HSPC Specification Document

# Introduction

The Computer Science Department at Kansas State University hosts an annual High School Programming Contest (HSPC) in which teams of high school students across Kansas come to compete in a programming contest. Currently, the contest is run on paper and all of the sign-ups are manually sent out. High School advisors from a compiled list of schools are emailed google forms to sign up teams from their school for the contest and an advisor at Kansas State manually reviews the sign-ups and manually signs the teams up for the contest. This long and tedious process requires a lot of work and effort from many different people. The following document provides an overview and specifications of a web-application whose purpose is to solve the issues presented above and provide a platform for future advancements in the programming contest.

# Project Overview

The primary goal of this project is to develop a web-based app to modernize and simplify the management of the High School Programming Contest at Kansas State University. This application will serve various stakeholders, including high school advisors, contest volunteers, judges, organizers, and IT administrators. It will address the problem of inconvenient manual processes by providing a digital platform for registration, information distribution, and event management. Key features of this system include user-specific interfaces, live scoreboards, digital team management, and streamlined communication. Design decisions are influenced by the need for ease of use, scalability, and secure data management.

# Development and Target Environments

The project is built using React Javascript to provide an interactive front-end web application to the user through their web browser. The benefits of developing a web application means that the only hardware requirements for our project are having a computer with access to the internet. From a developer stand-point all of our code is developed using Visual Studio Code through code environments hosted on a web-server. Choosing to code online means that we can develop our project from anywhere as long as we have a computer with access to the internet. Unfortunately, the project does not currently support mobile web applications, but it is a goal of ours to allow for the application to function on a mobile phone. If finished, our project will be hosted using K-State’s dedicated web servers.

An important note about the coding practices of the React Javascript project is that it uses a class structure instead of a functional structure. And the application state is unfortunately, in its current state, not handled neatly or consistently. While there is some top level data that utilizes the Redux package to store data globally, it is not consistently used and in most components, data is loaded in via http requests upon opening each page instead of being stored at the top level. The plan is certainly to refactor for the future by either using redux or at least moving the data to the state of the top component. A decision has yet to have been made on which route to focus on.

# System Model

The system context is displayed below in Figure 1. It serves as a visual for a quick understanding of the software system.

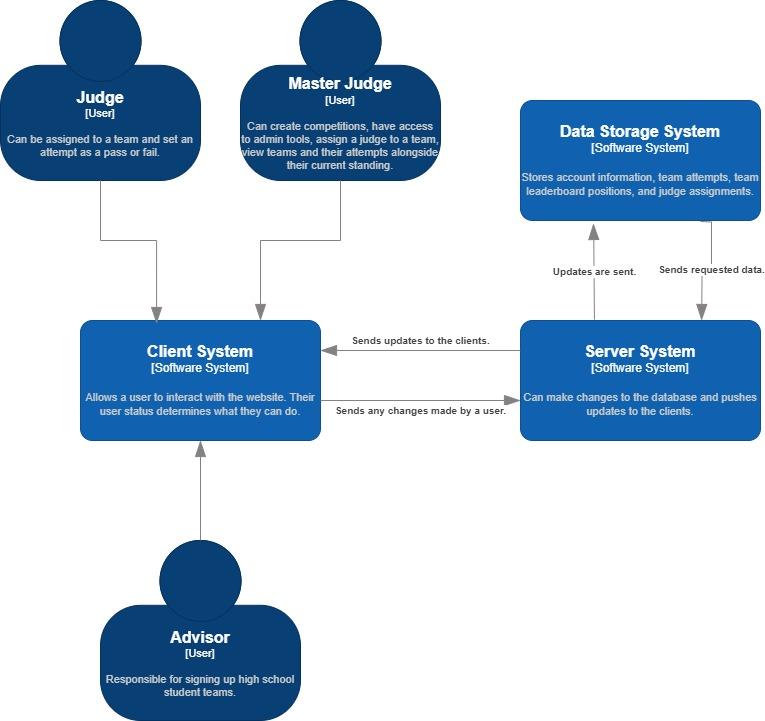


Figure 1: Context Diagram

The container diagram is displayed below in Figure 2. It shows the distribution of responsibilities across the system and some of the technologies used in it.

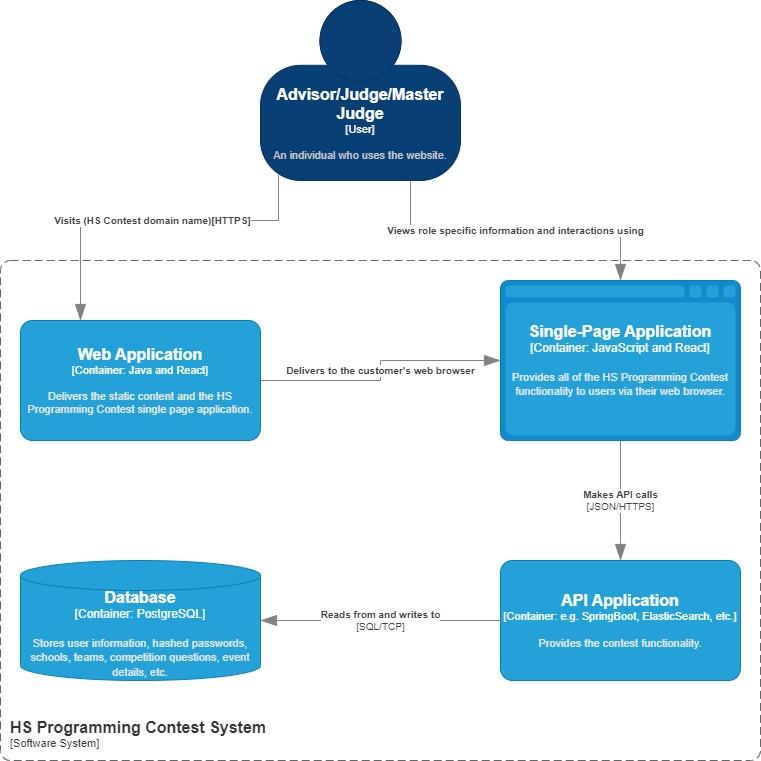


Figure 2: Containers Diagram

# User Interaction

Our application serves several purposes for several different users. The users of our website fall into certain categories which are a generic user, a high school advisor, a contest volunteer/judge, a contest organizer, and an IT master user. Below are all of the user interactions based on the category of the person using it.

A generic user could possibly be a random internet traveler, a parent wanting more information, or a high school student participating. A generic user accessing the website can…

* Access a home page displaying all relevant event information.
* Access the problems and solutions from previous events
* Access scoreboards from past events.
* Access a log-in/registration page if they desire a larger role in the contest.

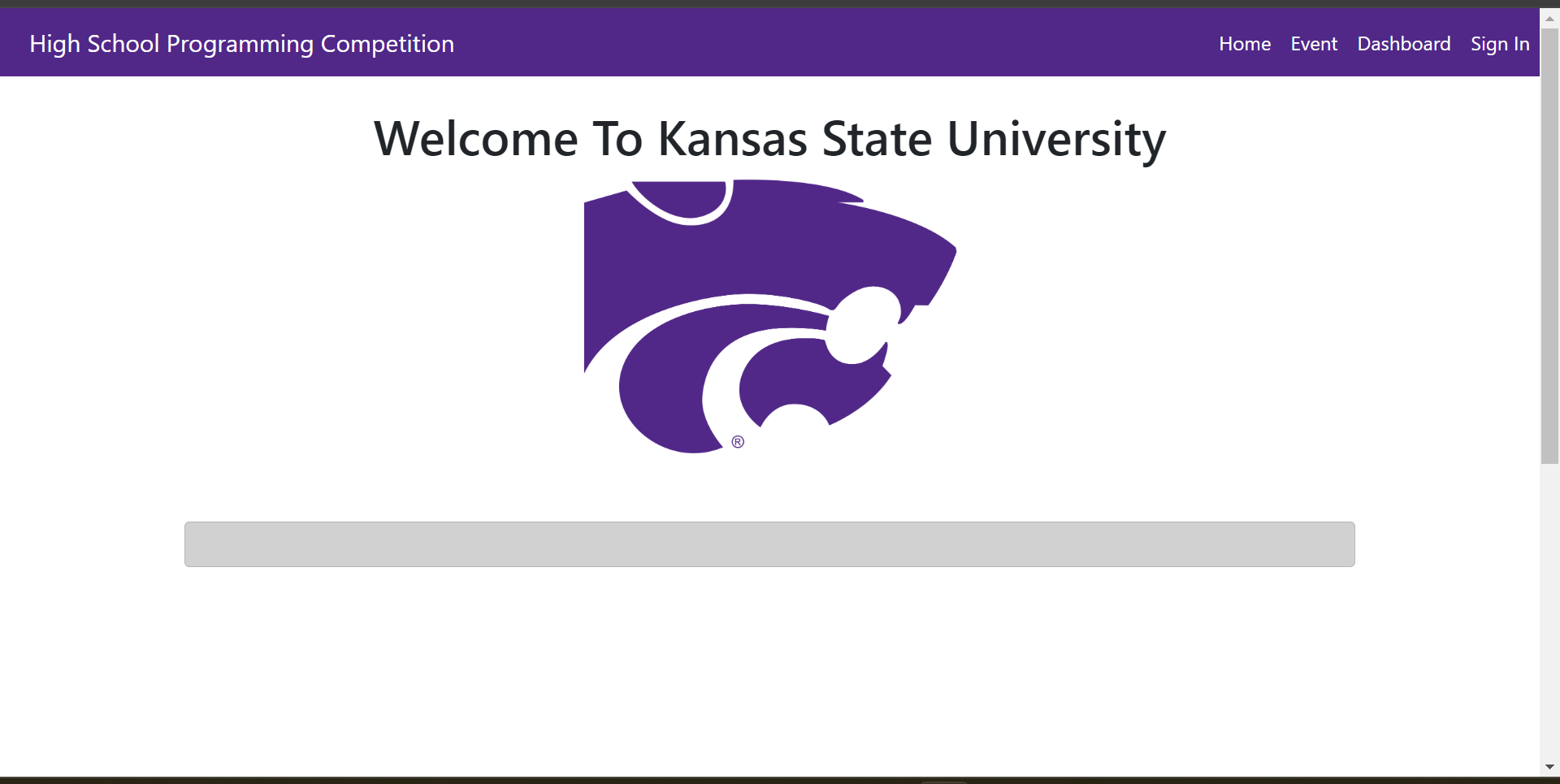


Figure 3: Generic User Screen

Including the use cases of the generic user a high school advisor can…

* Create teams and students associated with their school for the purposes of signing up teams for the events.
* View a live scoreboard of the event.
* View timer of rounds during active event

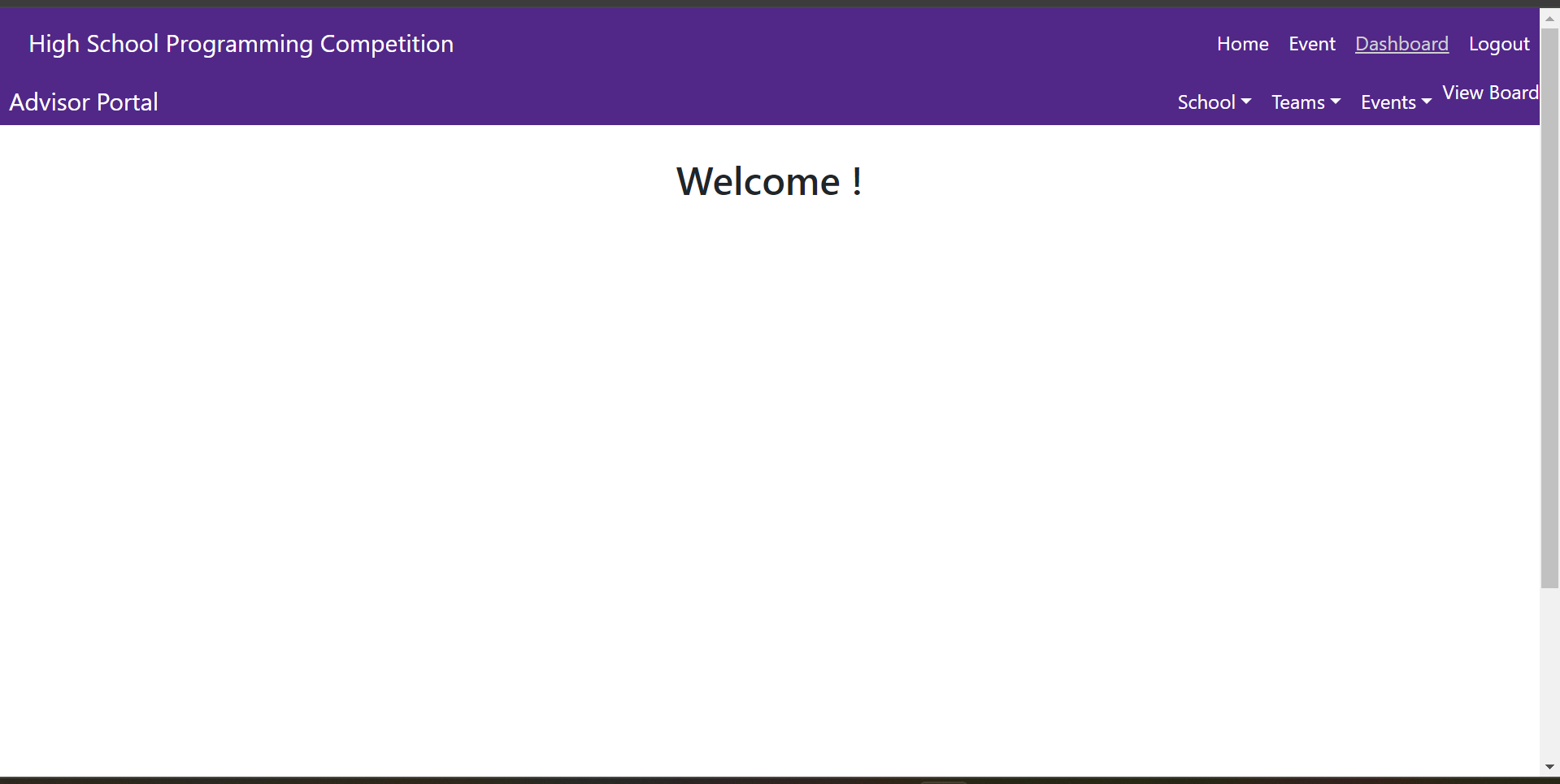


Figure 4: Advisor Screen

Including the use cases of the generic user, a contest volunteer can…

* Sign up to volunteer for an event
* Gain further insight on their volunteer position (if assigned to volunteer).
* View timer of rounds during active event

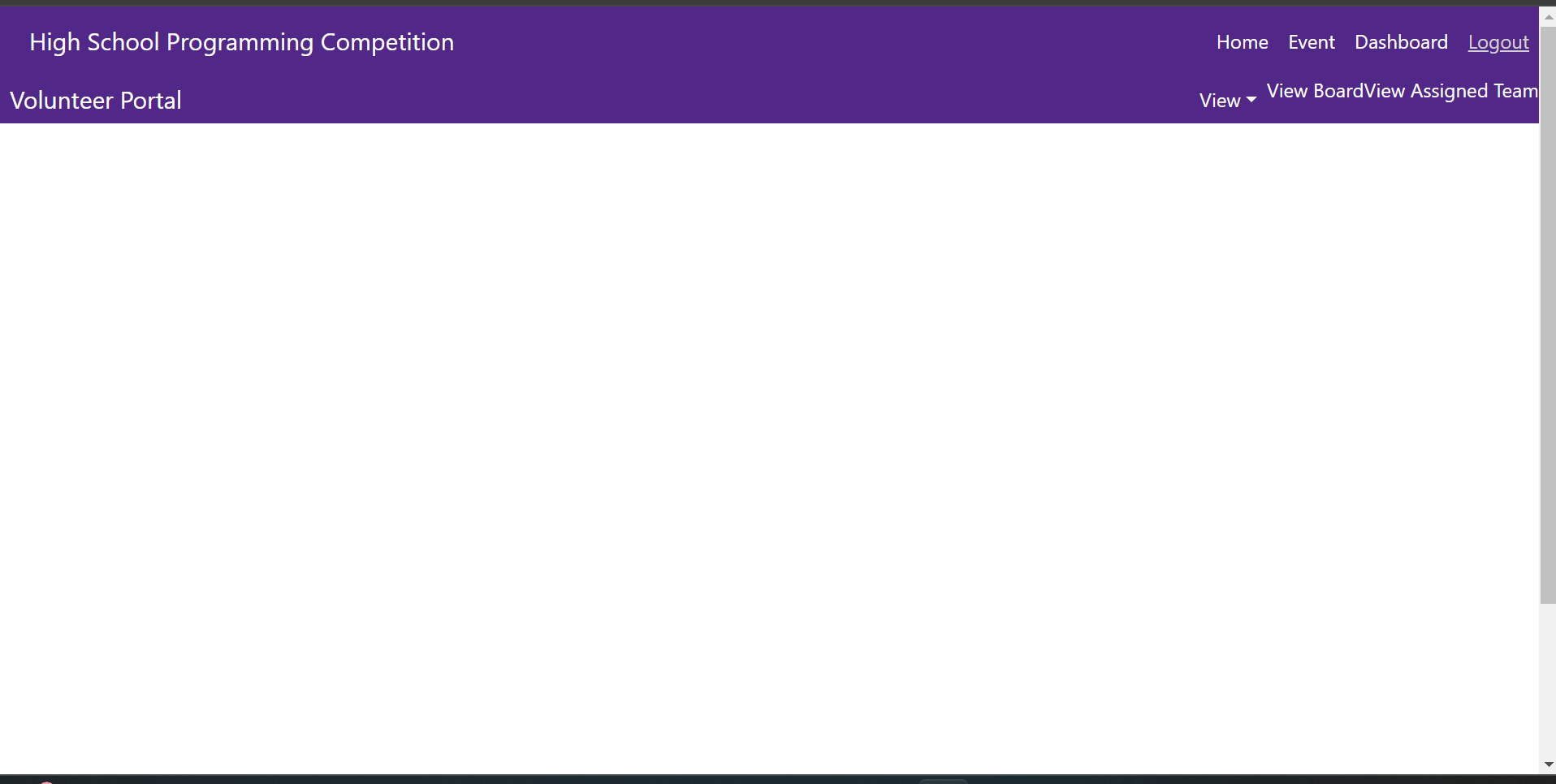


Figure 5: Volunteer Screen

Including the use cases of the contest volunteer, a contest judge can…

* Assign themselves to judge a team during the contest.
* Judge a team during an active contest.
* View teams for an active contest.

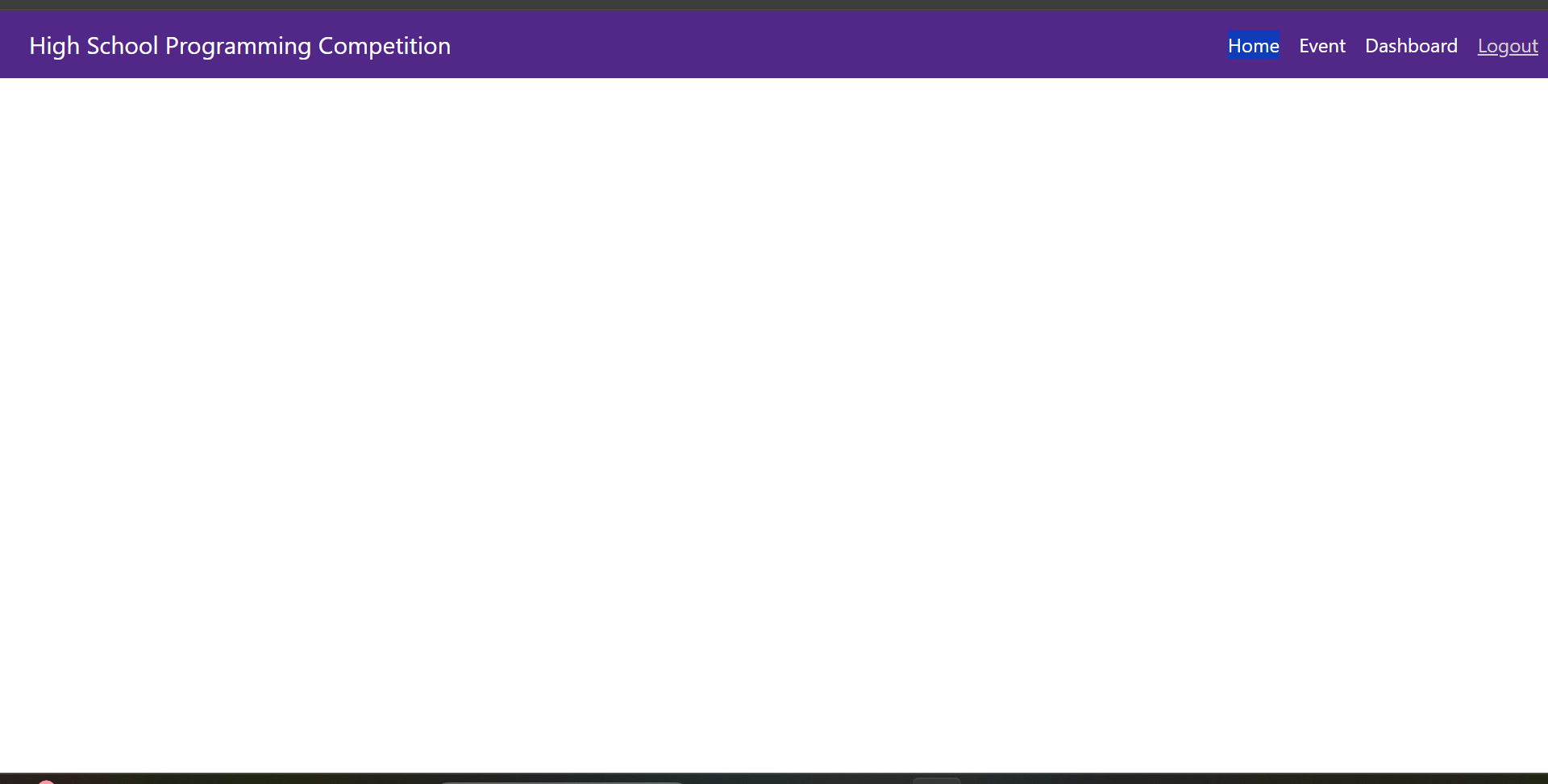


Figure 6: Judge Screen

Including the use cases of a generic user, a contest organizer (also known as a master judge) can…

* Approve/deny advisor accounts.
* Approve/deny team registrations.
* Approve/deny volunteer accounts.
* Assign volunteers to a contest.
* Approve new high schools to compete.
* View all volunteers.
* View all teams.
* View all advisors.
* View all schools.
* View all previous contests.
* Control active contests and active rounds.
* Ability to judge teams during the contest (If they so choose).

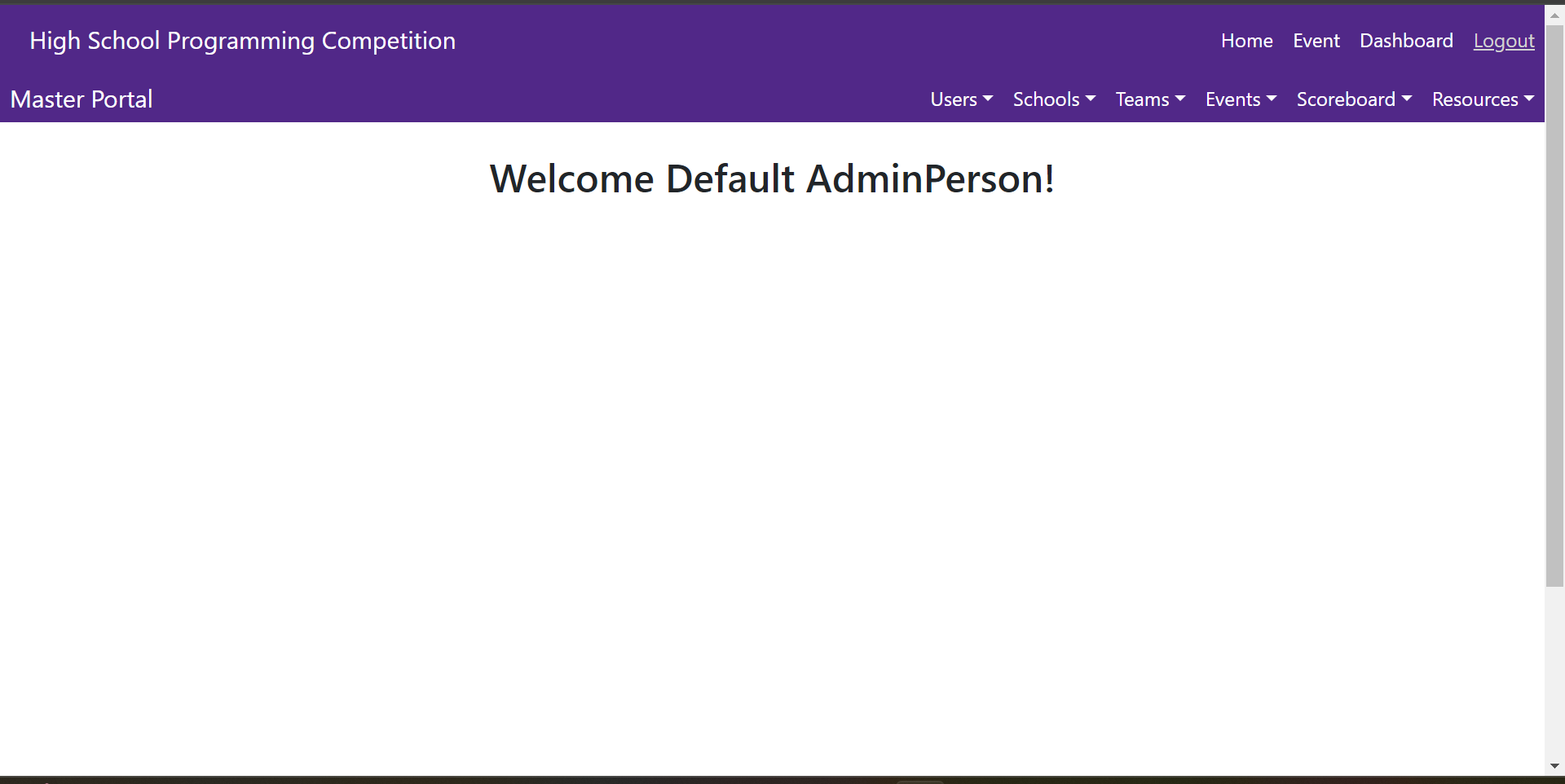


Figure 7: Master Screen

Including all of the previous use cases of all roles, a master account (that may be used by IT) can…

* Manipulate all data from all tables in the database.

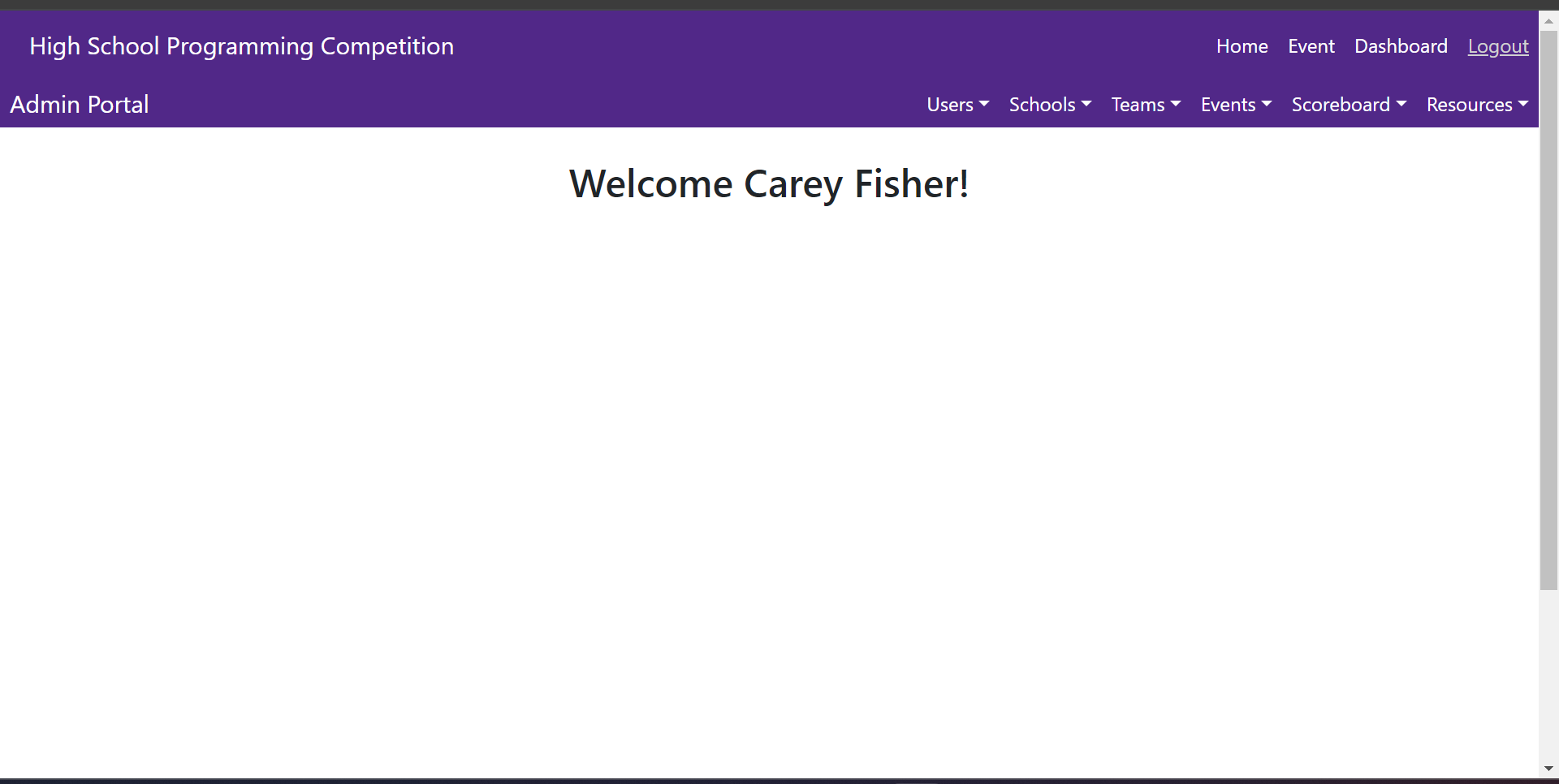


Figure 8: Admin Screen

# Functional Requirements

1. The application shall be web-based for global access for all stakeholders
2. The application shall provide a home page with easily accessible information to users about the event. The home page shall provide…
   1. What the event is
   2. When the event is
   3. Where the event is
   4. Details of the event
   5. How people can sign up
3. The application shall provide details of previous contests’ problems and solutions and the contests’ scoreboards.
4. The application shall contain a login/registration page for contest organizers, volunteers, and high school advisors.
   1. The registration page shall require a first name, last name, email, password, and phone number.
   2. The registration page shall require an associated school for advisor account registrations.
5. Upon login additional options shall become available based on the account type
   1. Volunteer accounts shall have the following additional controls:
      1. A volunteer shall be able to apply to volunteer for an upcoming high school programming contest.
   2. Advisor accounts shall have the following additional controls:
      1. An advisor shall be able to create students in the application to bring to a contest.
      2. An advisor shall be able to create teams and assign students to them to register and bring to contests.
6. The application shall provide a live scoreboard of the event to users logged into an account.
7. Additional user account types are available to contest organizers that are seeded into the database. A contest organizer shall…
   1. Approve/Deny Advisor registrations
   2. Approve/Deny Team requests for a contest
   3. Approve/Deny Volunteers for the contest
   4. Assign volunteers to the contest
   5. Assign volunteers to be judges for the contest
   6. Control all other aspects of the contest
   7. Create new problems for future contests
8. A volunteer assigned as a judge shall be able to score teams during a contest
9. A master account shall be able to manipulate all data in the database.
10. The application shall be stylized in K-State colors
11. The application shall contain an email system to communicate with participants to be used by contest organizers

# Nonfunctional Requirements

1. Hardware:
   1. The software shall make use of K-State's physical web server infrastructure.
2. Software:
   1. The software shall be hosted on K-State's web servers.
3. Performance:
   1. The software shall make use of K-State's servers which should be able to handle the expected amount of traffic to the application.
4. Usability:
   1. The application shall be designed to use readable text and readable colors to all users.
5. Cultural Sensitivity:
   1. The application shall be designed to be equally accessible to all people.
6. Availability:
   1. The application shall be easily accessible through the internet by anyone interested in participating in the High School Programming Contests.
7. Reliability:
   1. The application should be designed to be fail safe and report proper error messages to the user when necessary.
8. Maintainability:
   1. The application shall be designed so that issues and bugs may be fixed at a fast pace by K-State's IT department.
9. Extensibility:
   1. The application shall be designed to be easily updated by K-State's IT department.
10. Security:
    1. The application shall keep passwords secret through cryptography.
    2. The application shall limit control of users based on their role through encrypted web tokens.

# Semester Goals

## Minimum Viable Product (MVP)

The MVP will include a home page that provides contest updates through a news feed, ensuring all relevant details are easily accessible to users. To provide contest history, an archive of past events will also be available. An integral part of the MVP is the user authentication and data management systems, which will ensure secure and efficient handling of user and event data. Lastly, an email notification system will enhance communication with participants and other stakeholders, providing timely alerts and information regarding the event. Advisors and volunteers will have the ability to login and sign up for the contests. In summary, a MVP will primarily handle sign-ups and information regarding the contests.

## Enhanced Version

Ultimately, the application will replace all administrative paper used to run the contest. This means that features of a complete version will include the ability to run the entire contests from the application. A live contest scoreboard will provide immediate updates on contest progress and results to logged in users.The contest will use a timer built-in to the application and a scoring feature for judges to score teams. A contest organizer will be able to create problems with solutions and test cases for the contest.

# Appendices

Database Diagram:

