

Pied TA Piper

Design Document

10/12/2016

Version 1.0

Equipo

Sam Mahan

Austin Kappl

Sean Inouye

TABLE OF CONTENTS

I.	Introduction..	3
II.	Architecture Design..	3
II.1.	Overview..	3
III.	Design Details.	3
III.1.	Subsystem Design.	4
III.1.1.	Login/SignUp	4
III.1.2.	Edit Info	5
III.1.3	Create Course	5
III.1.4	TA Selection	5
III.1.5	Apply to TA a Course	6
III.1.6	Emailer	6
III.2.	Data design.	6
III.3.	User Interface Design.	7

I. Introduction

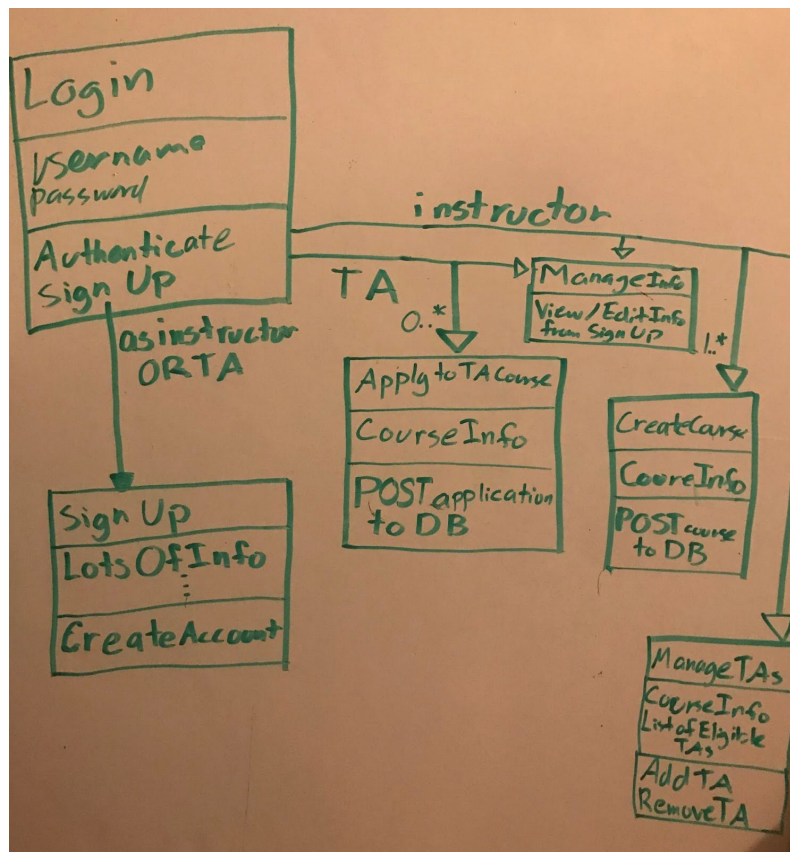
This is the design document for the first iteration of our application. In this version there are added detail to each of the subsystem designs. The main goal of this application is to help professors find TA's for courses. We hope to provide a streamlined experience for both professors and potential TA's. Section II includes an overview of the high level design for the website. Section III houses all of the design details for the application, which includes use cases, data storage design, and the UI design. Along with this document is the first iteration of code for the website. The code itself mainly consists of skeletons for the different components.

Document Revision History

- Rev 1.0 2016-10-12 Initial Design Doc

II. Architecture Design

II.1. Overview



- Login authenticates users and directs new users to sign in. Manage Info allows users to view and change their saved info. Create Course allows an instructor to create a new course to add TAs to. Manage TAs allows instructors to add and remove TAs from their created classes. Apply to TA Course allows TAs to apply and have their name added to the list of potential TAs for courses.

- All interaction between subsystems will be done either through URL variables or JSON.
- Login/Sign Up uses the bcrypt library to encrypt user passwords

Subsystems were designed to each encompass a related set of tasks. For example, sign-up and login are together because they are similar and logging in requires having previously signed up. No subsystem is truly dependent on another besides in very obvious fashions. The coupling is that you must be logged in to access user parts of the website, for a TA to apply for a course that course must have been created already, and for an instructor to assign TAs to a course, the course needs to already be created and have had TAs apply for it.

III. Design Details

III.1. Subsystem Design

Subsystems:

- Login/Sign-up
- Edit info
- Create course
- TA selection
- Apply for to TA a course
- Emailer

The diagram you provided is a class diagram rather than an "architecture diagram". When you revise your document, please include an UML component diagram to illustrate your architectural design.

Assuming you adopt an architecture similar to the smile project, your architecture should include a frontend (HTML+JS) , a backend (Rest API), and a database. Further, the frontend and backend can be designed based on MVC pattern where the view and the model were separated. In your document you should explain:

- How MVC is applied at the front end? What is role of the model, view and controller? How is model and view are decoupled. Don't only provide generic description for MVC. Provide specific details and examples.
- How MVC is applied at the backend end? What is role of the model, view and controller at the back end? What is the interface for the backend API? What routes/requests does it support? Again, please provide specific details and examples (rather than a generic description).

III.1.1. Login/Sign Up

- Users will first be prompted to choose that they are either a Potential TA or a Professor.
- They will then enter their information which will be sent to the server through a POST request for validation.
- Passwords are not stored on the server, rather they are salted and hashed and the hash is stored.
- A session will be created for that user, specifying which pages they are allowed to access.
 - If they try to access an inaccessible page, they will be redirected to the user homepage
 - After an hour, the session will timeout and the user will need to log back in
- If the user clicks sign up, they will be directed to a sign-up form, customized based on whether they chose TA or Professor.
- On submission of this form, a POST request is made to the server that will get routed to account creation. This is where the password gets salted and hashed (using bcrypt).

As I mentioned above, the above diagram looks like a class diagram. For iteration2, please revise it and include as a class diagram. Please make sure to use UML notation. Use a UML or drawing tool to create the diagram.

You can utilize the text you have in this section to describe the classes in your class diagram.

- After account creation, professors will get redirected to a course creation page and students will get directed to a course application page

III.1.2. Edit Information

- Current information stored in database is displayed in the same input boxes in which it was entered via the sign-up.
 - When the edit info button is pressed, a GET request is sent to the server with the TA-space-tag and a json containing all the information for that specific TA in the TA table is sent back.
 - Information is parsed from the json and displayed in the input fields
- This information is mutable and the user can alter any of the information within the input boxes.
 - Editable text boxes will be implemented to allow the altering of information by the user
- When the user decides they are finished manipulating their info and they click submit, the data is then stored back in the database.
 - Once the submit button is clicked the entered data will be “jsonified” and a POST request with the embedded json is sent to the server where it will be stored in either the Professor table or TA table depending upon a tag in the URL.

III.1.3 Create Course

- User can be sent here from a redirect after sign up or through a link in their My Courses section.
- Form contains fields specifying the course as well as the number of TAs that will be required and their roles.
 - For example, for CptS 122, Andy may require 10 Lab TAs and 5 Utility TAs, where Sakire only needs 2 Regular TAs for Cpts 322. All these will be possible to make and will be represented by a JSON string in the db. (This will be specified more in iteration 2 when it is actually implemented)
 - On submission, a POST request is made to the server to store the course.

III.1.4 TA Apply for Course

- The user can be sent to this page via a link in the toolbar denoted “Apply”. If they have yet to make any applications then the TA landing page will contain a link to the Apply page and a prompt to fill out an application.
- The form is made up of input fields with their accompanying labels, and the user will be prompted to fill in all required fields
- When the user clicks the submit button the information is sent to the server and stored in the database.
 - The data in the input fields is jsonified
 - A POST request containing the json data is sent to the database, the json data is converted into a row object and then stored in the Applications table

III.1.5 TA Selector

- A GET request is sent to the server when the page is loaded to retrieve all classes created by the instructor.
- Each class is displayed in an accordion format (click on the class for more detail).
 - When the class is clicked on, a TA selector will come up where the instructor can add or remove TAs.
 - When they are satisfied, they will submit which will send a POST request to the server, updating the JSON string that holds TAs

III.1.6 Emailer

- When a TA is added to a course, they will be notified via email

III.2. Data design

We will have four data tables: potential TAs, TA applications, Courses, and Instructors.

Instructors and Courses can be joined via `Instructor.id = Course.instructorID`.

TA applications can be joined with Courses via `Course.id = application.courseid` and with potential TAs via `TA.id = application.taID`

Instructors Table

id(WSU)	loginHash	firstName	lastName	salt	email
---------	-----------	-----------	----------	------	-------

TAs Table

id	login-Hash	first-Name	last-Name	salt	major	cum-GPA	expect-edGrad	email	prev-TA
----	------------	------------	-----------	------	-------	---------	---------------	-------	---------

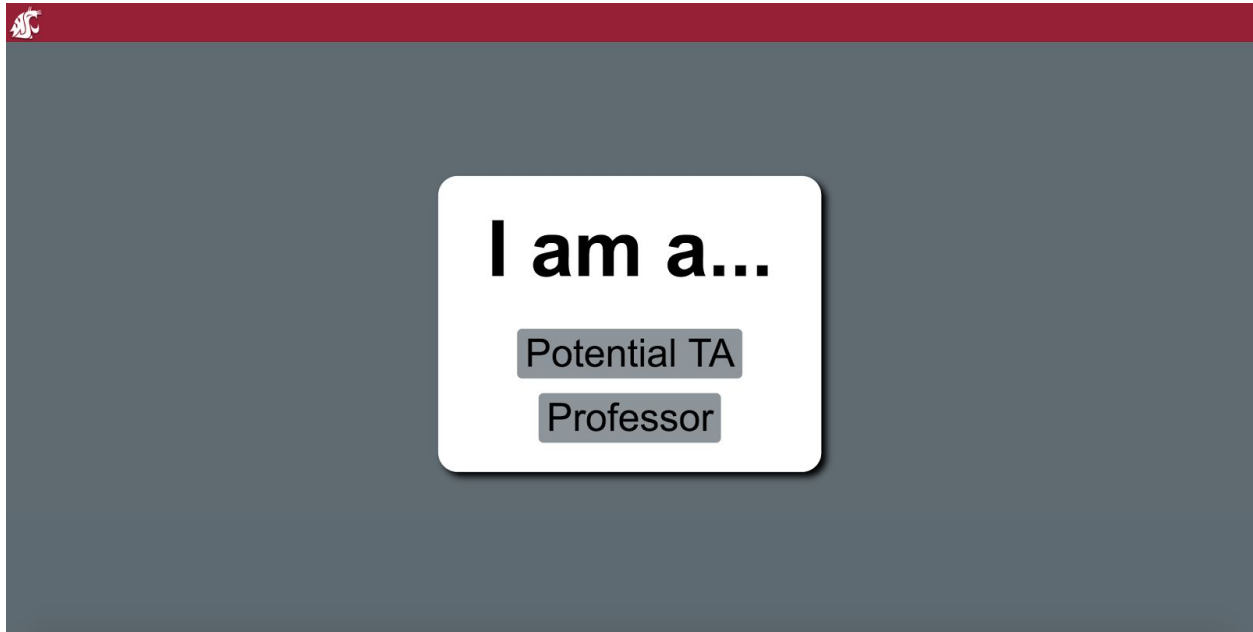
Courses Table

id	subject	courseNumber	description	profIdentifier	sections
----	---------	--------------	-------------	----------------	----------

Applications Table

id	TAIdentifier	GradeIn-Class	semApply	semTaken	courseTA	courseNum
----	--------------	---------------	----------	----------	----------	-----------

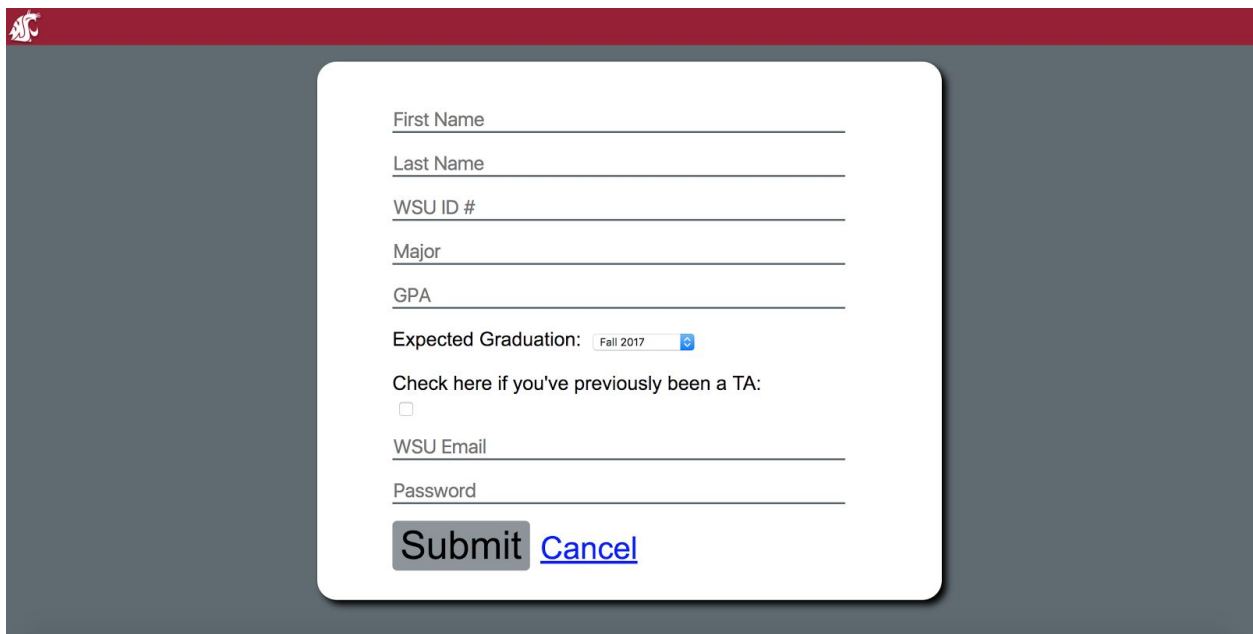
III.3. User Interface Design



I am a...

Potential TA

Professor



First Name

Last Name

WSU ID #

Major

GPA

Expected Graduation: Fall 2017

Check here if you've previously been a TA:


☐

WSU Email

Password

Submit [Cancel](#)

TA Sign Up Page: Student 1



First Name _____

Last Name _____


WSU ID # _____

WSU Email _____

Password _____

Submit [Cancel](#)

Instructor Sign Up Page: Instructor 1



Email:

steve.jobs@wsu.edu

Password:

Login [Sign Up](#)
[Cancel](#)

Sign In Page: Instructor and TA 2