CS 467
Spring 2017

Pyxis: Project Plan Employee Recognition Website

Meet the Team:
Joshua Hesseltine
Siara Leininger
Stephen Smith

Outline & Scope

The team will be developing a web application that will be considered a registered users only awards tool. The tool will allow registered user the ability to generate awards for specified people in the form of an award certificate and will email the award to the email specified. The project will be conducted in an agile development environment. We will conduct weekly sprints with code reviews posted by the end of each sprint. Each team member is required to submit his/her work at the end of each sprint. The weekly code reviews will be conducted via google hangouts or by chat software such as Slack. After the code review a team member will submit the groups progress report in the form of a video recording and submit on behalf of the group.

The following deliverables are required:

1) 5 weekly progress reports submitted in the form of a 2-3 minute video
2) 1 Mid-Point Project Check
3) 1 Final Report
4) 1 Demonstration of Project

Each team member will be responsible for updating the code repository and communicating any issues/roadblocks or constraints during the weekly sprint reviews.

Epic Analysis

The following is a list of 6 core Agile-EPIC descriptions that comprise the core functionality of this webapp.

1) Basic Authentication
2) User Roles/Privileges
3) Administrator Roles/Privileges
4) Awards SQL Database
5) HTTP Request/Response Objects
6) Email Service

Requirements Specification

Requirement Name & ID	Description
1: User Sign-On	1.1: The website will only function when a user is logged in. 1.2: The main page will be the sign on page granting access to the endpoints in the URL 1.3: The login page will accept a username and password 1.4: Login page shall allow for recovery of username and password
2: User Account	**A user account will consist of:** 2.1: An email address (the username). 2.2: A password. 2.3: The name of the user. 2.4: The timestamp of when the account was created. 2.5: The signature of the user, stored as an image file. **A user logged in should be able to also do at least the following:** 2.6: Change his or her name. 2.7: Delete from the system awards that this user has previously given.
3: Admin Account	**An Admin account is deemed a super User.** 3.1: Admin account **cannot** create awards
4: Awards Criteria	4.1: There must be at least two classes of awards - for example, perhaps the Employee of the Month, and the Employee of the Week.
5: Awards CREATE	Once logged in, a user may create a new "award entry" that consists of at least: 5.1: The type of award being given 5.2: The name of the person getting the award. 5.3: The email address of the person getting the award. 5.4: An index into the table of users, indicating which user account created the award.

	5.6: A time and date chosen that allows the user to specify when the award was granted.
6: Awards Email Service	6.1: Once the award has been created, the system should email a PDF certificate to the person getting the award, that has been custom generated using LaTeX, with the person's name, date, and the name of the user who authorized the award, along with the authorizing user's signature image.
7: Admin CREATE, UPDATE(PUT), DELETE	7.1: Add/edit/delete normal users (who can create awards). 7.2: Add/edit/delete admin users (who cannot create awards). 7.3: Perform Business Intelligence reporting operations, for example: querying for specific data (which users created awards, which region had the most awards, etc.). 7.4: Export the above queries as a CSV file and/or display the results on-screen. The on-screen reports should include appropriate graphical charts, such as those provided by Google Charts (Links to an external site.).

Architecture Design and Proposed Software

The Model, View, Controller (MVC) development paradigm and architecture is planned for the web application.

1) Model: MySQL is a Relational DBMS used to store user information and the creation of awards information. Tables will be created to store user information and awards criteria and awards associated with each unique user account.
2) Controller/View Model: The Controller logic will be a Relational State Transfer (REST) API that takes a URL endpoint and performs the CREATE, READ, UPDATE, DELETE (CRUD) operations on the tables in the model. This is currently planned for using NodeJS
3) View: The views will consist of user driven events to sign onto the application, perform queries to the Model and will render the results to the user in the UI, which will likely be Google Chrome. The proposed software will be BootStrap, HTML5 and AngularJS for client side rendering and HTTP method calls.

A hosting service will be used for demonstration purposes: Most likely the OSU School Server.

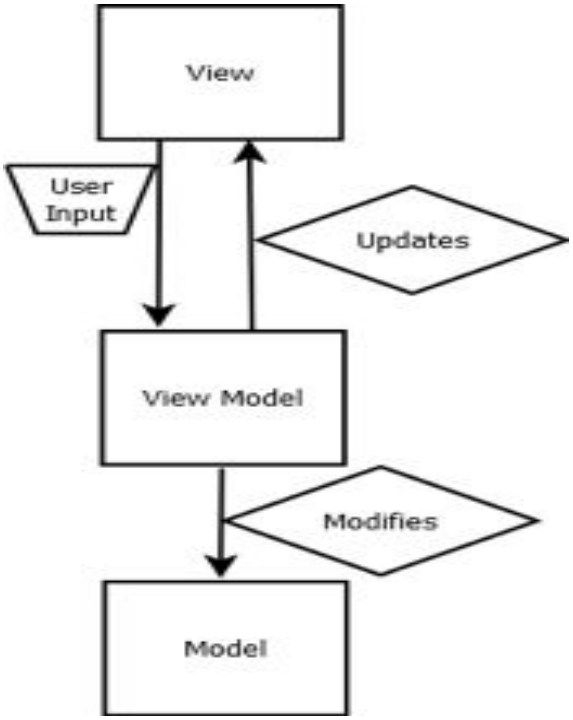
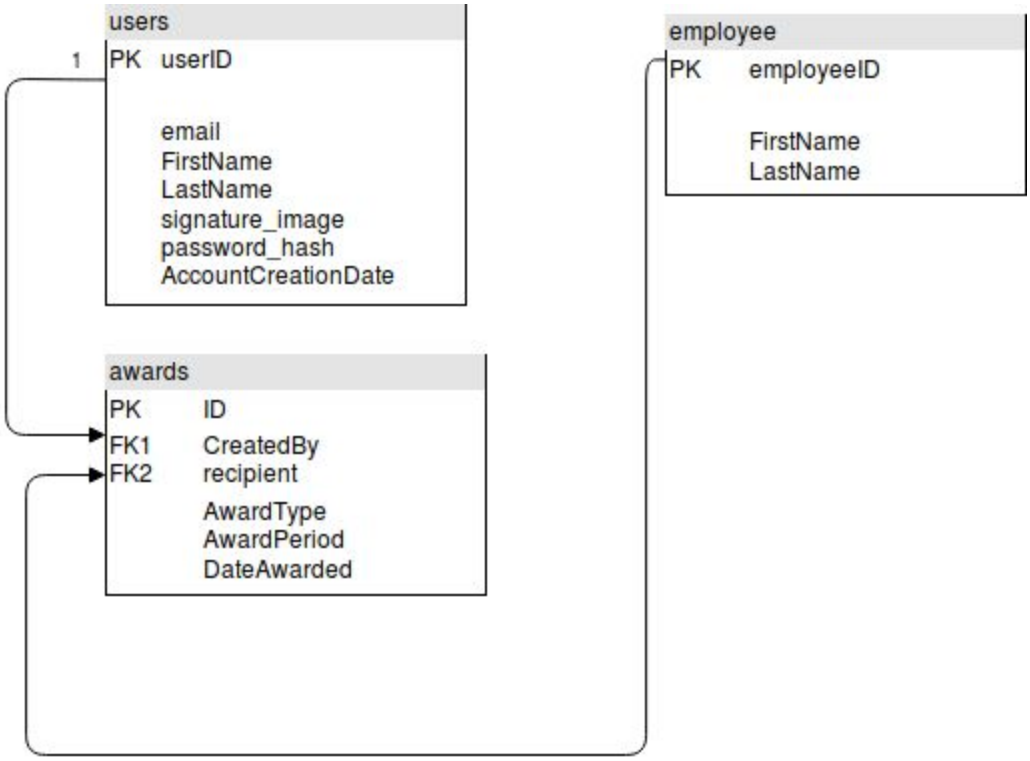
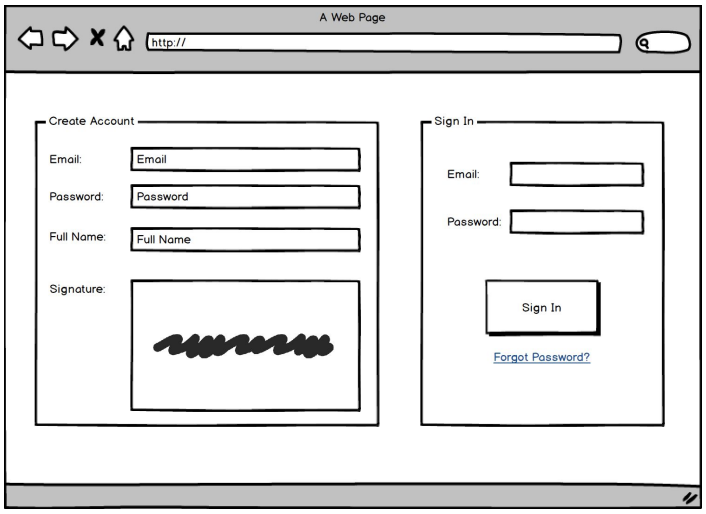


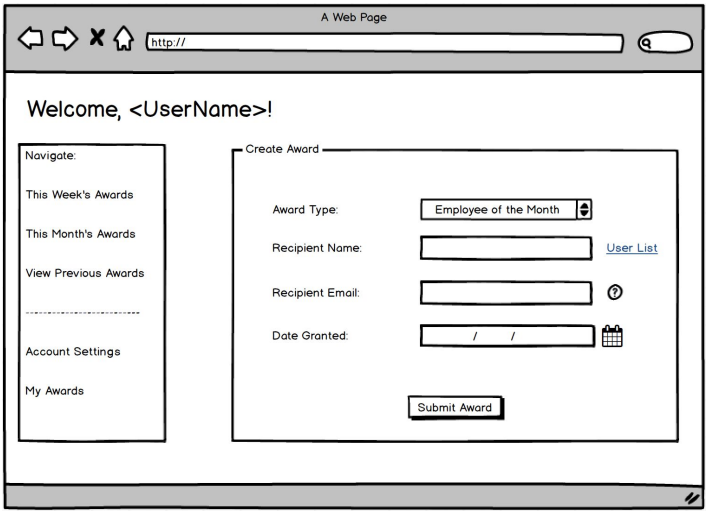
Figure 1: Model, View, Controller Architecture

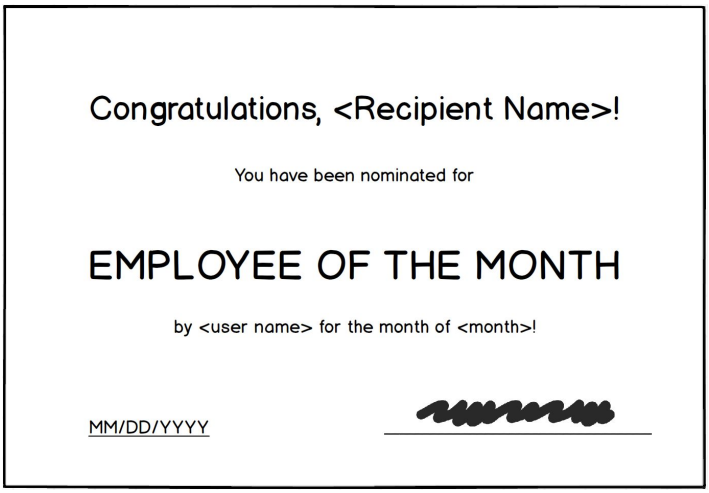
Database UML Diagram:



User Interface Prototype Diagrams







Testing

Each team member will perform unit and functional tests prior to checking in their code when applicable.

Unit Testing: Our Unit testing will be specific to each piece of the architecture, with the exception of View(s) which will be tested during functional and integration testing phases.

REST Endpoint Testing:

1) Authentication
    a) Request object returns a Response object that authenticates user
    b) Request object returns a Response object that allows a user to recover password and/or username
2) User Driven
    a) Request object to create an award
    b) Request object to update user information
    c) Request object to delete a user award
3) Awards Creation
    a) Request object to update an award
4) Admin Account
    a) Request object to create new users, update users, remove users
    b) Request object to query database for user information

Database Testing:

1) Create Award
2) Delete Award

3) Create User Account
4) Delete User Account
5) Create User information table
6) Delete User Information table
7) Update User Account

Server Email Service Testing:

1) Email award to email address from request object
2) Validate creation of signature from user

Functional Testing

1) Verify a user can login to app
2) Verify an admin can login to app
3) Verify a user can recover password
4) Verify only an authenticated user can access the site
5) Verify a user can create, delete awards
6) Verify a user can create a signature
7) Verify a user can change their display name
8) Verify an admin user can add/edit/delete users
9) Verify an admin and produce reports on users and awards stored in database
10) Verify an admin can export reports to csv or show reports on screen

Integration Testing

This is the final testing phase. After the end of Sprint 7, all developers will commit a final checking of their code for peer review and will build the entire app and run through all testing necessary to validate requirements.

Code Tasking Plan

Sprint 1 (Week 3)	**Josh**	**Siara**	**Stephen**
Est. Hours	10	10	10
Requirement ID/Name	Controller Req 1 and 2	View Req 1 and 2	Model Req 1 and 2
Sprint 2			

Est. Hours	10	10	10
Requirement ID/Name	Controller Req 3 and 7	View Req 3 and 7	Model Req 3 and 7
Sprint 3			
Est. Hours	10	10	10
Requirement ID/Name	Controller Req 3 and 7	View Req 3 and 7	Model Req 3 and 7
Sprint 4			
Est. Hours	10	10	10
Requirement ID/Name	Controller Req 4 and 5	View Req 4 and 5	Model Req 4 and 5
Sprint 5			
Est. Hours	10	10	10
Requirement ID/Name	Controller Req 4 and 5	View Req 4 and 5	Model Req 4 and 5
Sprint 6			
Est. Hours	10	10	10
Requirement ID/Name	Controller Req 6	View Req 6	Model Req 6
Sprint 7			
Est. Hours	10	10	10
Requirement ID/Name	Controller Req 6	View Req 6	Model Req 6
Integration Testing	All	All	All

Client Reporting Deliverable Team Responsibilities

Deliverable	Due Date	Format	Assignee
Project Plan	14 April 2017	PDF	All
Week 3 Progress Report	23 April 2017	Video	Siara
Week 4 Progress Report	30 April 2017	Video	Stephen
Week 5 Progress Report	7 May 2017	Video	Josh
Mid-Point Check	12 May 2017	All files + PDF	All 1 member submits
Week 7 Progress Report	21 May 2017	Video	Siara
Week 8 Progress Report	28 May 2017	Video	Stephen
Final Report	09 June 2017	PDF + Installation/Run Instructions	All
Demonstrate Project	09 June 2017	All files in zip file	All

Conclusion:
Team Pyxis will create an Employee Recognition web application where users can create awards and nominate their co-workers to receive them. We will be using the OSU school server to host the app. Our project will take at least 300 hours to complete, and we will do our best to maintain the schedule outlined in this proposal.

References:

UI:
https://github.com/angular
http://getbootstrap.com/

Controller:
https://nodejs.org/en/download/

Model:
https://www.mysql.com/downloads/