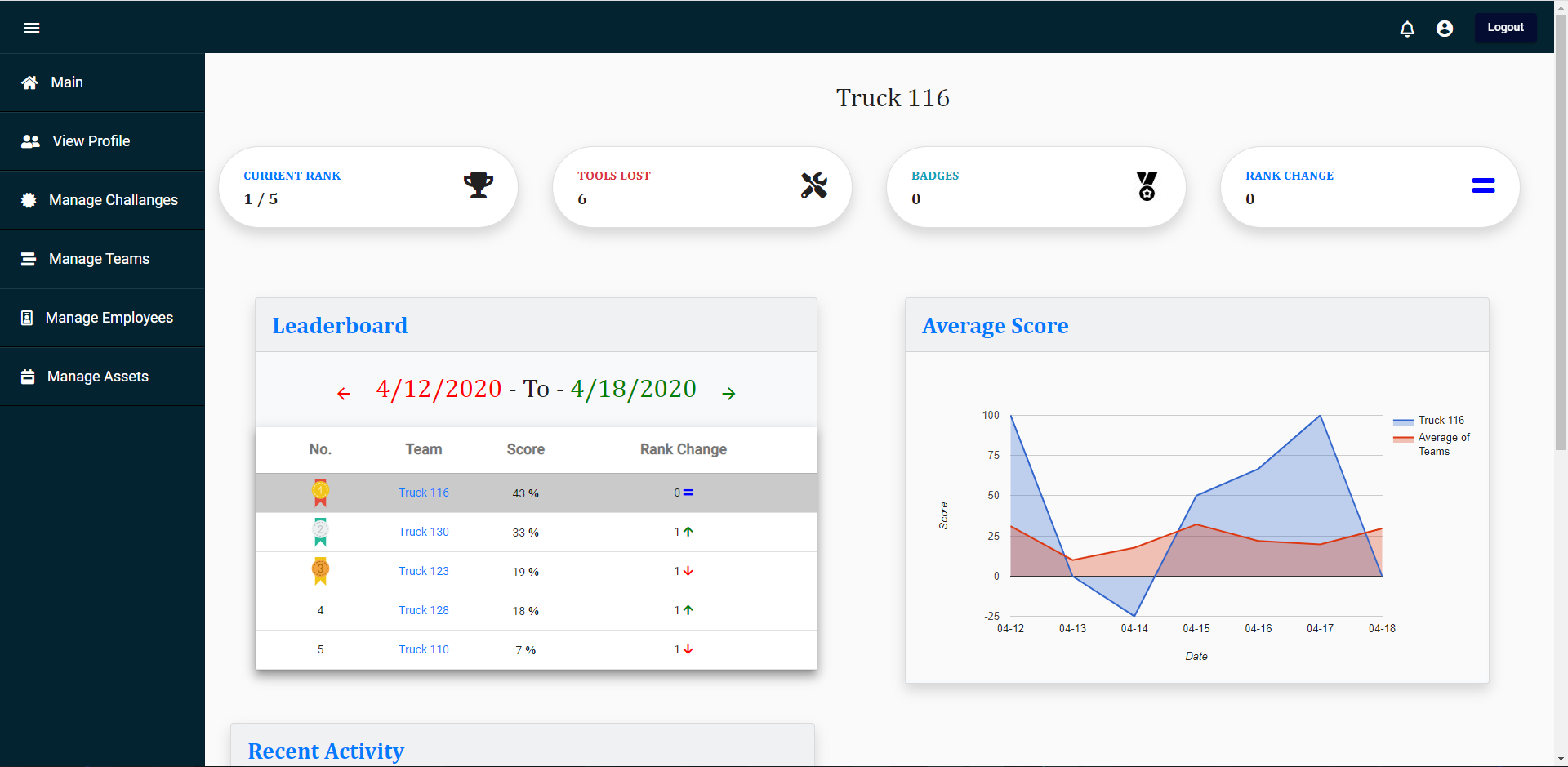
Once you’ve done that, the service will be ready to add new assets into the database at midnight.

## Finishing Notes

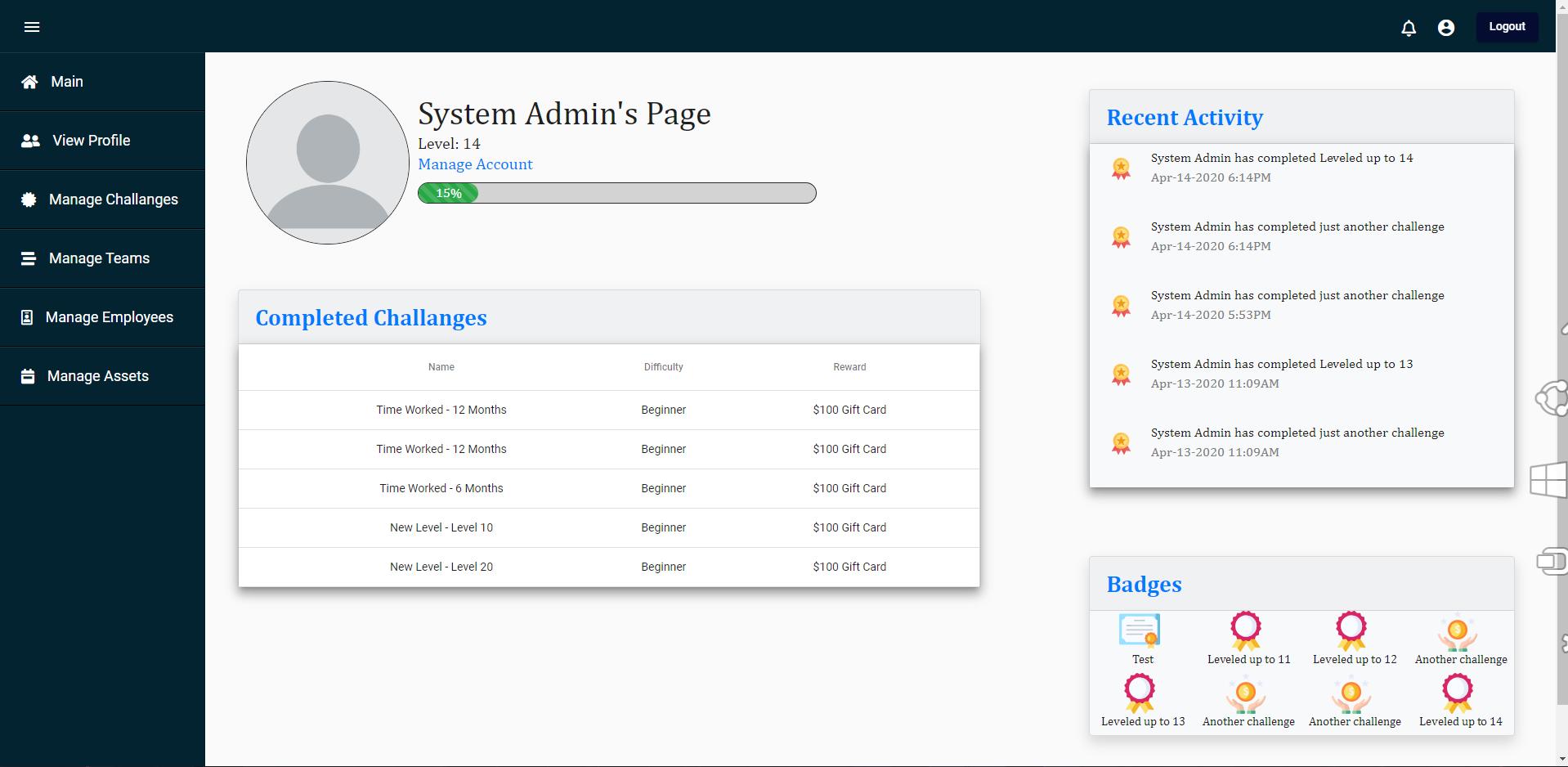
Someone needs to set up a SMTP server and configure the credentials in EmailService.cs. All that needs to be done is adding a Username and Password in the SargonEmail and SargonEmailPassword, respectively.

If anyone updates the tables within the database, make sure to update the schema that creates a new shard within the service. This is crucial and if this doesn’t happen, new shards will not have the newly updated tables.

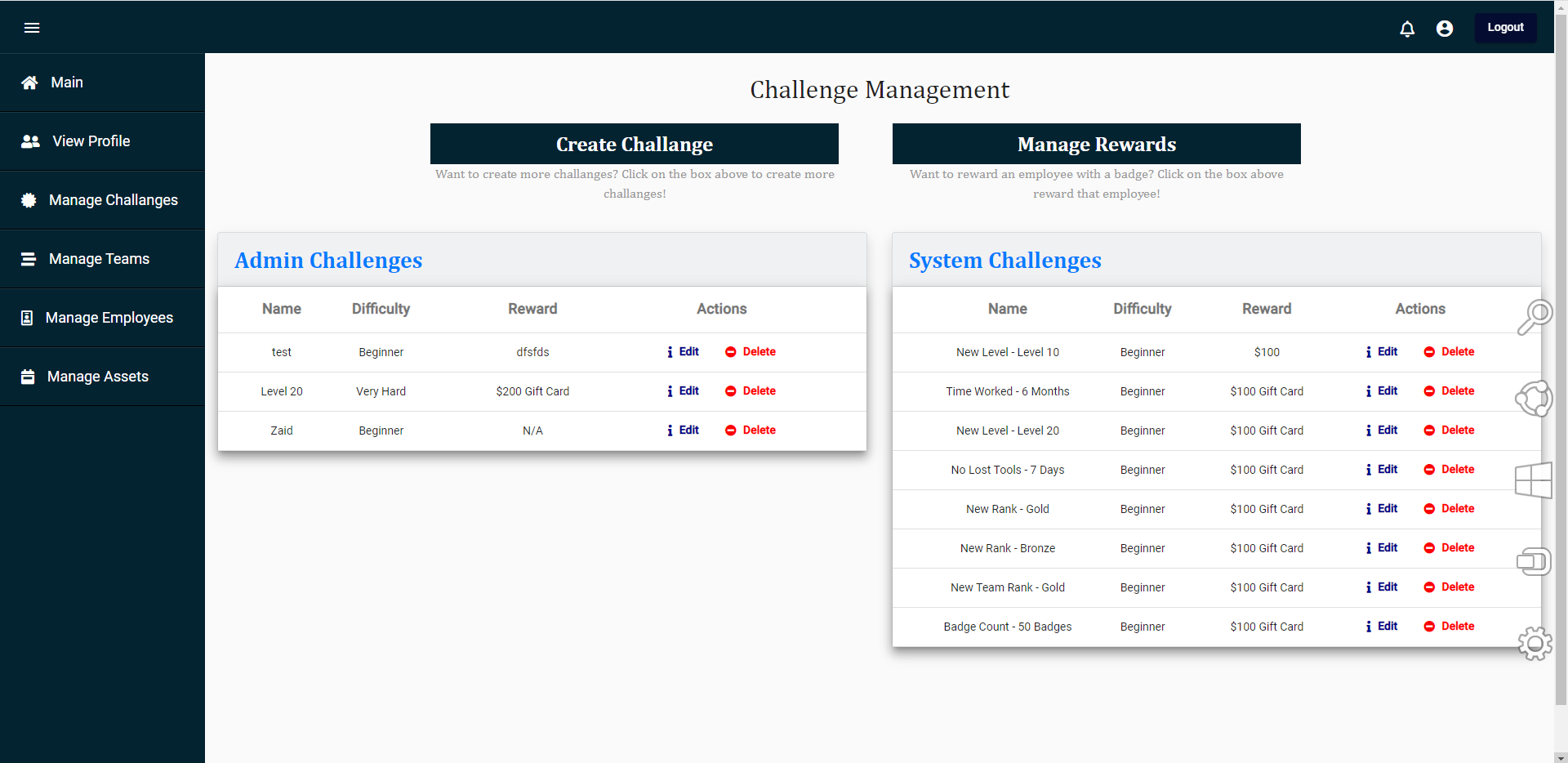
# Appendix B - Program Listing

Dashboard when a user logs in

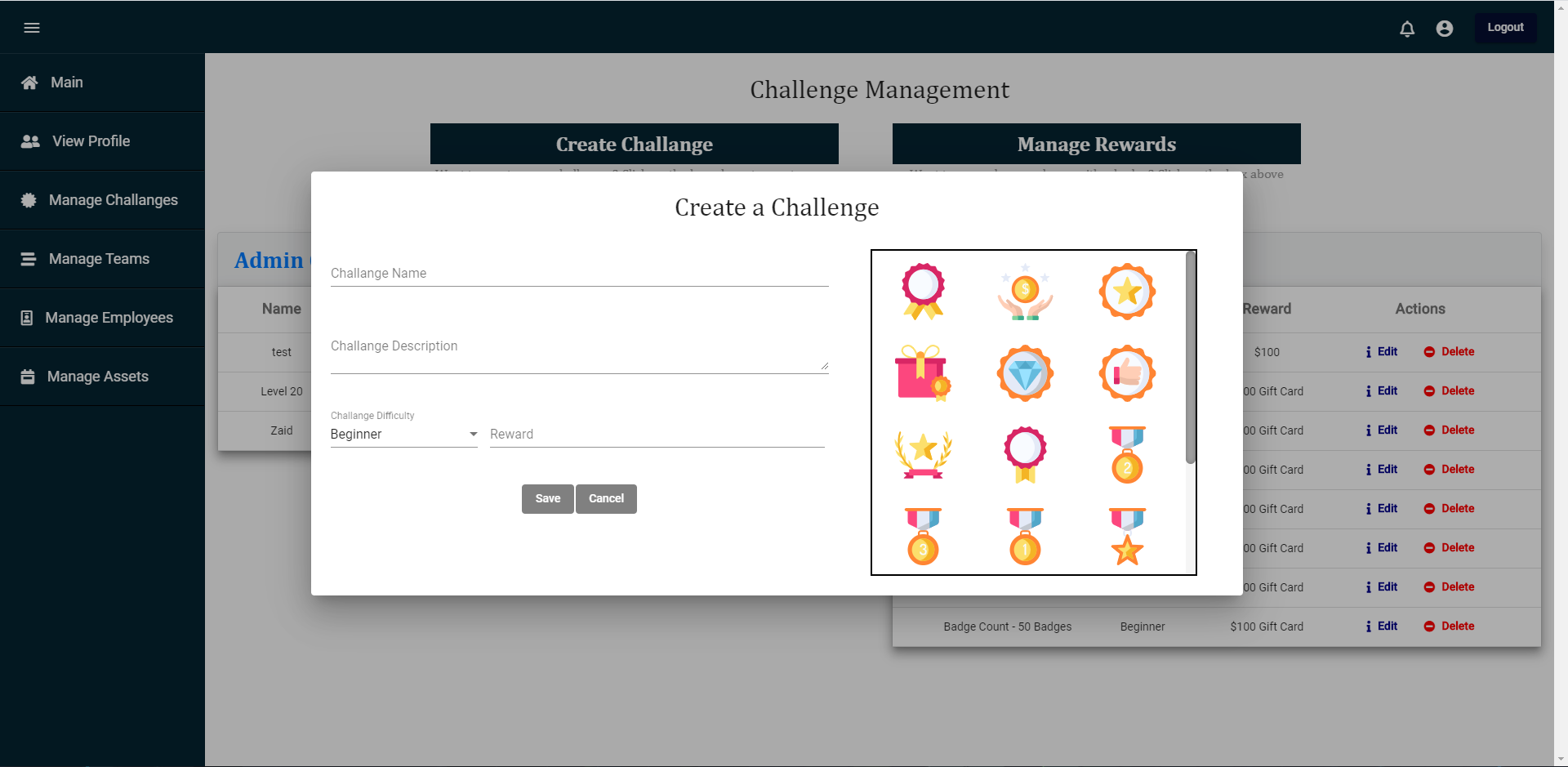
View profile page



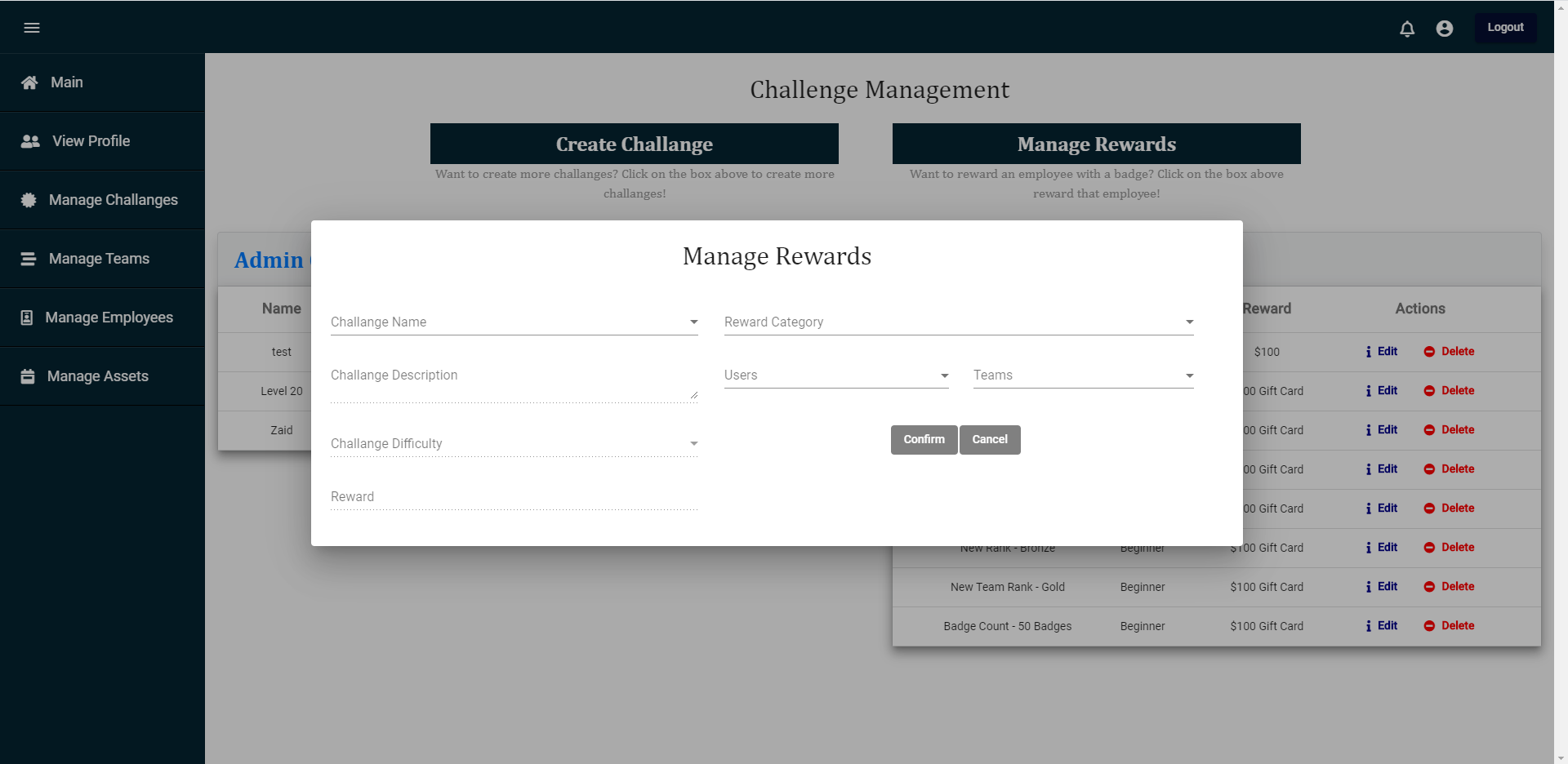
Admin View - Challenge Management page

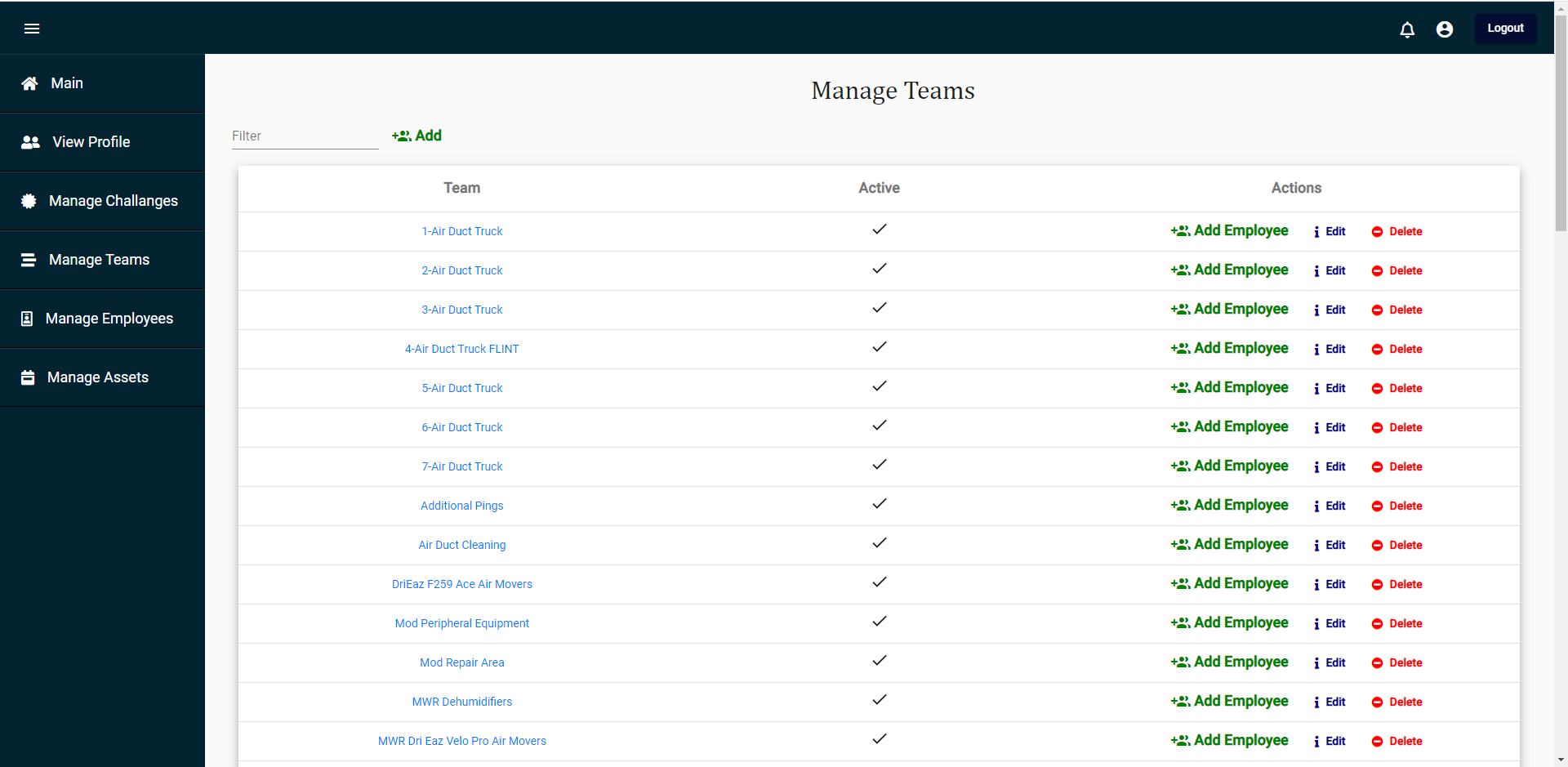


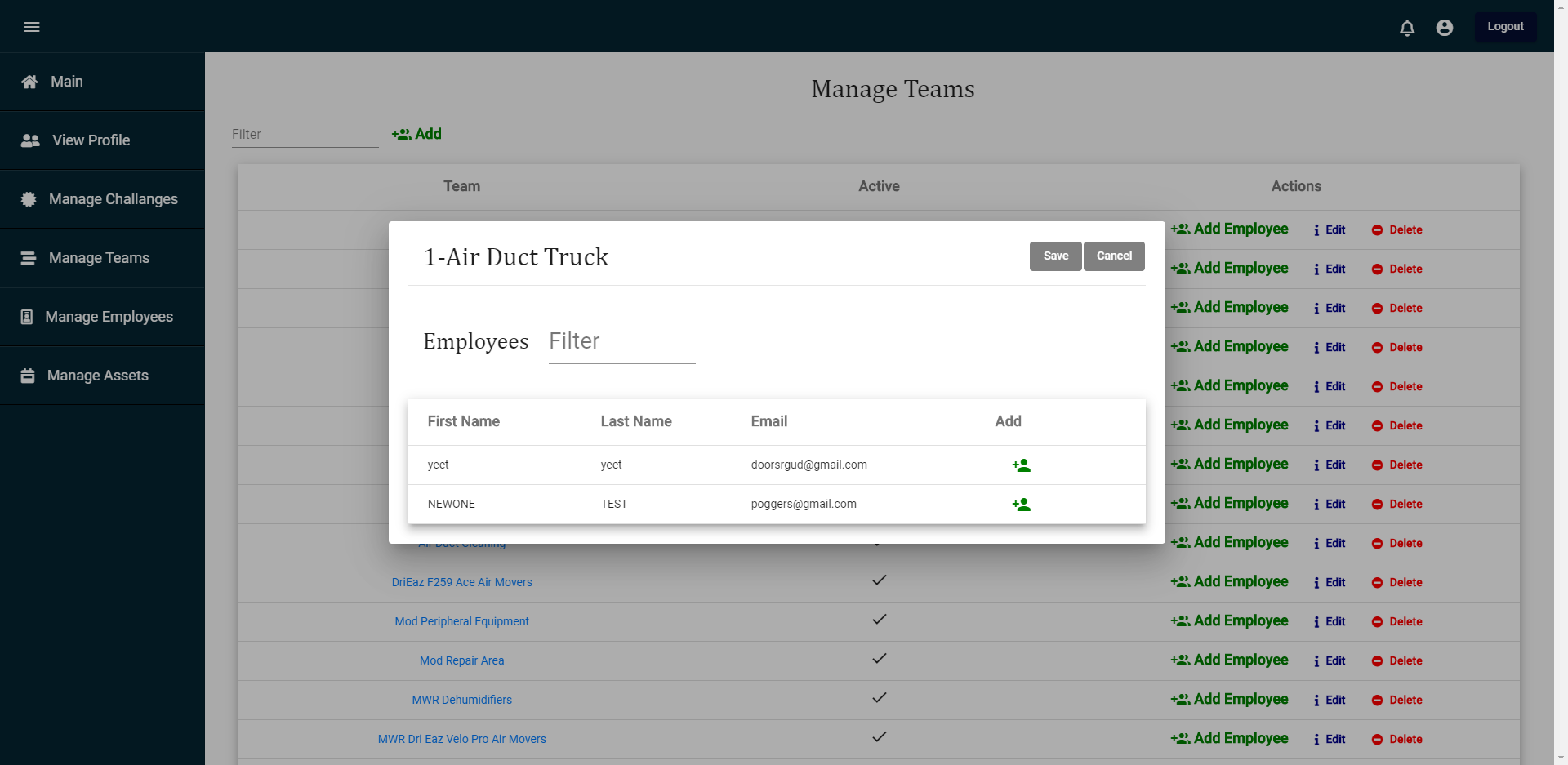
Admin View - Create a Challenge Screen

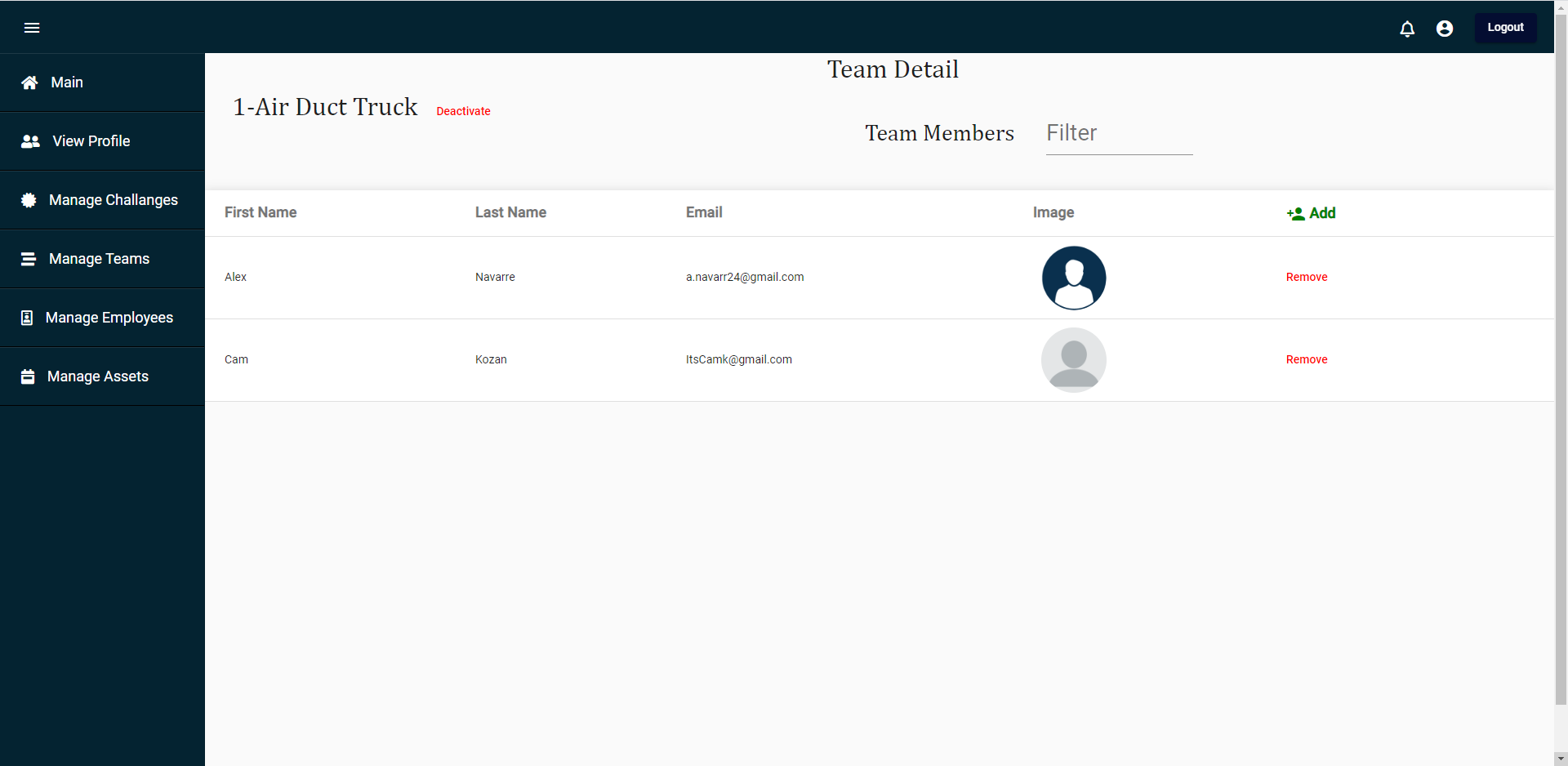


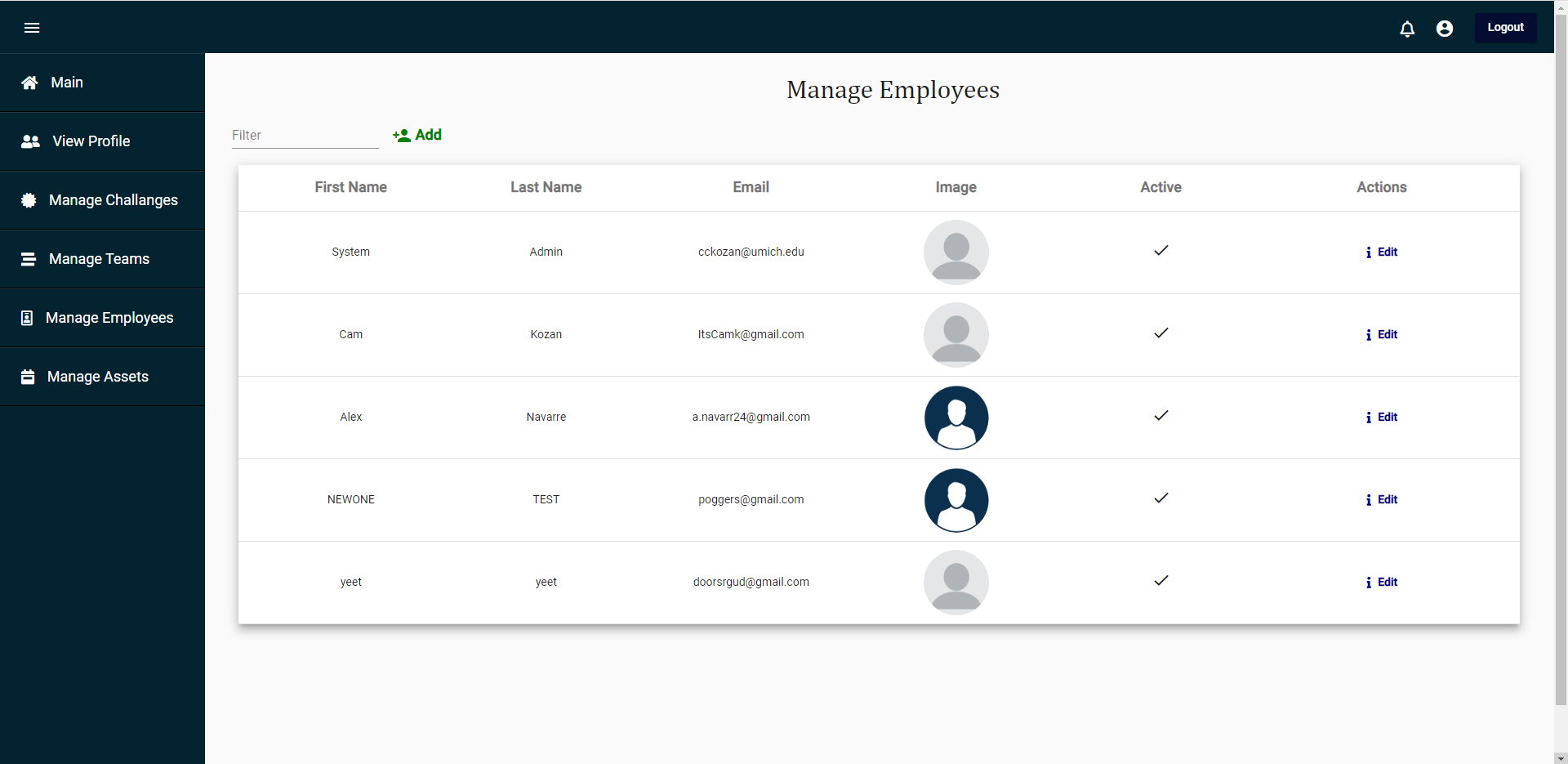
Admin View - Manage Challenge Screen (Reward a User with a Challenge)

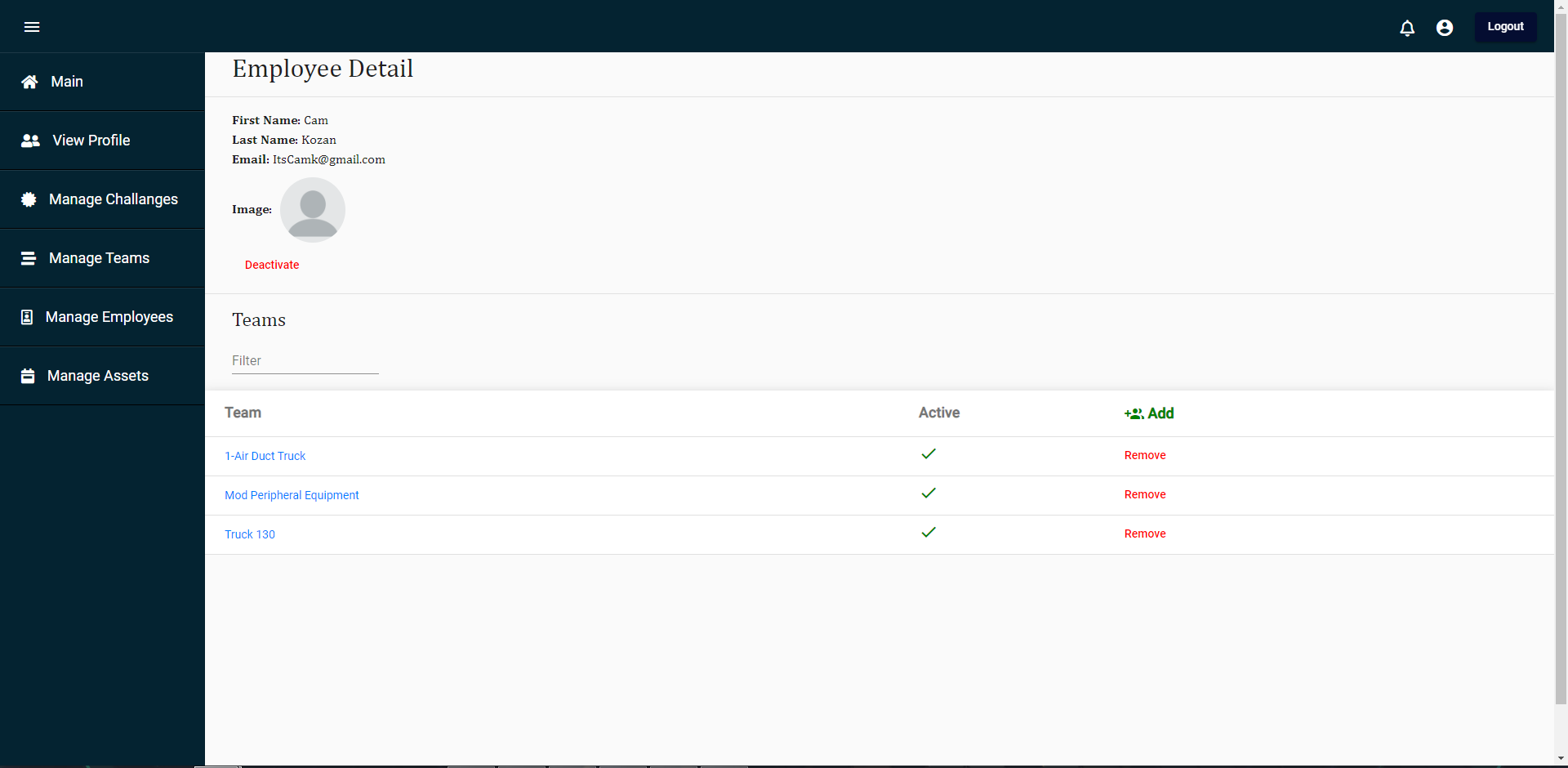


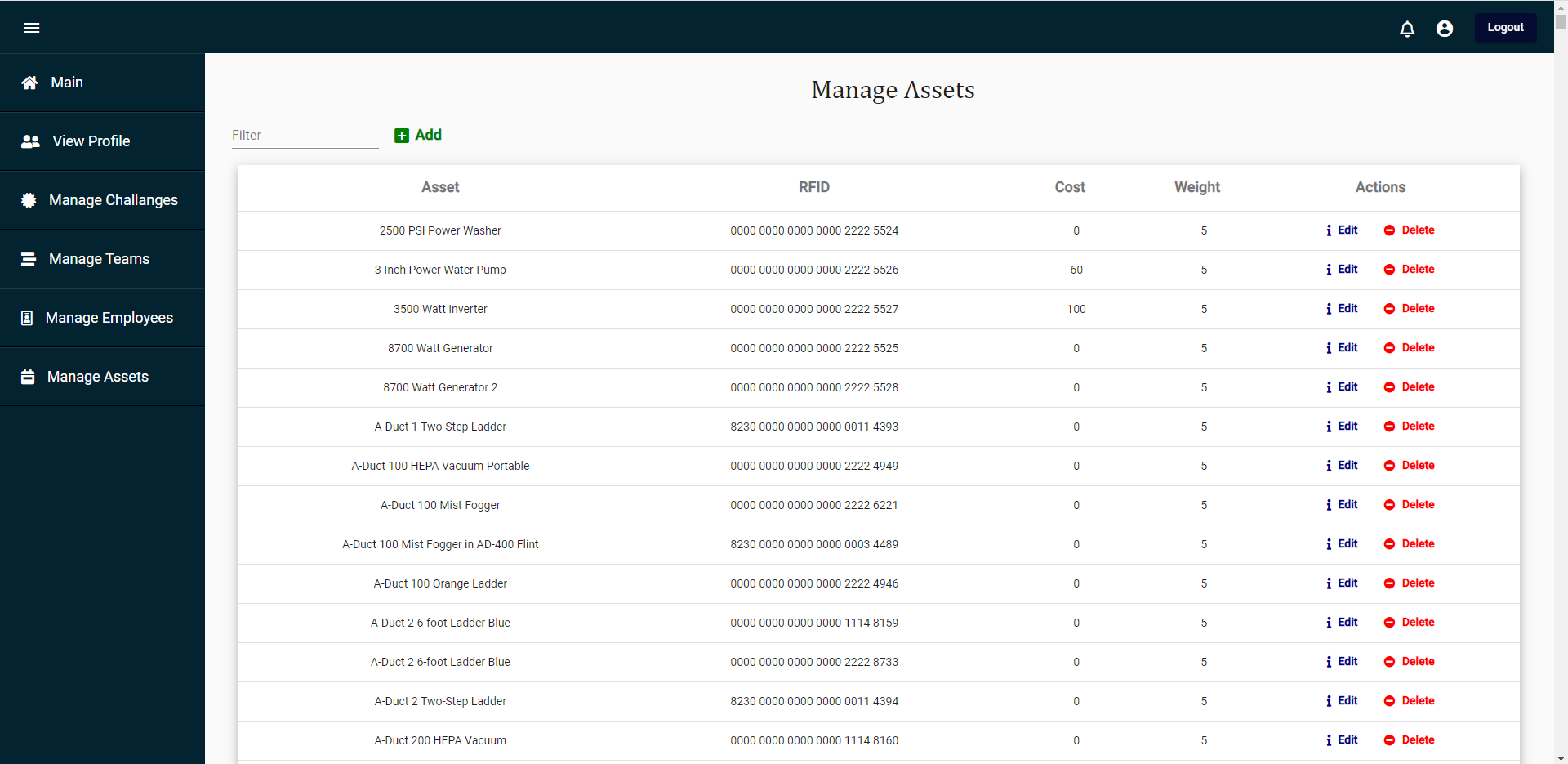
Admin View - Manage Teams Page

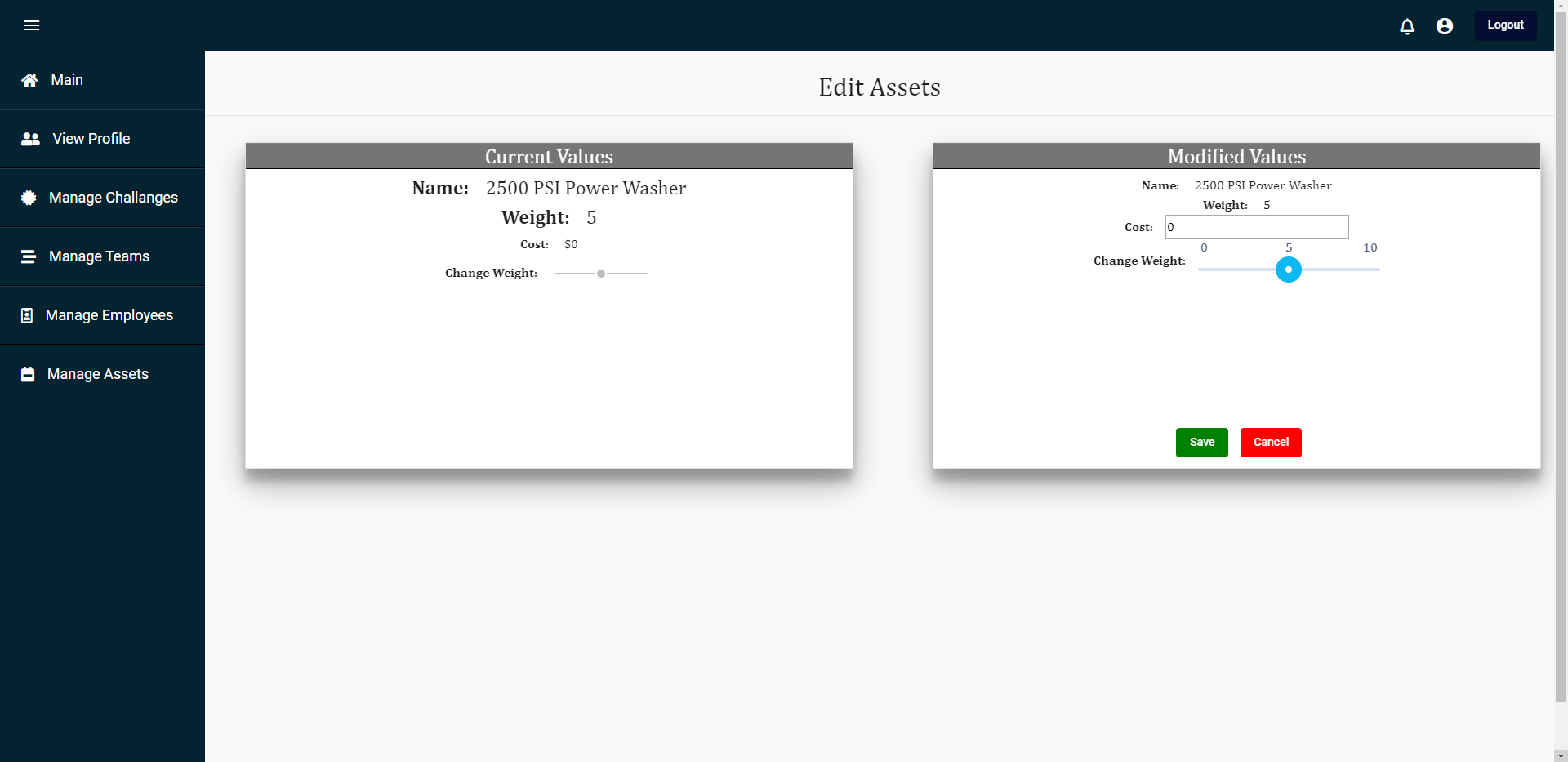
Admin View - Adding an Employee to a Team (Quick Add)

Admin View - Editing a Team

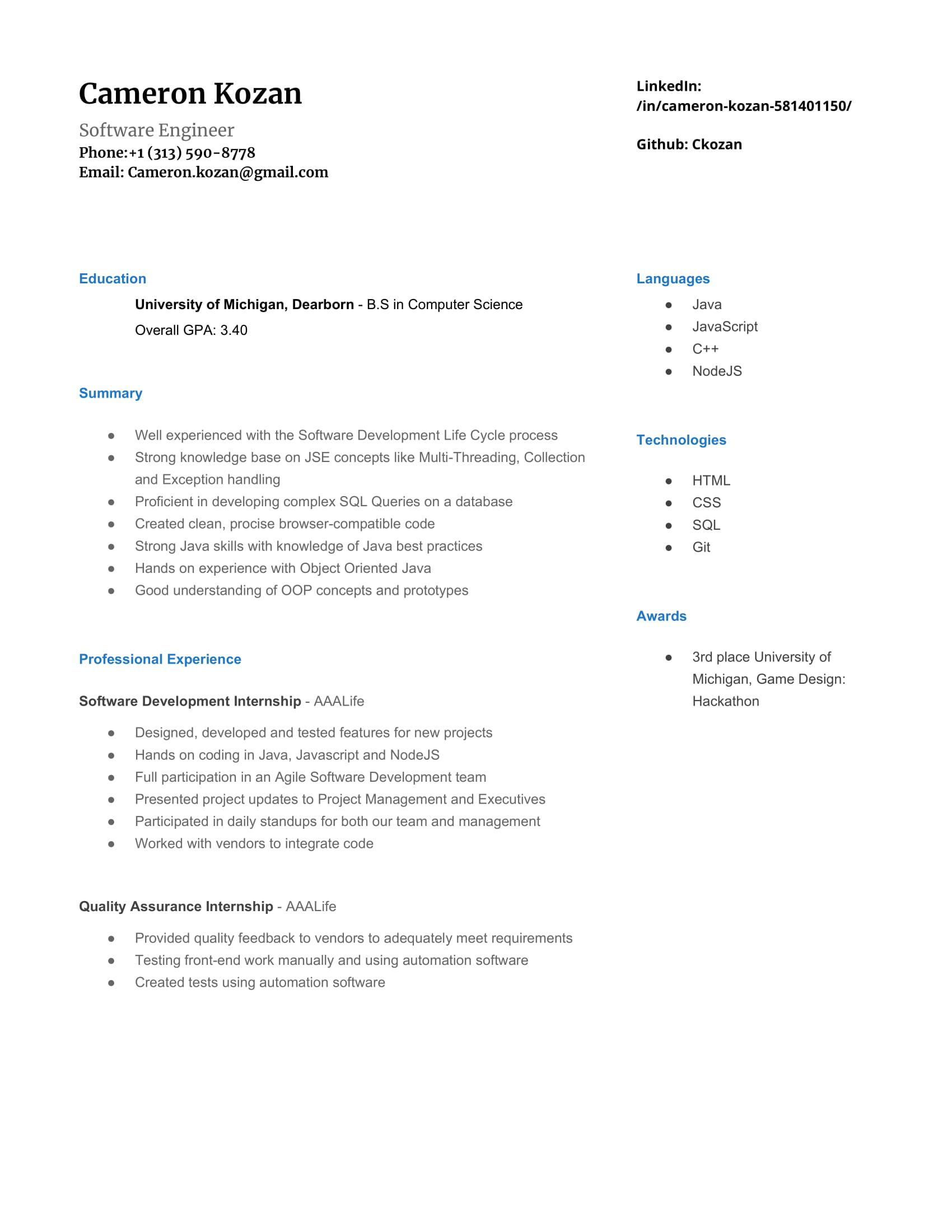
Admin View - Manage Employees

Admin View - Employee Detail Page

Admin View - Manage Assets

Admin View - Edit Asset

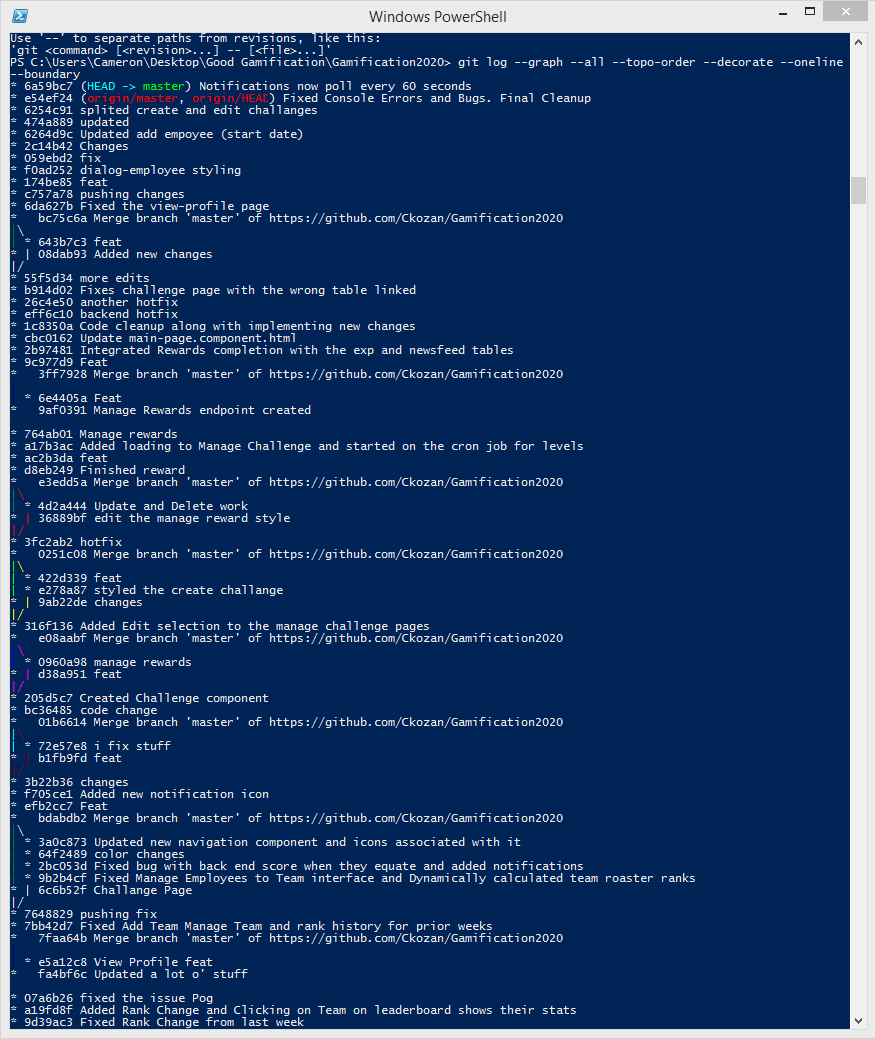
# Appendix C - Team Member Resumes

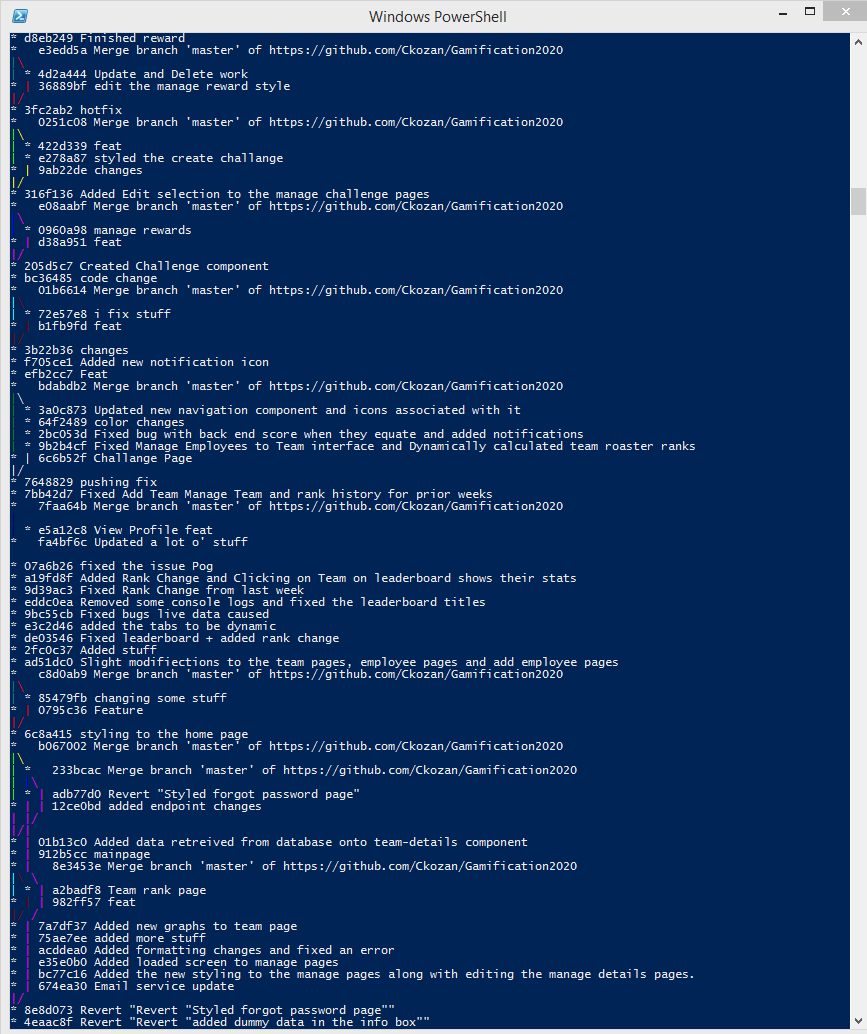




# 

# Appendix D - Project Plan & Log Book





# 

Task Net Diagram

# 

# 

# Appendix E - Project Demo Notes

* Login
* Logout
* Dashboard
* View profile
* Manage teams
* Notifications
* Manage assets
* Manage employees
* Create a challenge
* Reward a challenge
* Delete a challenge

# Appendix F - Final Presentation Slides

Look at attached slideshow