Nicholas Shahan (nshahan@cs)
Evan Whitfield (evanw2@cs)

Queueing Theory: A web application for creating and managing wait queues.

In this project, we will implement a general purpose, online queueing system for people to create and manage their own queues online. It provides a simple and easy to use online waiting list. After the queue was created, an administrator would be able to share a URL to link to the queue, where others could add themselves to the queue and wait their turn. For an example of a similar idea with polling, see strawpoll.me. Also this concept is similar to the calendar scheduling application at doodle.com.

As far as we would tell, there are no readily available alternatives online that solve this problem. There are likely places where similar systems are in use. For example, the UW Introductory Programming Lab has an online queue for students to sign in to, but this is very application specific. Our application would allow anyone to quickly create and customize an online queue for any purpose. Features could be added to the queue management system as needed to meet customer demand.

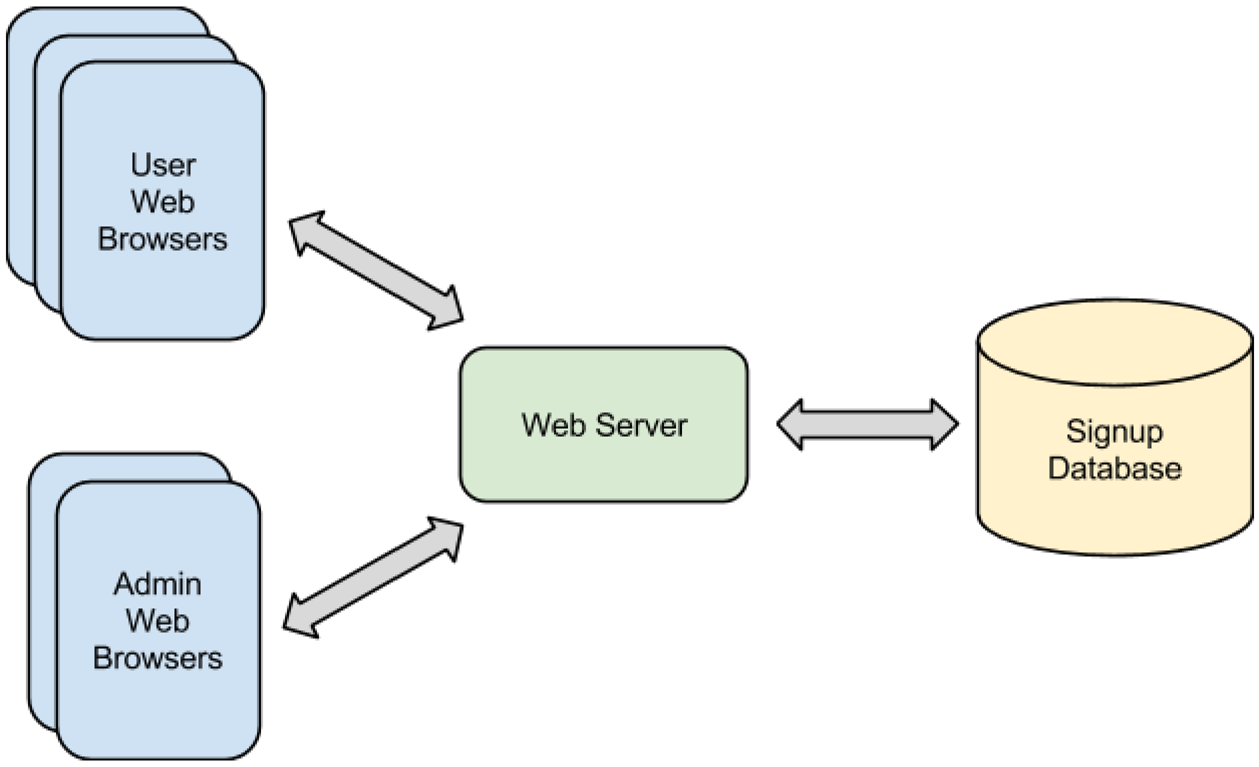
An example use case of this system is that CLUE, the center for learning and undergraduate enrichment at UW could use it to manage the people who are waiting for different types of tutoring. Currently, CLUE uses whiteboards for people to sign in and wait for a tutor, but having a computerized system could bring advantages such as allowing people to check how busy the tutoring center is without being physically present.

Other example use cases exist anywhere that offers drop in appointments like academic advisors, nurses, or barbers. People wanting to drop in could check the list online first, choose to add their name, and then show up.

Software Architecture

Project Feasibility:
This project utilizes a fairly simple web application architecture that can be created and iterated quickly to create a usable product in our short time frame.

Architecture:
Web browsers will be communicating with a web server via HTTP requests. The web server will be saving and looking up information in a backend database, and generating web page content for the browsers request. We would like to utilize asynchronous communication to ensure that the user is always seeing up to date information. For example when a user is looking at a specific queue, their view should be dynamically updated as other users are dequeued, or sign up.



The asynchronous communication between the web server and web browser will be provided by utilizing existing javascript web application frameworks like angular.js or backbone.js. These frameworks will allow us to create dynamically changing HTML documents backed by a database.

The web page creation on the server can be in Python, or possibly PHP depending on the experience of the group.

Challenges and Risks
Learning a new web application framework, will be the largest risk in this project. We hope to select a framework that at least one team member has some familiarity with so that they can act as a mentor to the others during development.

If faced with a situation where no team members have any prior experience with the frameworks we want to use, then we can minimize our risk by assigning a team member to strictly learn and help others as they develop. They would spend all their time learning the ins and outs of the framework and helping others to find solutions when they get stuck.

There are a number of possible use cases that the application could serve, so it could be a challenge to avoid adding too many features and options to the service. Feature creep can be avoided by creating a clear plan from the beginning and only looking for more features to implement if we finish early.