

Term Project

Design Document

10/23/2019

Version <1.0>



SpaghettiCoders

Jimmy Zheng

Hien Duong

Peng-Yu Chu

Eric Furukawa

Course: CptS 322 - Software Engineering Principles I

Instructor: Sakire Arslan Ay

TABLE OF CONTENTS

I.	INTRODUCTION	2
II.	ARCHITECTURE DESIGN	2
II.1.	OVERVIEW	2
III.	DESIGN DETAILS	4
III.1.	SUBSYSTEM DESIGN.....	4
III.1.1.	[Subsystem Name].....	<i>Error! Bookmark not defined.</i>
III.1.2.	[Subsystem Name].....	<i>Error! Bookmark not defined.</i>
III.2.	DATA DESIGN	5
III.3.	USER INTERFACE DESIGN	5
IV.	PROGRESS REPORT.....	8
V.	TESTING PLAN	9
VI.	REFERENCES	9

I. Introduction


This design document has been provided to outline and detail the appropriate subsystems introduced as part of the greater TA webapp in order to improve program timeliness, quality and scope. Subsystem details includes the associated databases, interaction with other subsystems and user interface accommodations. Additionally, the architectural design of choice for the project will be introduced and explained in the document. Progress and changes are also recorded to improve organization. This is the first iteration draft of the design document and thus no changes need to be discussed here. Section II will describe the architecture design, this section will include a UML component diagram. Section III will discuss design details. Such as subsystem designs, data design and user interface design. Section IV will discuss the progress of the project. Section V will include all references.

Document Revision History

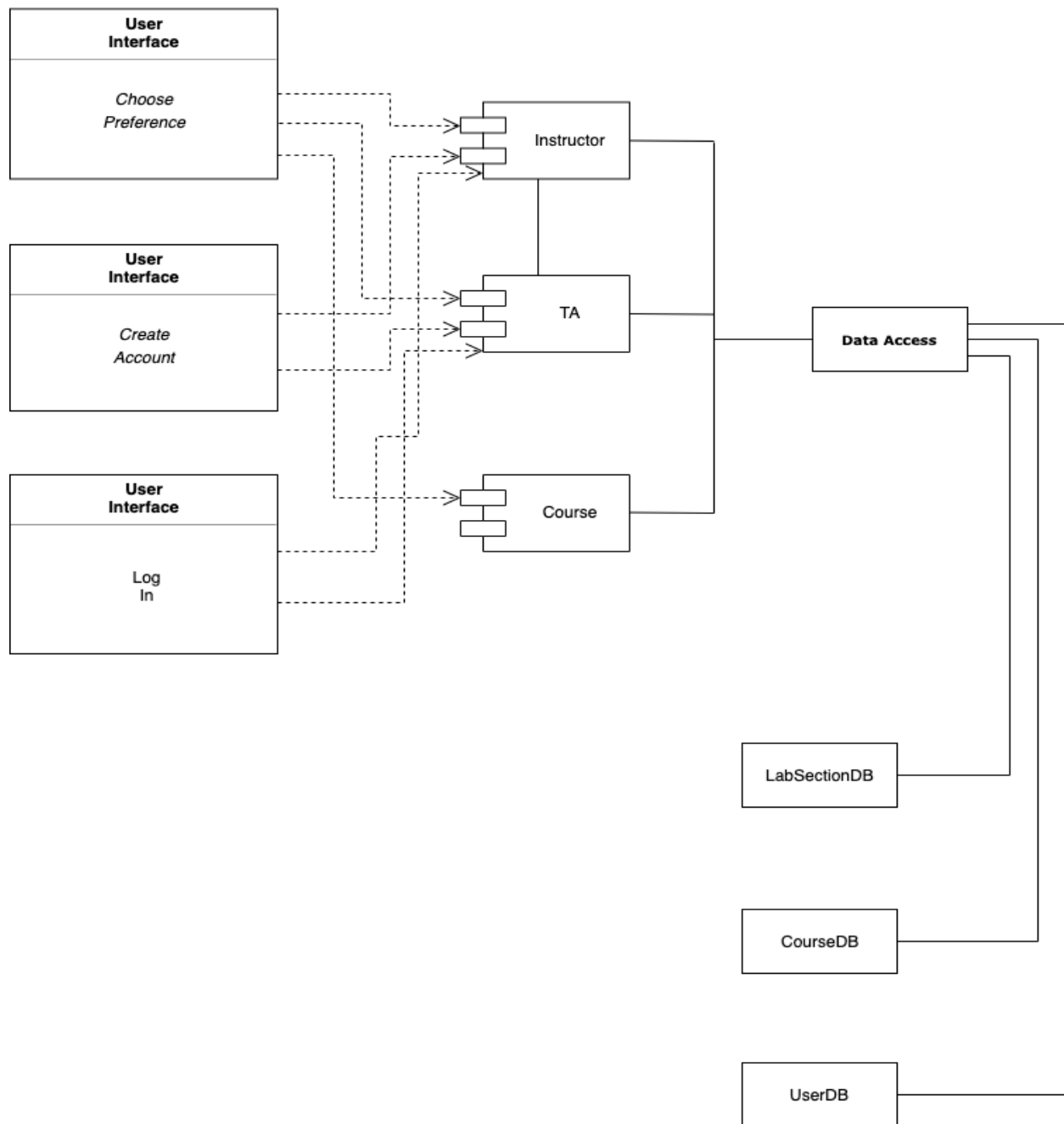
Rev 1.0 2019-10-23 Initial Version

II. Architecture Design

II.1. Overview

The decided upon architectural pattern is the Client-Server Architectural Pattern. The TA webapp has a defined and easily identifiable front-end and back-end side to its implementation, correlating to the client and server side described in the C-S pattern respectively. Additionally, there is no need for any direct peer-to-peer communication which is generally a less reliable architectural pattern due to we  peer nodes. The TA webapp will need to be able to receive information from the user, perform data calculation and/or correlation and then return the appropriate information back to the user. This process fits the C-S pattern's functionality.

The coupling of the client front-end and server back-end implementations must be kept low. This will be reflected in the fact that the only interaction between the two large subsystems will be the receiving of data through the HTML forms. Cohesion will however be kept through the similarities in the teacher and the TA forms as many functionalities, especially in login information, will overlap. This is why the subsystems will focus on division between tasks instead of between user types. The functionality of all login functions, for example will be cohesive within the login component. It will be important to keep in check the coupling between the user types as functionality is similar but not the same.



The first level of decomposition on the webapp splits the program into a choose preference functionality, login functionality, and a create account functionality. The choose preference component is responsible for allowing users to enter their preferred TA selection or class selection and have them associated to their account and displayed to the other respective party. The login component is responsible for allowing users to provide their username and password to associate their session with their section of the databases. The create account component is responsible for allowing users to provide their basic info and account credentials for use in accessing their specific database section. Each of the top layer components will be used for the instructor, student and class pieces of the program. The instructor component encompasses all of the functions available to instructors. The TA component does

the same for all student functions. The course component will be used in parallel to keep track of preferences per account, but only under the choose preference top component. All of these components will interact with three databases (Labsection, Course, and User). These databases could be considered external components, but are more so a part of the internal program. No other external interaction features have been introduced.

III. Design Details

III.1. Subsystem Design

The HTML front end is composed of multiple HTML files and is primarily responsible for presenting the webapp with a visually appealing but utility based user interface. It also provides forms for the user to send information to the backend, which is done with help from the Javascript component.

The Javascript portion will follow the suggested MVC pattern to model the cycle of receiving information and displaying it. This will enable the client and server interaction components discussed below.

Messages in the login and create account components will contain entered information to which the back end will check against the database or create a new database for each respectively. The course preference selection will take a form of desired classes on the TA side and send it to teacher sessions which will in turn take the teacher's selections and send it to the TA's. In this case "sending" really means storing the information in the database from one side and get requesting it on the otherside when the other logs in.



(For iteration 2)

III.1.1. [Subsystem Name]

III.1.2. [Subsystem Name]

Make sure to describe the interfecace of your server (i.e., the routes of your Web API server)

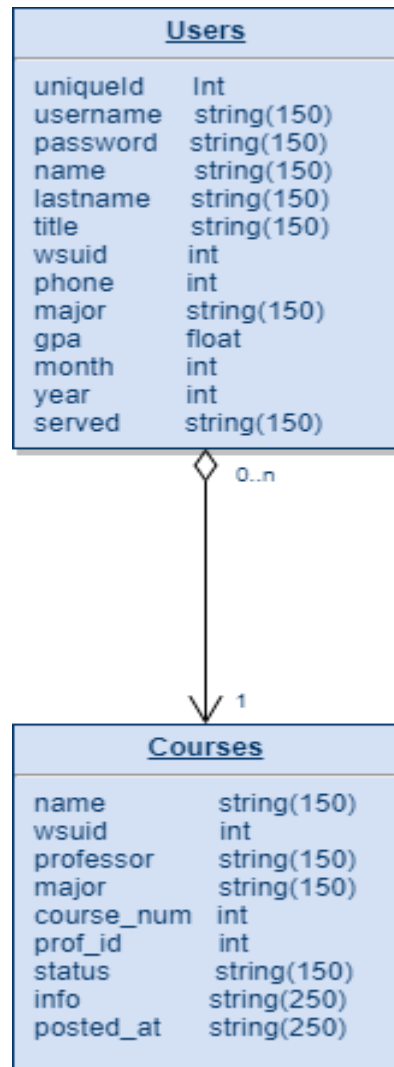
For each route mention:

- the URL
- sample data that is being send/received

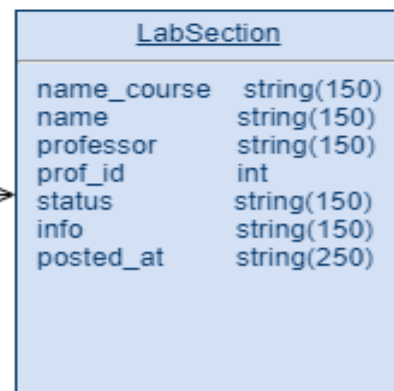
(I suppose your routes will exchange the data in JSON format; show examples of the JSON objects sent/received in requests to your routes)

III.2. Data design

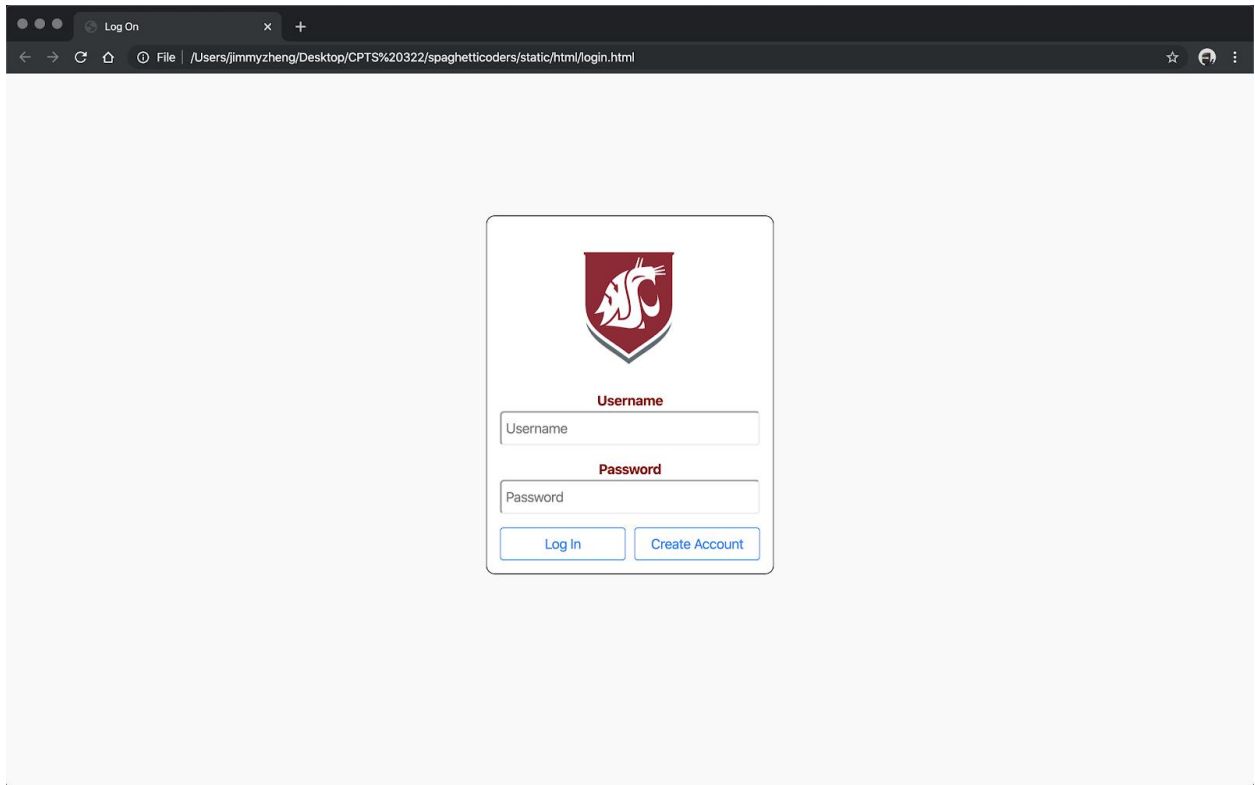
The database being used is a SQLAlchemy database. The user database will contain a list of all users, professor and student, Courses database will be a list of all courses that the professor wants a TA for. Students will be able to get the course and submit their request. Lab database will contain all the labs sections TA can a apply for. Professor can add lab sections to database.



- You should differentiate between instructor and student users. I suggest you to include 2 additional tables: one to store student data and another to store instructor. Store the student and instructor specific data in these tables . This table will be associated to the Users table through "uniqueid" attribute of user.
 - Separating lab sessions from courses will complicate the schema. You may assume that each course lab is a separate course.
 - You need an additional table to store information about TA applications.
 - Yu may need additional tables to store some other supplemental data as well.
- Please revise you schema and update this diagram.



III.3. User Interface Design



I. The user interface shown above is the login screen. It consists of two text boxes and two buttons. The first text box is for the user's username and the second is for the user's password. The button on the right labeled login is for the user to login to their account and the button on the right is for users who would like to create an account. The use-case #3, which is "login account" for all users utilizes this user interface.

Contact Information

WSU Email
Enter WSU Email

Password
Enter Password

Title
Professor

First Name
Enter First Name

Last Name
Enter Last Name

WSU ID
Enter WSU ID

Phone Number
Enter Phone Number

Submit Cancel

II.

This user interface above is for the user to input their information such as their WSU email and their password. This screen consists of six text box, one drop down menu, and two buttons. We have text boxes for the users WSU email, password, first name, last name, WSU ID, and their phone number. One drop down menu for their title which has student and teacher. The button on the left is for the user to submit their information. The button on the right is to cancel, which will bring the user back to the login screen. The use-cases #1 “Create Instructor Account” and #2 “Create Student Account” utilize this user interface.



Additional Information

Major

Enter Major

GPA

4.0

Graduation Date

January 2019

Served Before

Yes

Submit Cancel

III.

This user interface is the additional information page. This page will only show up if the user is a student. There is only one text box on this screen and it is used to mark the users major. This page consist of four drop down menu, the first one we will see is the GPA drop down menu which will allow the user to submit their GPA. The next set of drop downs we will see are the ones for the users graduation date. And lastly we have a drop down for if the user has served as a TA before. The button on the left is to submit additional information on this page. The button on the left will bring the user back to the login page. The use-cases #2 “Create Student Account” and #6 “Enter Additional Information” utilize this user interface.

IV. Progress Report

The first major iteration accomplishment was creating the multiple HTML files for the create account, login, student and teacher pages (including appropriate information forms). CSS styling were added to make the user interface neater and more visually appealing. SQLAlchemy/Flask backend was created for local account information storage. The javascript for logging into an account stored on the database was added to work with the login HTML page. Javascript for creating an account was also added to accompany the create HTML page, which takes form entered information and adds an entry to the database.

V. Testing Plan

For iteration 2...

VI. References

Google, google.com

Blackboard, laran.wsu.edu

MyWSU, my.wsu.edu

RateMyProfessor, Sakire Arslan, (Class Project)

No Journal Sources to reference as of now