Undergraduate Research Portal

Design Document

11/16/2021

Rev. 2.0

Team MED

Derek Wright

Michael Maddux

Kevin Evans

Course: CptS 322 - Software Engineering Principles I

Instructor: Sakire Arslan Ay

TABLE OF CONTENTS

I. Introduction

This document provides an overview of the Undergraduate Research Portal web application. This application provides a portal where research professors can post openings in their lab and undergraduate students can apply for these positions.

The proposed product is a web app that streamlines the process of WSU faculty recruiting students for their research projects. The design will provide a single place for all research projects in need of students to be listed. The research projects will be detailed in a way that allows students to easily find suitable options. Students will be able to apply to any position they choose, which will substantially increase accessibility to students looking for research opportunities while also providing many more options for the faculty in need of additional help. Students will create accounts consisting of information about themselves that is relevant to the positions they apply to, and faculty will be able to choose students based on this information. This system will benefit both students in search of research opportunities and faculty in search of suitable students greatly.

Section II provides an overview of the architecture, system structure and subsystem interaction. We have a UML diagram that pictorially describes the system and interaction. Screenshots are included of the current user interface.

Section III has a progress report on the current status of the project. We are projected to finish the system on-time.

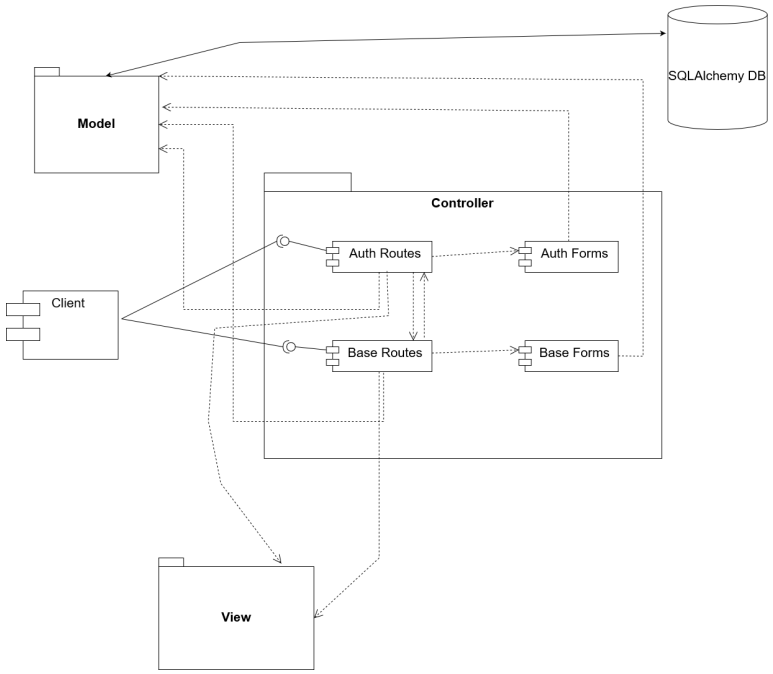
Document Revision History

Rev 1.0	10/26/2021 Initial Version
Rev 2.0	11/16/2021 Iteration 2 Version

II. Architectural and Component-level Design

II.1. System Structure

Our system is structured using the Model-View-Controller (MVC) framework or pattern. The MVC framework was the most logical choice for our use case due to the separation of primary systems. Our MVC framework is separated into three primary systems: The Model, the Controller, and the View. The model is where we define our database model. The controller handles most of the program logic, such as routes and forms. The view is where the UI written in HTML and CSS is defined. MVC will allow easy maintenance due to low coupling and high cohesion.



II.2. Subsystem Design

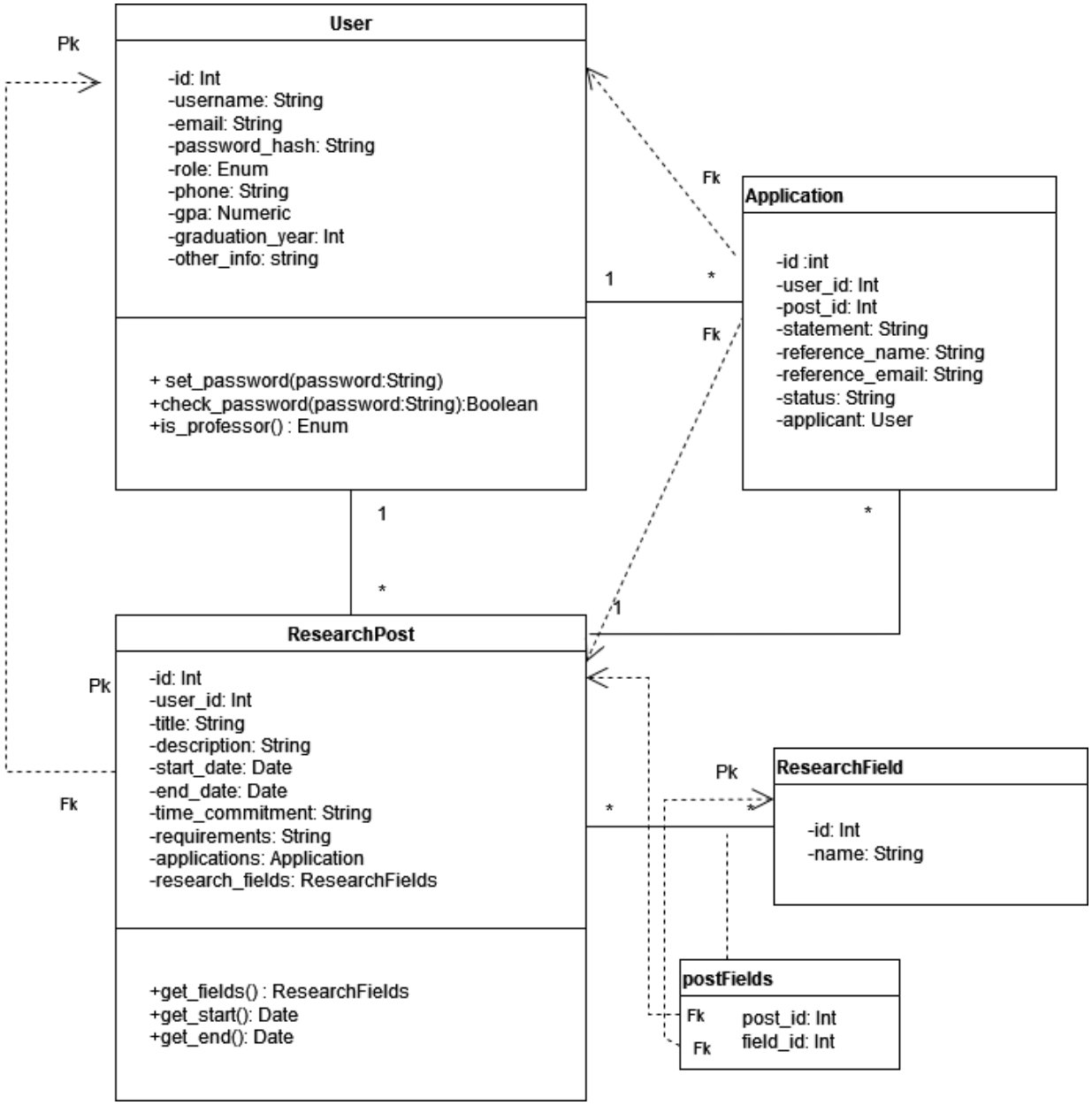
II.2.1. Model

The Model is where the system controls the storage of data. Our model defines several database tables used to store data about separate uses. The model works with the controller to provide data when needed.

Our current models:

Name	Purpose	Attributes
User	Store data about student and faculty users	id: the unique id of the user username: the unique username email: user email password_hash: the user's password hashed. Used to log in role: enum for student or faculty user phone: user's phone number gpa: user's gpa graduation_year: user's graduation year other_info: any other info the user provides
ResearchPost	Store data about each research post	id: unique id of the post user_id: the id of the user who created the post title: title of the post description: description of the post start_date: start of the research opportunity end_date: end of the research opportunity time_commitment: time commitment required for the research opportunity requirements: requirements for the research opportunity research_fields: the research fields associated with the research opportunity
ResearchFields	Store data about the research fields available	id: unique id of the research field name: the name of the research field
Application	A many:many join table that stores data on who applies to what application	id: unique id of the application post_id: the id of the research post user_id: the id of the user application_time: the time of creation modified_time: the time when the application was modified

		status: enum of the status (pending, approved for interview, hired)



II.2.2. Controller

Our controller is responsible for relaying data between the model and view. It is broken down into two subsystems, routes and forms, which each have two parts: default and authenticated. The authenticated forms and routes are to provide more security for logged in users as well as the data stored in the model.

In the controller, the routes handle all of the GET and POST requests (see Table 1 below). The routes are what pass the data necessary for each feature when that feature is being used. They are what provides the interfaces for the user to interact with and are the backbone for the functionality of the application. The forms provide places for users to input data that is then be transferred to the model. These need a route along with it, and are specifically for POST requests (Again see Table - 1). We use them to provide input fields for the user to provide various types of input that the application can then make use of later on.

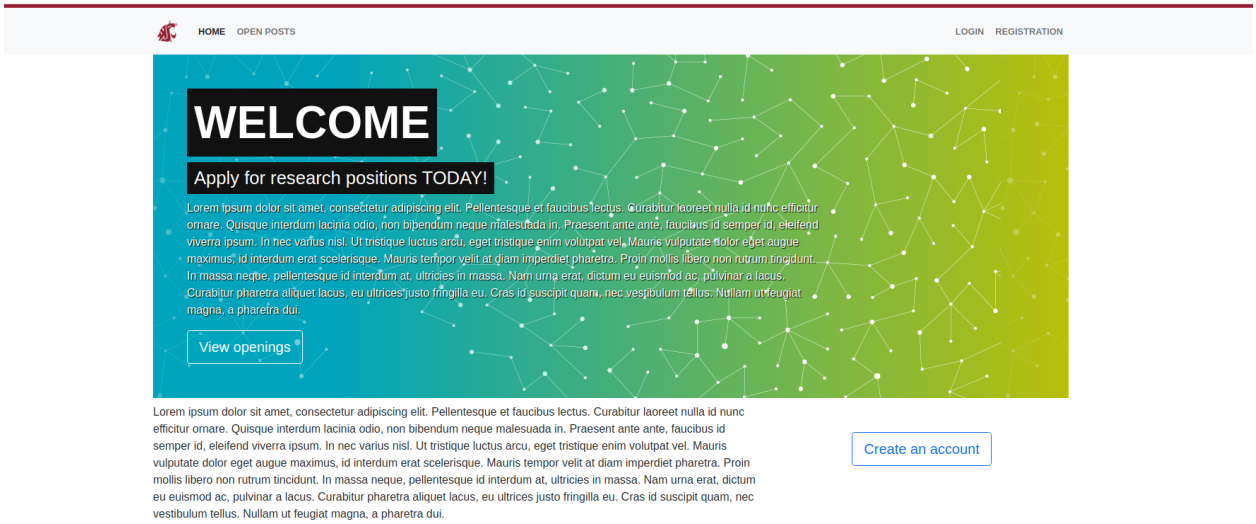
Table 1- Endpoints (Routes) of the web application

Methods	URL Path	Description
GET, POST	/	The landing page of the application
GET, POST	/index	The landing page of the application
GET, POST	/new-post	Creates a new research posting (/post/new)
GET, POST	/edit_post/<post_id>	Edits a research posting (/post/edit)
GET	/open_posts	Views the open postings that students can apply to (/post/view)
GET, POST, DELETE	/delete_post	Deletes a research posting (/post/delete)
GET	/register	Displays the option to register as a student or faculty
GET, POST	/registerfaculty	Registers a faculty member (/register/faculty?)
GET, POST	/registerstudent	Registers a student (/register/student?)
GET, POST	/login	Login form and authenticates
GET, POST	/logout	Logout the current user
GET, POST	/new-application/<post_id>	Applies for a research posting as a student
GET, POST	/post_detals/<post_id>/applicants	Views the applicants of a research posting
POST	/post/applicant/<app_id>/update	Updates the status of a student's application
GET	/my_posts	Views posts belonging to the logged in user
GET, POST	/post_details/<post_id>	Displays information on a post

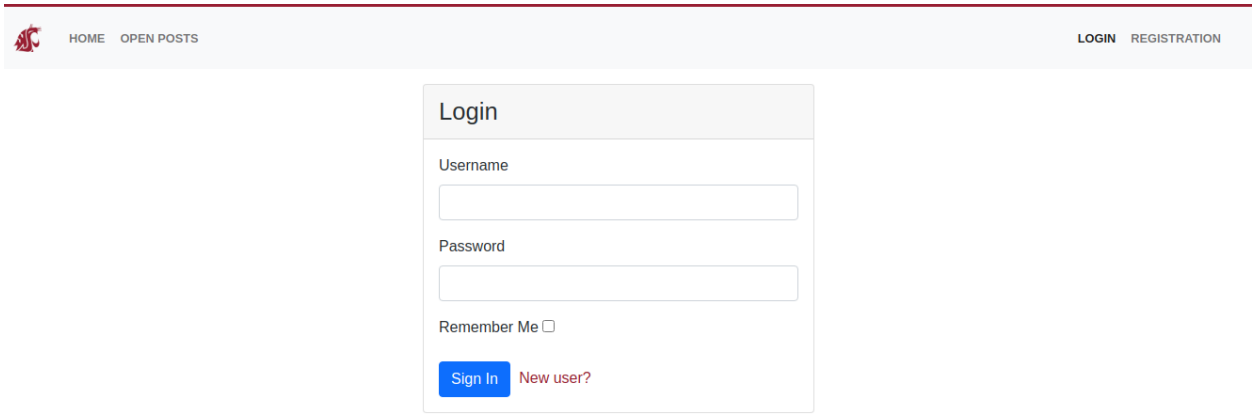
II.2.3. View and User Interface Design

The interface will use a palette to match the WSU website and their brand guidelines [1]. The user interface will use Bootstrap for the main interface components [2].

Index/landing page:

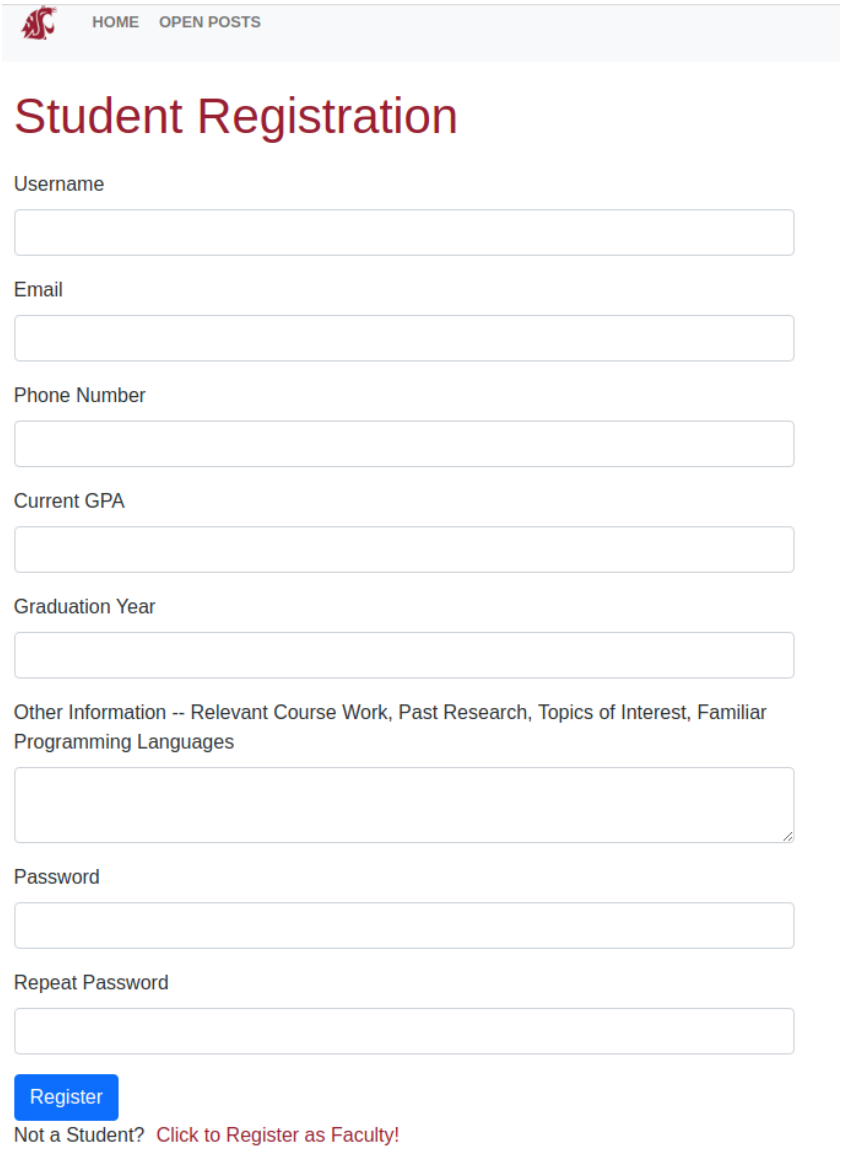


Login:

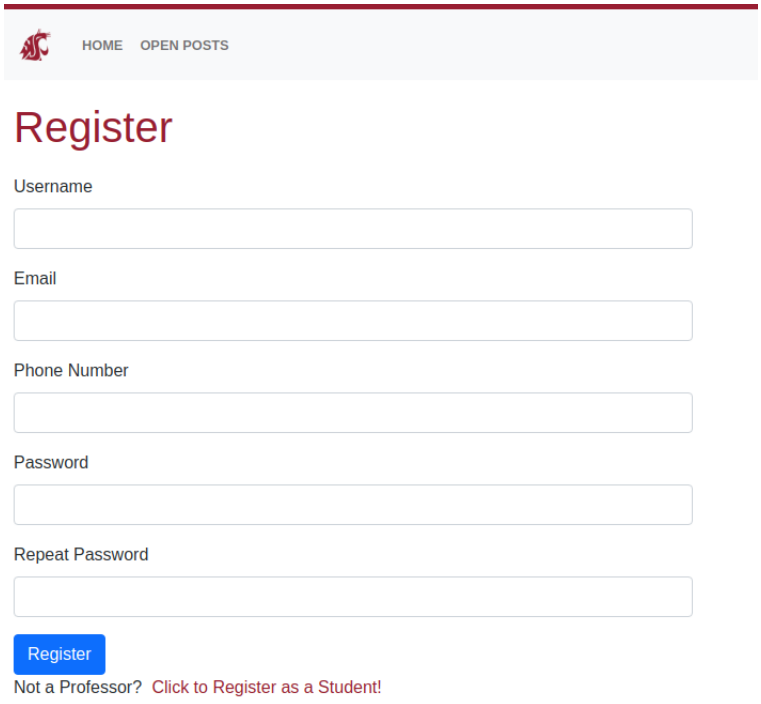


Student registration

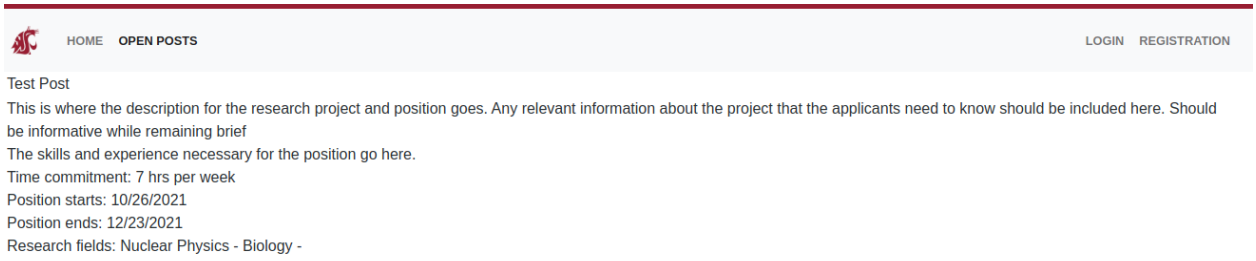
Design Document 7


HOME
OPEN POSTS
Student Registration
Username
Email
Phone Number
Current GPA
Graduation Year
Other Information -- Relevant Course Work, Past Research, Topics of Interest, Familiar Programming Languages
Password
Repeat Password
Register
Not a Student? Click to Register as Faculty!

Faculty registration:



Open positions* needs additional styling



III. Progress Report

Rev 1.0

We have finished all the items assigned to iteration1 by the deadline. Both students and faculty members can register and login. Faculty members can post research positions. Students can view all open research positions.

Rev 2.0

We completed all open iteration 2 issues. We added the ability to view applicants to a post, ability to apply to a position, created a better template for research posts, and added details to research positions. We also updated the model to include an applicant relationship.

IV. Testing Plan

Unit Testing:

Our plan is to use Pytest. We will test all of our major features, for example, creating user accounts, creating research posts, modifying information, testing Model functions, testing route functions. One of our members has set up a git workflow that will automatically run our testing suite for each new pull request. This will ensure no broken code gets committed to the iteration or main branch.

Functional Testing:

Each member will work through the list of use cases after a major commit.

UI Testing:

Similar to Functional Testing, each team member will navigate the site after a major commit.

V. References

[1] Web and Digital Style Guidelines, Washington State University, https://brand.wsu.edu/web-digital/

[2] Bootstrap Documentation, https://getbootstrap.com/