SOFTWARE REQUIREMENTS OUTLINE

for

Crew Scores Dashboard

Version 0.1

Prepared by Team 3

J. Steward, B. Olah, D. Rife, Z. Thompson, R. Hess

Contents

1 Introduction **3**
- 1.1 Purpose . . . 3
- 1.2 Project Scope and Product Features . . . 3

2 Overall Description **4**
- 2.1 Basic Requirements . . . 4

3 Database **5**
- 3.1 Schema . . . 5
  - 3.1.1 ER Diagram . . . 6
- 3.2 Translation . . . 6

4 User Interface **7**
- 4.1 Design . . . 7
- 4.2 Pages . . . 7
  - 4.2.1 Pageflow Diagram . . . 8
- 4.3 Wireframes . . . 9

5 Task Division **13**
- 5.1 Tasks . . . 13

1 Introduction

1.1 Purpose

This project has two purposes: a) to fulfill the requirements for the final group project in our web programming course, and b) to create a scoreboard for the crew competition in the CS department at Shippensburg University which will be useful going forward.

1.2 Project Scope and Product Features

Due to limited time, this project will have a narrow scope. In essence, it must track the points scored by each of the three teams on various events, track the events and what happened during them, provide a public interface where students may view the current point total, and allow faculty to update the stored information.

Although most of the faculty probably are comfortable manipulating mysql databases from the command line, it is not the smoothest user experience. Because of this an interface must be provided which will allow the faculty to update the points and events without leaving the comfort of their web browsers.

Because allowing public access to the stored data will quickly devolve into chaos, an access control system must be in place over the interface for updating points. This access control system must not be a burden, and should function like the user accounts would on any other site.

Finally, to meet the requirement of 5 tables in the database, a system will be provided which allows anyone to subscribe to email notifications which will be sent when events and points are updated. To prevent mischief, this system must comply with the CAN-SPAM act, and verify ownership of the email address entered before sending any further email notifications.

2 Overall Description

2.1 Basic Requirements

1. Track the points each crew earns from various events. This means tracking:
   - Teams (Team Name, Color, Logo)
   - Points (Points Earned by each Team at each Event)
   - Events (Event Name, Description, Date, Points)
2. New events should only be enter-able by faculty. This means a user system will be needed.
   - Track each user's info (Email, Password (hashed salted))
   - Existing users should be able to add/remove other users
   - Existing users should not be able to remove themselves
   - Existing users should be able to change their password
   - Once logged in, users should be able to add/remove/edit events, add/remove/edit teams, and add/remove/edit the number of points a team earned
3. Weekly email updates should be automatically sent out to anyone who signs up.
   - This must comply with the CAN-SPAM act. This means:
     - A user may unsubscribe via a single click of a single link in any email sent
     - A confirmation email must be sent and followed before any other emails will be sent to that user
4. The main page of the website should list each team, along with their current score totals
5. There should be a place to view a list of all of the events, sorted as desired by the user

3 Database

3.1 Schema

The database will need the following tables:

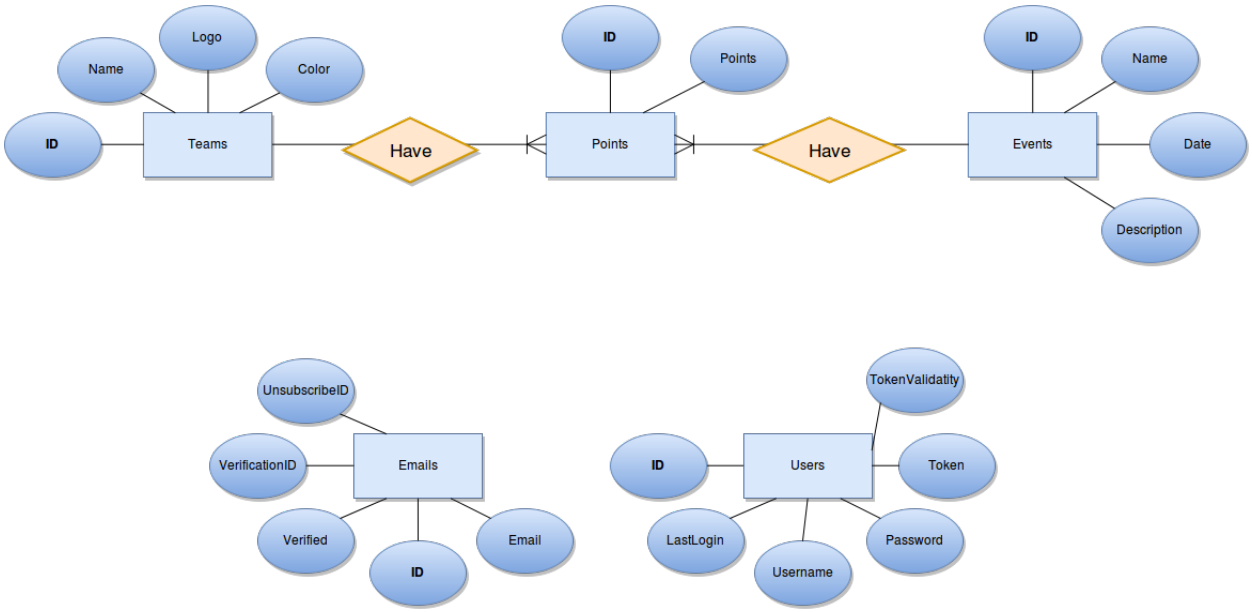
- Teams - Stores all the teams
  - ID: Integer, Primary Key - A unique Id
  - Name: VARCHAR(n) - The team name
  - Logo - The team logo, stored as a file path or a blob
  - Color: VARCHAR(6) - The HTML Color code of the team color
- Points - Linking table between teams and events, with added data (points)
  - ID: Integer, Primary Key - A unique Id
  - Points: Integer - Points earned by Team in Event
  - Event: FK(Events)
  - Team: FK(Teams)
- Events - Stores all the events
  - ID: Integer, Primary Key - A unique Id
  - Name: VARCHAR(n) - The short event title
  - Date: Date - The day the event happened, or will happen
  - Description: VARCHAR(n) - A description of the event and the outcome
- Users - Stores all of the user logons
  - ID: Integer, Primary Key - A unique Id
  - Username: VARCHAR(n) - The username
  - Password: VARCHAR(n) - The SALTED HASHED password. Yes it needs to be salted and hashed.
  - Token: VARCHAR(n) - A unique token used to track logins
  - TokenValidity: Date - The date the token was issued
  - LastLogin: Date - Date of the last time a user logged in
- Emails - Contains all of the email subscriptions
  - ID: Integer, Primary Key - A unique Id
  - Email: VARCHAR(n) - The email address
  - Verified: Boolean - Whether the email has been successfully validated or not

– UnsubscribeId: Integer, Unique - A unique ID (possibly a GUID) used to unsubscribe from email notifications

– SubscribeId: Integer, Unique - A unique ID used to verify email subscriptions

3.1.1 ER Diagram

Figure 3.1: ER Diagram



3.2 Translation

A translation layer should be written to sit above the database and provide a nice, abstract set of methods and objects to manipulate the database.

4 User Interface

4.1 Design

Overall, the project will follow a material design aesthetic.

Every page has the same header and footer.

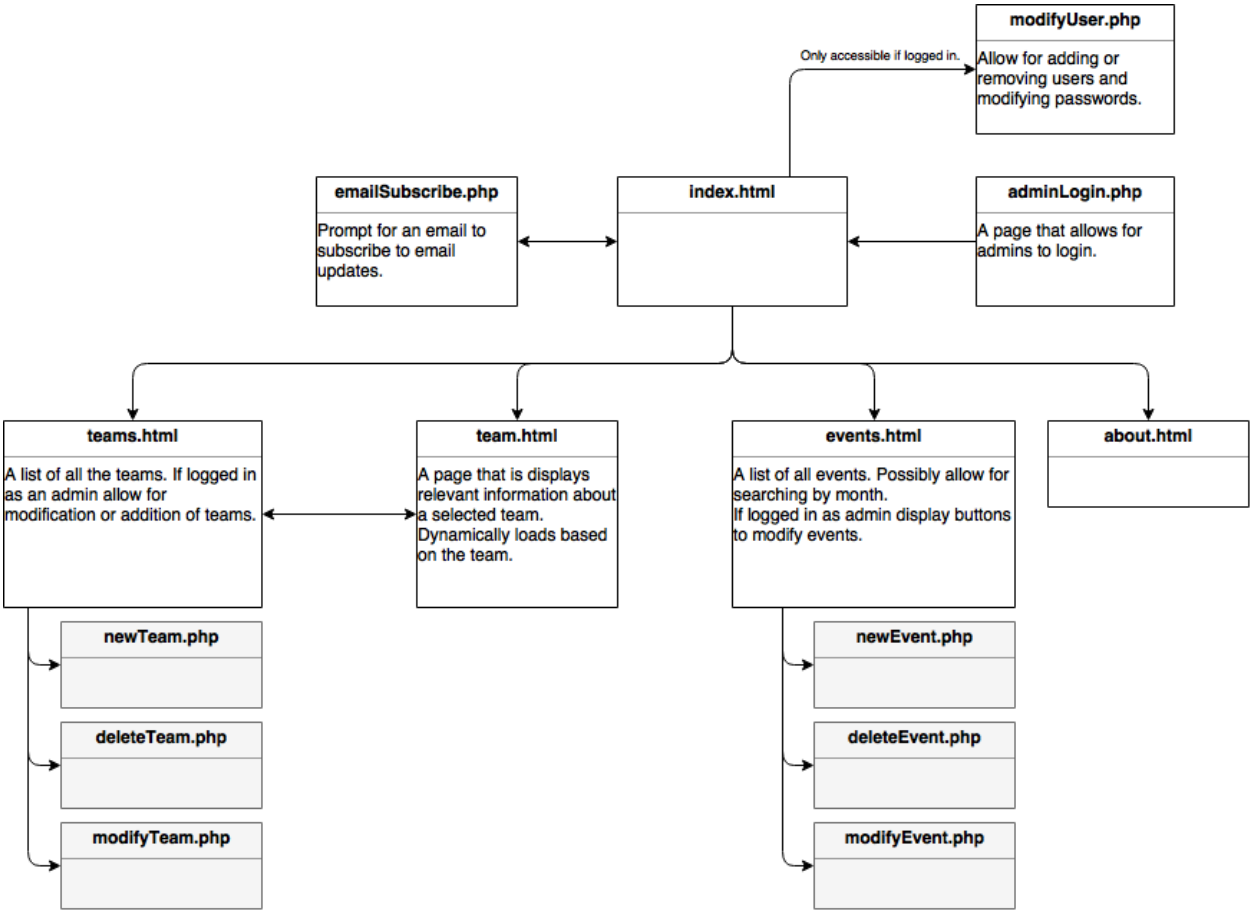
4.2 Pages

If the user is a professor and is logged in, a floating button will be at the bottom corner of the home page that connects to event creation page.

- Home: Shows overall score for each team, shows cards with recent news/events, upcoming events, teams, and point values.
- Header: connects to login page, user creation, subscribe, and home page.
- Footer: connects to about page, possibly more?
- News: Shows info about the event and connects to edit points page.
- Team: Shows info about the team, such as their score and team leader.
- Subscribe: Allows the user to enter an email address to receive updates/news.

4.2.1 Pageflow Diagram

Figure 4.1: Pageflow Diagram



4.3 Wireframes

The following wireframes are a rough mockup of most of the main, user-facing pages.

Figure 4.2: Homepage while signed out

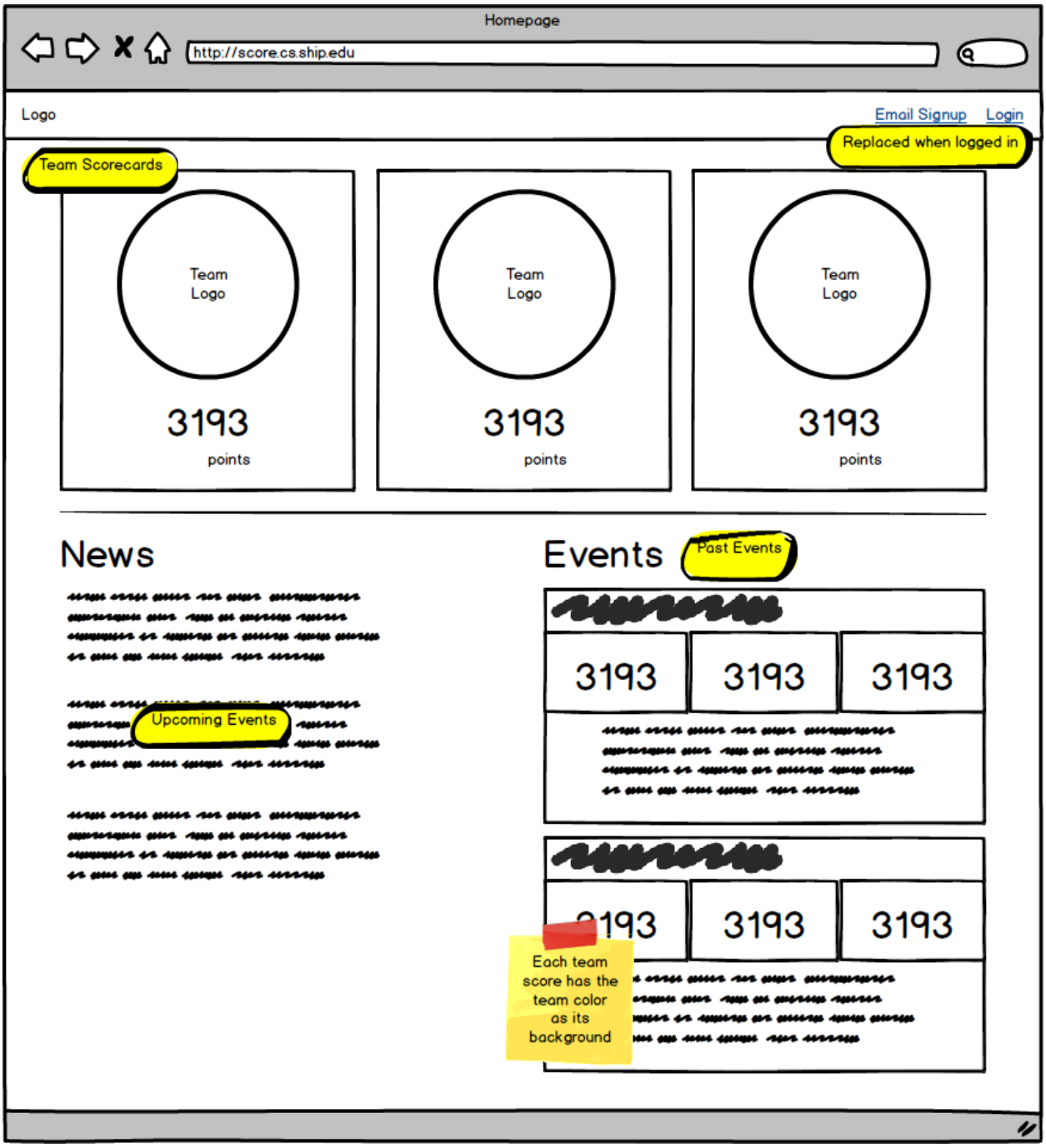


Figure 4.3: Homepage while signed in

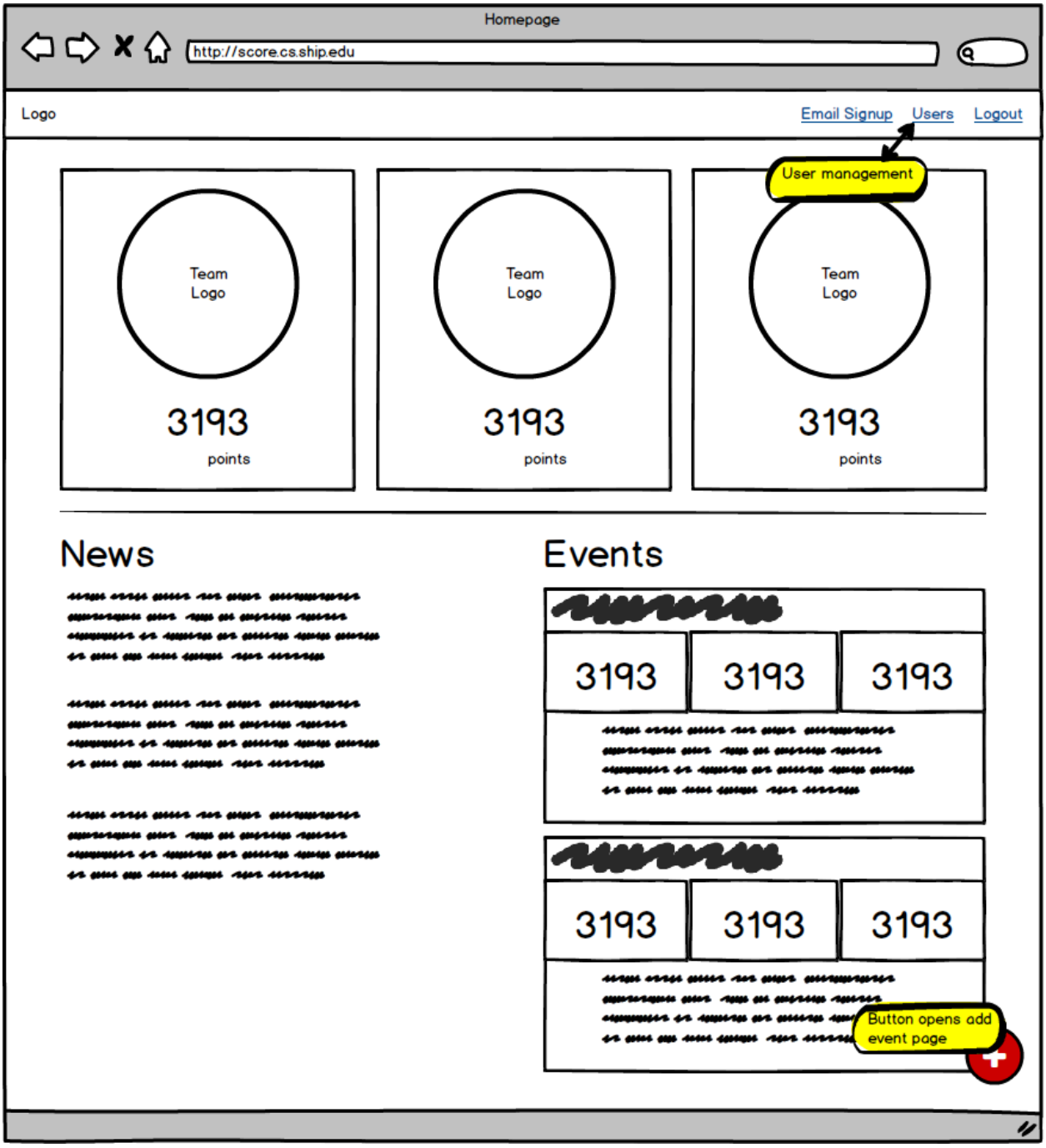


Figure 4.4: After clicking the '+' button while logged in

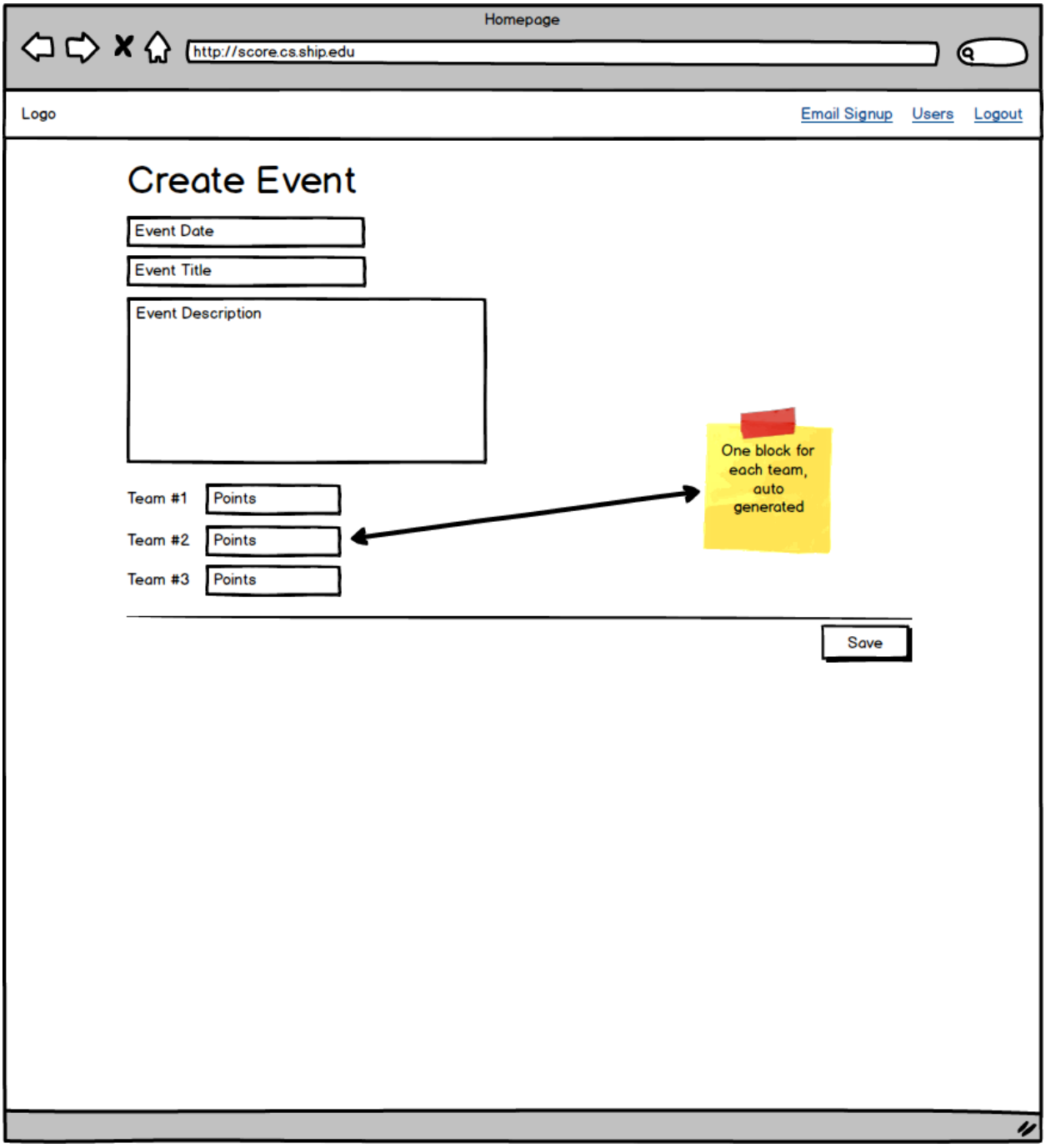
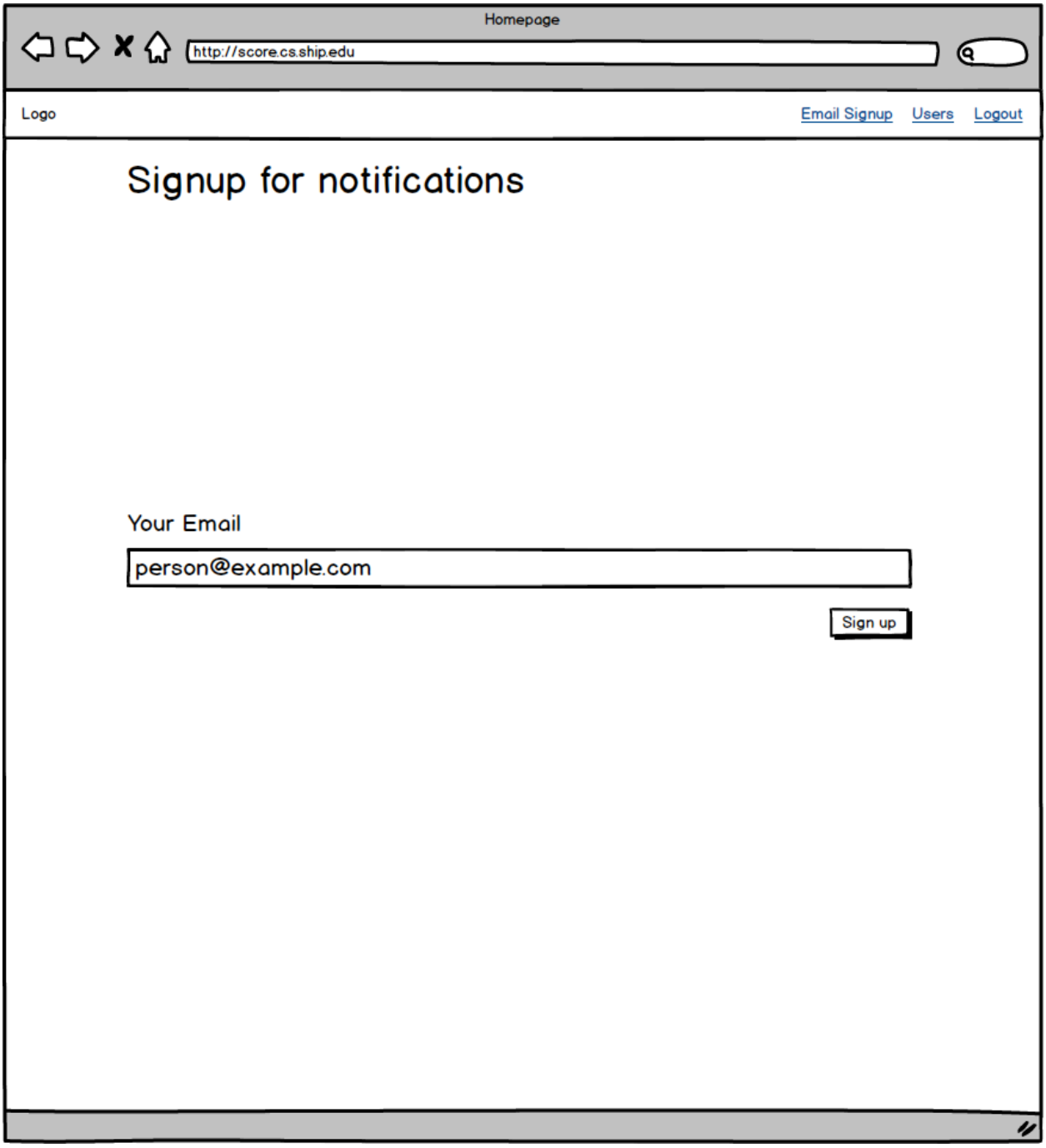


Figure 4.5: Subscribe to email notifications



5 Task Division

5.1 Tasks

- Creating the user system
- CSS
- Needs to be a way to send out emails
- Need a DatabaseGateway
- Set up the database
- Config file for email server (SMTP server) database connection string
- Framework for every page
- All the pages (add, update, delete)
  - Points (add, update, delete)
  - Teams (add, update, delete) -
    - Upload a logo
    - Team name
    - Team color
    - Events
    - Event names
    - Description
    - Date
    - Points
- Email subscription system
- Main Page