



# Final Report - CSCE 606

## Software Engineering

### Team Matrix

#### **Team Members:**

Harmanpreet Singh

Suman Kumari

Kushagra Jain

Rishabh Bassi

Udhav Gupta

Divyansh Bokadia

## **1. Project Summary**

The Engineering Department at Texas A&M hires TAs/PTs for undergraduate courses; the current hiring process and maintenance of the records of all these hires are done manually using an excel sheet, making it laborious and prone to errors. As a result, the Engineering department wants to create a portal where undergraduate students can apply for TA/PT positions; their profile information (contact, resume, etc.) can be received and reviewed; and a choice regarding their recruitment can be made. In order to facilitate communication, the department also stipulates that professors must be able to view the TAs' contact information on the portal.

Our primary stakeholders will be students selected as TAs/PTs, Professors, Hiring Managers, and coordinators. Our approach towards implementing the project was to split it into two major parts one for the students who can submit their contact details and profiles, and the second part where the users (Professors, Hiring Managers and Coordinators) can sign up and login on the portal and perform their respective actions. While signing up users, we ask for. The roles they want to sign up for and based on the role of the user logging in, we redirect them to respective views. If the user is Professor, they can see their courses and the details of the TAs allotted to them and provide feedback to the TAs/PTs. If a Hiring Manager or coordinator logs in, they can see a list of applicants with all their details and update their hiring status (review, hired, rejected, applied). They can also add new subjects/sections to the portal and assign them to the respective TA/PT hires. In addition to that, the hiring manager or coordinator can also assign professors to the subject. Lastly, we have also created a super user who can assign the initial hiring manager or coordinators to handle the project.

## 2. User Stories Description

### 2.1 Application for TA position

#### 2.1.1 UI FOR TA APPLICATION FORM

*Points: 4 and Status: Completed*

The first interface we implemented was of TA application form. The mock form had various data-fields to input details of applying student.

STUDENT TA APPLICATION PAGE

NAME

UIN

EMAIL  @tamu.edu

PHONE +1

Do you have a campus job?

Courses completed

Are you an undergraduate?

RESUME  (File types: pdf, doc, docx)

TRANSCRIPT

Figure 1: Mock UI for TA Application form

#### 2.1.2 FIT AND FINISH FOR STUDENT APPLICATION FLOW

*Points: 2 and Status: Completed*

Features like drop-down options to appropriate data-fields were added to the application form. Also, the application form is presented in a tabulated manner to the user.

#### 2.1.3 TA APPLICATION FORM UI INTEGRATION WITH CONTROLLER

*Points: 2 and Status: Completed*

The application is now storing all the incoming input details of the applicant in the database. All the information is now related to Students Controller.

#### 2.1.4 CHANGE RESUME AND TRANSCRIPT TO PROVIDE LINK

*Points: 4 and Status: Completed*

In order to collect resumes and transcripts from all students and provide them to HR, we require students to provide links to resume/transcript in respective data-field.

### 2.1.5 CSS FOR TA APPLICATION FORM

*Points: 2 and Status: Completed*

Presenting the application form in a rectangle box and other additions for easy user experience were included.

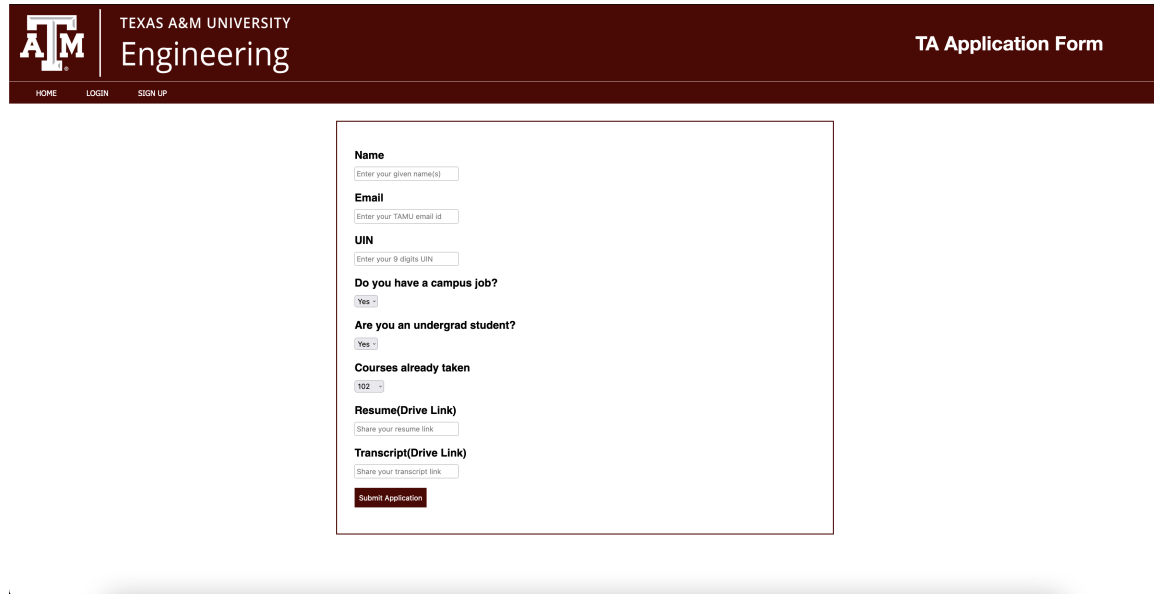
The image shows a web browser window displaying the TA Application Form. The header is a dark red bar with the Texas A&M University Engineering logo on the left, which includes the 'ATM' monogram and the text 'TEXAS A&M UNIVERSITY Engineering'. On the right side of the header, it says 'TA Application Form'. Below the header, there are three small links: 'HOME', 'LOGIN', and 'SIGN UP'. The main content area is a white rectangle with a thin red border. Inside this rectangle, the form fields are as follows: 'Name' with a text input field labeled 'Enter your given name(s)'; 'Email' with a text input field labeled 'Enter your TAMU email id'; 'UIN' with a text input field labeled 'Enter your 9 digits UIN'; 'Do you have a campus job?' with a 'Yes' radio button; 'Are you an undergrad student?' with a 'Yes' radio button; 'Courses already taken' with a dropdown menu showing '102'; 'Resume(Drive Link)' with a text input field labeled 'Share your resume link'; 'Transcript(Drive Link)' with a text input field labeled 'Share your transcript link'; and a red 'Submit Application' button at the bottom.

Figure 2: TA Application form

## 2.2 Hiring Manager View

### 2.2.1 HIRING MANAGER CONTROLLER

*Points: 4 and Status: Completed*

The Hiring Manager accesses all the active applications for TA, and they are presented to HR in a tabulated manner.

### 2.2.2 ADD MORE FUNCTIONALITY ON THE HIRING MANAGER VIEW

*Points: 4 and Status: Completed*

The Hiring manager can update status (hire/reject) on any active TA application. Features of sorting applications based on name/email/status were added

### 2.2.3 ALLOW ENTERING NEW COURSE AND SECTION FUNCTIONALITY AND VIEW

*Points: 2 and Status: Completed*

The hiring manager or coordinator can use this feature to create a new course and also make the required sections for it.

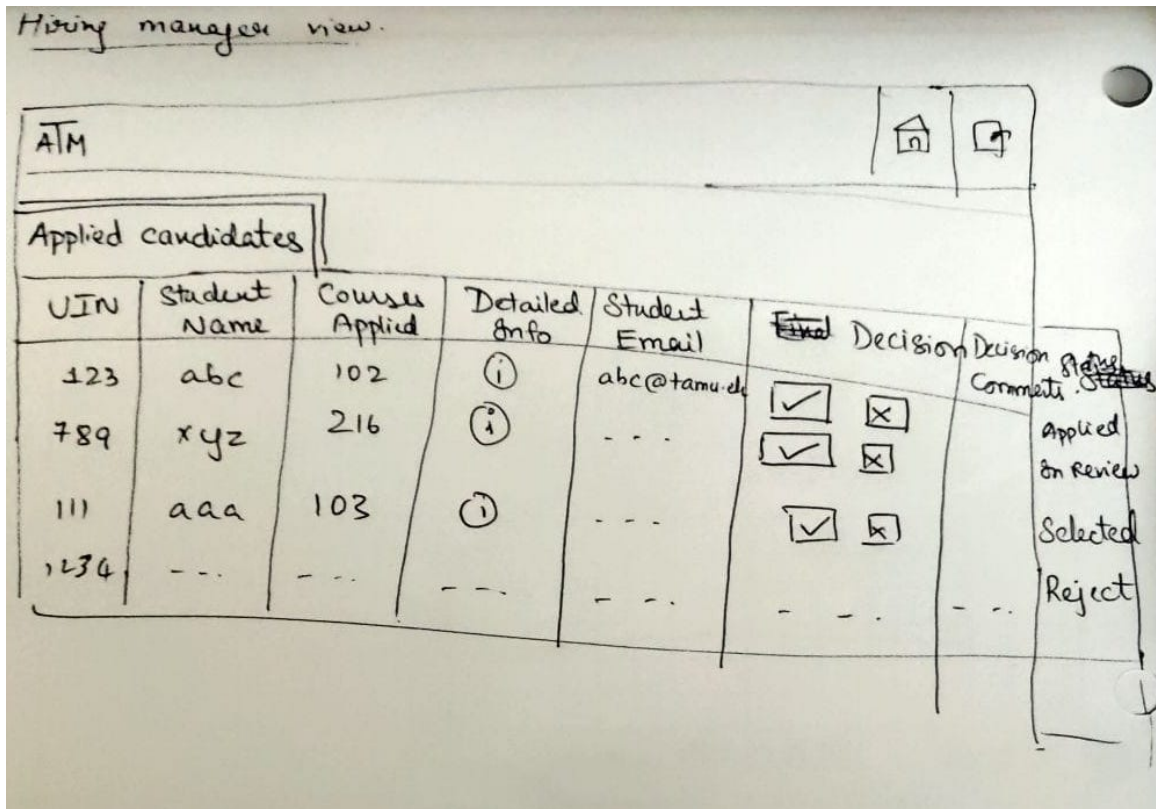


Figure 3: Mock UI for Hiring Manager View

#### 2.2.4 CSS STYLING FOR STUDENTS

*Points: 4 and Status: Completed*

#### 2.2.5 INSERTION OF MULTIPLE SECTIONS FOR COURSE ADDITIONS IN A SINGLE CLICK

*Points: 4 and Status: Completed*

As there can be many sections in a course, to avoid the process of adding a course-section pair individually, this feature is added to add all the sections of a particular course by inputting all the sections in comma-separated manner.

#### 2.2.6 PAGINATION FOR APPLICATION

*Points: 4 and Status: Completed*

Owing to the fact that there are a large number of applications for TA active a particular time, this feature displays 20 applications on a webpage at a time. The user can toggle between pages using list of page present at the bottom.

Name	Email	UIN	Undergraduate	Campus Job	Resume	Transcript	Application Status
<a href="#">aaa bb</a>	asa@gmail.com	121232212	true	true	<a href="#">Click</a>	<a href="#">Click</a>	applied <input type="button" value="Update"/>
<a href="#">aaaa</a>	aaa	1223	false	false	<a href="#">Click</a>	<a href="#">Click</a>	applied <input type="button" value="Update"/>
<a href="#">qqqq</a>	qqqq	1234	false	false	<a href="#">Click</a>	<a href="#">Click</a>	applied <input type="button" value="Update"/>
<a href="#">qqq</a>	qqq@gmail.com	123123	true	true	<a href="#">Click</a>	<a href="#">Click</a>	applied <input type="button" value="Update"/>
<a href="#">John Doe</a>	jdoe@gmail.com	122132341	true	true	<a href="#">Click</a>	<a href="#">Click</a>	applied <input type="button" value="Update"/>

< 1 2 >

Figure 4: Hiring Manager View

## 2.3 Coordinator View : Assign courses to Hired TAs

### 2.3.1 ADD COORDINATOR FUNCTIONALITY TO CHOOSE COURSES AND SECTIONS

*Points: 4 and Status: Completed*

Following the acceptance of a student to become a TA, the coordinator can assign the TAs to existing courses and sections. Also, the TA for a course and section can be updated for the following semesters if needed.

### 2.3.2 ALLOW COORDINATOR TO INPUT ONLY UNIQUE COURSE AND SECTIONS

*Points: 2 and Status: Completed*

Restricting the coordinator to assigning a course-section to multiple TAs.

### 2.3.3 COORDINATOR FUNCTIONALITY AND VIEW FOR HIRED STUDENTS

*Points: 4 and Status: Completed* The coordinator can view all the hired TAs in a tabulated manner, where he can assign each TA with a course (using dropdown) and a section (textbox). Updates afterwards can also be made.

### 2.3.4 CSS STYLING FOR SUBJECTS

*Points: 2 and Status: Completed*

## 2.4 Professor View

### 2.4.1 PROFESSOR AND SUBJECT MAP

*Points: 2 and Status: Completed*

To maintain professor details for a course, and also allow updating these details for any course at any time.

CO-ORDINATOR VIEW :

SELECTED CANDIDATES :

UIN	STUDENT NAME	ASSIGN COURSE	ASSIGN SECTION	EMAIL	
123	abc	<input type="checkbox"/>	<input type="checkbox"/>	abc@tamu.edu	i
456	def	<input type="checkbox"/>	<input type="checkbox"/>	def@tamu	i
789	xyz	<input type="checkbox"/>	<input type="checkbox"/>	xyz@tamu	i

Figure 5: Mock UI for Coordinator View


<div>  <div>           TEXAS A&amp;M UNIVERSITY Engineering         </div> </div>	Assign Courses to Hired TA				
HOME STUDENTS NEW SUBJECT ASSIGN PROFESSORS		LOG OUT			
Name	Email	UIN	Course name	Course section	Actions
Divyansh Bokadia	dbokadia@tamu.edu	733000615	102	A	<button>Update</button>
ABC abcd	ddt@tamu.edu	111222333	CSC	1	<button>Update</button>
lol ty	lol@gmail.com	564654655	102	A	<button>Update</button>
adaf test	sdfsd@gmail.com	234232222	216	B	<button>Update</button>

Figure 6: Coordinator View

## 2.4.2 PROFESSOR'S VIEW

*Points: 2 and Status: Completed*

The professor can view the details of the TA that is allotted for the course. Also there is provision to provide feedback to the TA, which in turn can also be viewed by HR in order to evaluate a TA's credibility.

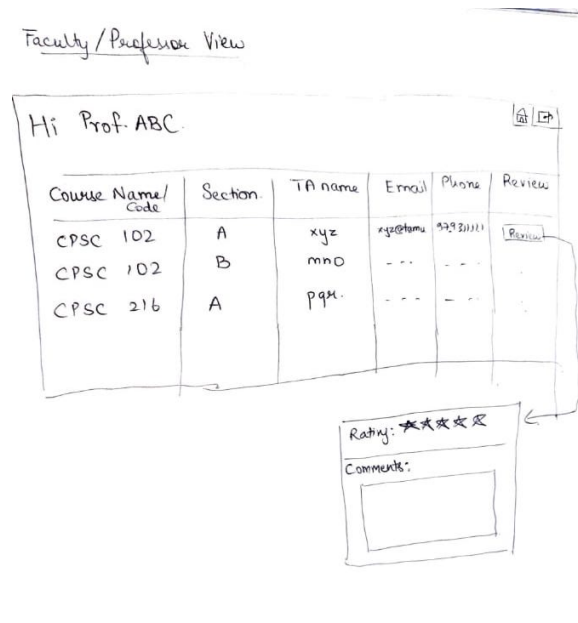


Figure 7: Mock UI for Professor View

### 2.4.3 CSS STYLING FOR PROFESSORS PAGE

*Points: 2 and Status: Completed*

TEXAS A&M UNIVERSITY

Engineering

Professor Info

Course list: Comp

Course section: 100

TA: Cap America

TA Email: acp@gmail.com

Feedback good work!

Submit Review

Figure 8: Professor View

## 2.5 Main Required Features for Application

### 2.5.1 LOGIN/SESSION FOR USERS

*Points: 2 and Status: Completed*

The image shows two hand-drawn mockups for a web application. The left mockup is titled 'TA MANAGEMENT' and 'WELCOME!'. It features a 'LOGIN:' section with input fields for 'USERNAME' and 'PASSWORD', a 'LOGIN' button, and a link that says 'Don't have an account? SIGN UP'. The right mockup is also titled 'TA MANAGEMENT' and features a 'SIGN UP' section with input fields for 'USERNAME', 'PASSWORD', and 'CONFIRM PASSWORD', a 'SIGN UP' button, and a link that says 'Already have an account? Login'.

Figure 9: Mock UI for Signup and Login Page

### 2.5.2 SIGNUP FORM

*Points: 4 and Status: Completed*

Creating new user profiles with credentials an different access levels (TA/ Hiring Manager/ Coordinator/ Professor) can be done using the Signup form.

### 2.5.3 LOGIN FUNCTIONALITY AND UI

*Points: 4 and Status: Completed*

Users of diverse access levels can login into the portal using their correct credentials. De-



The screenshot shows the 'Sign Up' page for the Texas A&M University Engineering portal. The header is dark red with the 'ATM' logo on the left, 'TEXAS A&M UNIVERSITY Engineering' in the center, and a 'Sign Up' button on the right. Below the header, there are links for 'HOME' and 'ALREADY REGISTERED? LOGIN HERE'. The main content area contains a registration form with the following fields: 'Name' (text input), 'Access level' (dropdown menu with 'TA' selected), 'Email' (text input), 'Password' (text input), and 'Confirm password' (text input). A 'Create Account' button is located at the bottom of the form.

Figure 10: Signup Page

pending on the access level of the user, the page will be redirected to a different location.

The screenshot shows the 'Login' page for the Texas A&M University Engineering portal. The header is dark red with the 'ATM' logo on the left, 'TEXAS A&M UNIVERSITY Engineering' in the center, and a 'Login' button on the right. Below the header, there are links for 'HOME' and 'NOT REGISTERED? SIGN UP HERE'. The main content area contains a login form with the following fields: 'Email' (text input) and 'Password' (text input). A 'Login' button is located at the bottom of the form.

Figure 11: Login Page

#### 2.5.4 CSS TO HOME PAGE

*Points: 2 and Status: Completed*

#### 2.5.5 SAME HEADER AND FORM CSS STYLING APPLIED TO ALL OTHER FILES

*Points: 2 and Status: Completed*

The basic layout of all pages is similar, with Texas A&M logo on top-right corner and maroon background. Rest of the page is based on white background.

#### 2.5.6 HOME PAGE DESIGNING AND NAVIGATION

*Points: 2 and Status: Completed*

The home page of the TA application portal, provides 3 buttons to an user; Apply for TA position/ Login/ Signup. User can navigate through the portal according to particular needs.

#### 2.5.7 FLASHES ON ALL VIEWS TO SHOW MESSAGES

*Points: 4 and Status: Completed*

To highlight any message to the user in a better way, feature of flashes is used. This helps in easy user-experience.

#### 2.5.8 CHANGE RAILS LINK TO COLORS AND DESIGN

*Points: 4 and Status: Completed*

To display links on all the pages of portal in a different and more highlighted manner, we have used particular color and design to display links.

#### 2.5.9 NAVIGATION BAR ON ALL PAGES AND ITS CSS

*Points: 4 and Status: Completed*

To allow the user to move between pages in a easy way, a navigation is provided on all pages of the portal. This allows you to move to all accessible pages from the current page.

### 2.6 Admin Page

#### 2.6.1 ADMIN CREDENTIALS AND OTHER CHANGES

*Points: 2 and Status: Completed*

The database was seeded with admin credentials through which he can give a Hiring Manager and Coordinator access to the authorized individuals. This was one form of authentication added to prevent misuse of the application.

### 2.7 Logic Development and Setups

#### 2.7.1 REQUIREMENTS GATHERING

*Points: 24 and Status: Completed*

Details gathering, working on setups, planning documentation and effort estimation. We met with the client in the initial week to understand and gather the requirements and thereby do planning, setups and estimation.

### 2.7.2 CREATE CLASS DIAGRAM FOR THE TA MANAGEMENT SYSTEM

*Points: 2 and Status: Completed*

Following the requirements of client, we designed the needed classes and the data fields in each. This was followed by creating connections between all the present classes.

### 2.7.3 DESIGNING DB SCHEMA

*Points: 4 and Status: Completed*

Following the finalization of class diagram, we designed our database schema based on the requirements of class diagram.

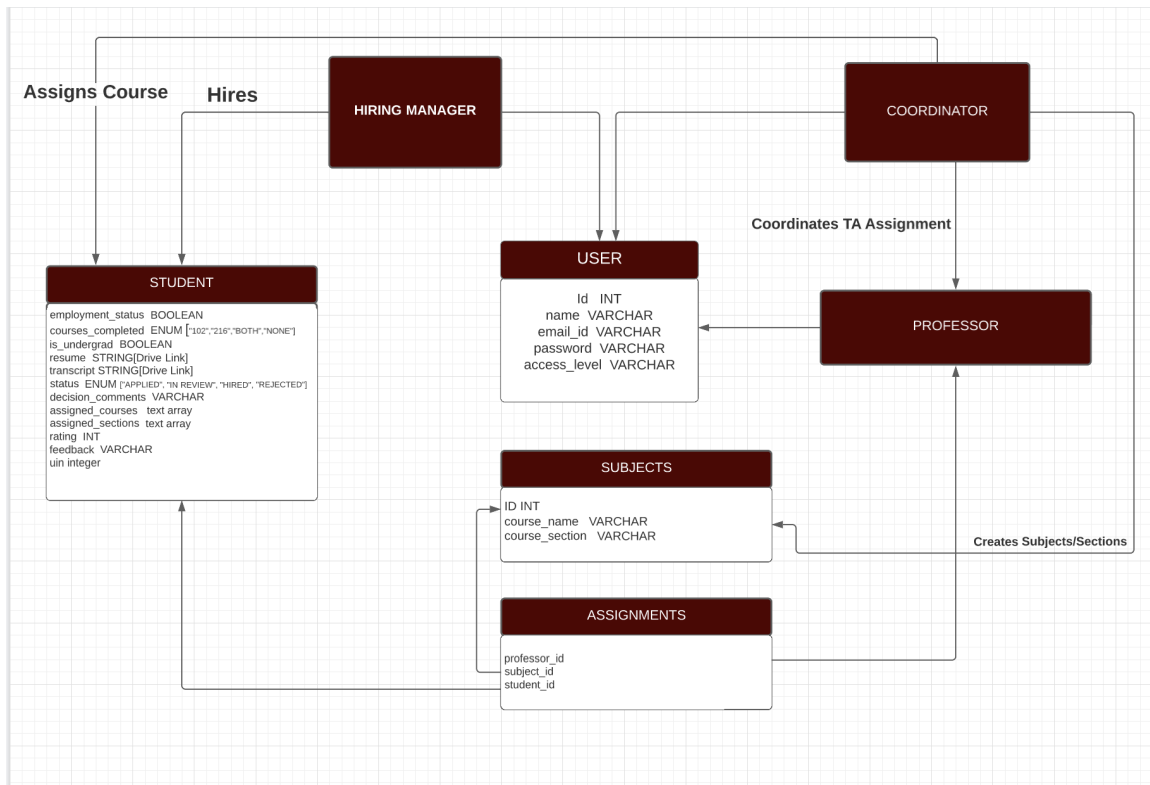


Figure 12: DB Schema

### 2.7.4 IMPLEMENTING AND SETTING UP THE DB SCHEMA IN RAILS

*Points: 8 and Status: Completed*

Having our database schema finalized, we incorporation of Database to our application using PostgreSQL.

### 2.7.5 DEPLOYMENT AND MIGRATION TO PG

*Points: 2 and Status: Completed*

### 2.7.6 DESIGN FOR STUDENT SUBJECT PROFESSOR ASSIGNMENT

*Points: 2 and Status: Completed*

To track Professor-Subject relation and TA-subject relation, so we can feasibly access the professor-student mapping and display the details to the professor.

## 2.8 Testing

### 2.8.1 WRITING CUCUMBER FEATURE FILES

*Points: 4 and Status: Completed*

To make the portal more vulnerable to extreme situations, cucumber tests were written which included successful and exceptional cases. This was performed for all the pages in the application. This gave an initial idea of possible failures and we were able to plan, implement and resolve accordingly.

### 2.8.2 RSPEC FOR CONTROLLERS

*Points: 4 and Status: Completed*

We used Rspec for test-driven development, which involves writing unit and functional tests for code before the code is written itself. These test cases were written for our controllers such as students, users and subjects.

### 3. Team Roles

Iteration	Product Manager	Scrum Master	Developers
0	Suman Kumari	Harmanpreet Singh	Divyansh Bokadia, Rishabh Bassi, Kushagra Jain, Udhav Gupta
1	Divyansh Bokadia	Rishabh Bassi	Harmanpreet Singh, Suman Kumari, Kushagra Jain, Udhav Gupta
2	Kushagra Jain	Udhav Gupta	Harmanpreet Singh, Suman Kumari, Divyansh Bokadia, Rishabh Bassi
3	Rishabh Bassi	Kushagra Jain	Harmanpreet Singh, Suman Kumari, Divyansh Bokadia, Udhav Gupta
4	Suman Kumari	Harmanpreet Singh	Divyansh Bokadia, Rishabh Bassi, Kushagra Jain, Udhav Gupta
5	Suman Kumari	Udhav Gupta	Divyansh Bokadia, Rishabh Bassi, Kushagra Jain, Harmanpreet Singh

We as a team firmly believed in providing all team members with first-hand experience in all roles. This made it easier for us to comprehend the subtleties involved in playing each role. Every member of the team had the opportunity to work on both the frontend and backend aspects of the project and communicate with the client in the capacity of a product owner. Every team member also received the opportunity to play the role of a scrum master in order to hone their project management abilities.

### 4. Scrum Iteration Summary

#### 4.1 Iteration 0

Story Points: 12

The Engineering department's current hiring procedure and system for assigning tasks to teaching assistants (TAs) is quite laborious and requires a lot of human interaction, which causes delays. As a result, our client wants a platform that can speed up the hiring process and facilitate the assignment of TAs (Teaching Assistants) to the various subjects in accordance with their schedule. The software will be utilized by four personas. They are the hiring manager, coordinator, faculty, and applicant. In order to apply for TA positions, the applicant will interact with the application platform. The hiring manager is in charge of setting up interviews and identifying qualified applicants. The coordinator is in charge of allocating TA positions in accordance with the demands of the faculty. The team intends to create software that will let students submit job applications via a straightforward web form. Depending on the profile, the hiring manager may accept or reject the applications. Once accepted, the student can arrange the interview according to the hiring manager's schedule. The manager can update on the portal whether or not a specific student has been hired based on the interview. The coordinator can then assign the hired TAs to the available positions. The faculty can also see the TAs' profiles and get in touch with them as necessary. We worked on gathering the above requirements from the client.

## 4.2 Iteration 1

Story Points: 38

For the TA Management System project's first iteration, we first created the "UML Class Diagram," which allowed us to examine how each class interacted with the others. We also finalized the data-fields and functions in each class. The user story we have selected for this iteration is "Feature: Apply for TA position." The user interface (UX) for the "Student Application form" was created, and it has a number of fields where the student must enter personal information. As a result, a button labeled "Create Student" is available to store a new student's data. When clicked, this button adds the student's data to the existing database. This iteration involved designing the database's schema. The Heroku setup was completed. On the Heroku server, the codebase was connected and deployed.

## 4.3 Iteration 2

Story Points: 48

The "Student Application page," which inputs all the necessary details of the students applying for TA, was the first component of iteration 2 of the TA Management System that we finished. The user is taken to the "Application Submitted" page after submitting the application. The Signup Functionality was the next thing this iteration focused on. The appropriate users can create their profiles on the signup pages for Professor, Hiring Manager, and Coordinator. The hiring manager can view the tabulated data of all the live applications by pulling it from the database. The hiring manager can also use the functionality to search for a specific student by name. We have included the functionality to sort this data based on any data field, including name, email, and UIN. The Hiring Manager can use this feature to access, review, and take appropriate action on all applications. Additionally, we have created R Specs for the Student Controller and Signup features. In order to validate various end case scenarios, Cucumber Test Cases are also deployed for the Signup and Student Application Forms. The scenarios for the test cases are: confirming that the UIN is correct, that is, that it shouldn't contain alphabets, confirming the password and email, etc.

## 4.4 Iteration 3

Story Points: 36

We have a fully functional "Student Application page" in iteration 3 of the TA Management System, which inputs all the necessary information of the students applying for TA with proper and updated inputs for Resume and Transcript. Some of the client's review comments were implemented and advanced in this iteration. The "Hiring Manager View," which allows hiring managers to view all applications in a tabulated format, is the main change made in this iteration. Additionally, the hiring manager can now update the status of a student's application in one of four ways: Applied, Rejected, Review, or Hired. Additionally, he has the option to sort these data based on data fields like name, functionality, and student name searches. The signup pages for Professor, Hiring Manager, and Coordinator, which were implemented in Iteration 2, allow the relevant users to create their profiles. The Login and Logout functionality was also implemented in this iteration. Users such as Professors, Hiring Managers, and TAs can now log in with their access level, perform their

operations, and log out at any time. In addition to the above, we have implemented a significant portion of Coordinator View. This now allows the coordinator to assign specific courses and sections to specific students. After logging in, the Coordinator can also add Courses and Sections.

#### **4.5 Iteration 4**

Story Points: 36

We worked on integrating the login for users with various access levels in iteration 4 of the TA Management System. One can register as a coordinator, hiring manager, professor, or TA. After that, when a user logs in, a particular webpage will be requested based on his access level. The hiring manager will be directed to the page for student applications, the coordinator to the page for assigning courses to hired TAs, where he can divide up the section and course among the hired TAs, and the professor to the page with the assigned TA's information. The views for the "Homepage," "TA application form," "Signup page," "Login page," "Student applications page," and "Assign courses to hired TA page" were also worked on and implemented. We accomplished this using CSS. Additionally, we added the Professor view and its features. The professor can view the TA details for his course and section after logging in, and he can also access the TA's contact information. On this page, you can also use the functionality for submitting the TA review. A new course's professor can be chosen through the professor assigning page, and the professor's information for an already-existing course can be changed.

#### **4.6 Iteration 5**

Story Points: 34

We worked on the implementation of admin. Until admin adds someone as a coordinator or hiring manager, only admin can access the TA system for the first time. The TA system portal would only permit the administrator to log in, so a table was made for the administrator to add the desired people with the required access level. After that, a specific webpage will be requested based on a user's access level when they log in. The Coordinator will be directed to the Assign courses to Hired TA page, where he can distribute the course and section among the hired TAs, the Hiring Manager to the Student applications page, and the Professor to a page displaying the specifics of the assigned TA. Furthermore, we now support pagination for multiple pages that may contain more data and require multiple pages to view the entire data set. We implemented pagination rather than having a single page with 100-200 rows and requiring scrolling down for a better user experience. Additional flash messages have been integrated to display informative messages when the user takes an action. Knowing the outcome of his or her actions will improve the user's overall experience. We also worked on the styling and UI changes for the subject, student, and professor pages. We also made a few bug fixes and minor styling changes. The Professor view's functionality was styled. Views have been improved as a result of UI changes, and screenshots are included below.

## **5. Customer Meeting Summary**

### **5.1 Iteration 0 : 16th September**

In the first meeting with our client, we discussed and understood the expected requirements of the customer from the application. We drafted the basic blueprint of the web-application based on the requirements, and presented it to the client for improvements and approval.

### **5.2 Iteration 1 : 30th September and 7th October**

We presented the formal UML Class Diagram to the client and verified the suggested data-fields and functions. We discussed the user story ‘Apply for TA position’ and presented its user interface, enabling a student to apply by inputting his details.

### **5.3 Iteration 2 : 21st October**

We presented to the client a functional TA application page which accepts the applications from the students and stores the information in the database. We discussed the user story ‘Hiring Manager view’ enabling HM to view all the live applications and perform operations. Further, we presented the signup functionality allowing creation of users of different roles.

### **5.4 Iteration 3 : 4th November**

We presented the Hiring Manager page to the client, allowing him to view and take actions on all the applications. Other features for HM were also presented. We further showed the login-logout functionality and coordinator view in their developing phase. Application page and Signup functionality were presented in their fully functional state.

### **5.5 Iteration 4 : 18th November**

We presented our application with CSS implementation on pages Homepage, TA application page, Signup-Login pages, Applications page, and courses assigning page. Working on the user story ‘Professor view’, we presented it in its developing phase. Webpages where users of diverse access levels are redirected upon logging in were synced and presented.

### **5.6 Iteration 5 : 2nd December**

We presented an additional ‘Admin page’ which allows creation of Hiring Manager and Coordinator user profiles. Further, all pages with CSS implemented were presented, along with additional features such as Pagination, Navigation bar and flash messages. Functional professor’s view was also presented.

## **6. BDD/TDD Process**

For behavioral-driven development (BDD), our project group first followed the general guidelines for generating user stories for Iteration 0 documentation. BDD involved creating user stories, which are descriptions of the features desired by the customer, to show how the application will work. We started by compiling and describing a general list of user stories based on the customer’s initial meeting with us. As the project progressed, the



customer proposed additional user stories or features that we added to the Pivotal Tracker after unanimous agreement by our team members. After completing the ideation process, we also proposed changes to the user interface using low-fidelity diagrams and basic sketches that reflected the combination of different user stories. With the BDD/TDD processes, it becomes clear how a function should work and shouldn't be over complicated while still meeting the required standards, it also helps us think about making the code modular so that each function takes care of only one job. The main advantages of the BDD and TDD process when utilizing Rspec, Cucumber, and the Agile software development methodology were that it kept each group member responsible and aware of the overall tasks that needed to be completed. Along with integrating testing functionality with development, BDD and TDD forced us to include user stories in our iterations that were pertinent to and reflective of what the users actually desired from the application. We used Rspec, a Ruby unit test framework, for test-driven development (TDD), which involves writing unit and functional tests for code before the code is written itself (because of the step definitions for a new story). In order to account for both the common and exceptional circumstances, each Cucumber test included both successful and exceptional cases (happy or sad paths). These tests were written for all our webpages, written in terms of the users requirements, and started with the idea of fail first, and then implement those feature sets accordingly to pass those tests. This helped us implement the various scenarios and make enhancements over the subsequent iterations. Cucumber tests covered the general use cases, such as submitting a TA application, user signing up, and logging in, whereas Rspec test cases were written specifically for the student, subject, and user models.

## **7. Configuration Management Approach**

As a configuration management tool for the project, we employed GitHub. Each team member started a fresh development branch and pushed their code updates there. The patch was then merged into the main branch after raising a pull request. Over the course of five iterations, we pushed close to 60 pull requests and have around 23 branches with 5 releases. Git's branching feature made merging simple and affordable, provided a separate environment for changes related to a particular user story, and promoted Agile development across many team members. Out of five iterations, our project team created releases that were labeled with the appropriate "Iteration #" at the end of each iteration. Future software engineering groups might see how the code was developed during the semester by releasing code at each iteration. We used spikes when it was necessary to orient the team to new software such as getting familiar with Cucumber or thoroughly examine a problem and help divide the work among appropriate team members. For instance, on a few occasions during the project, we had to determine how much work would be required to implement a particular feature or provide everyone involved the chance to become comfortable with a particular API.

## **8. Issues with tools/deployment**

The project began with SQL as the database management system, but when it was deployed on Heroku, we encountered several issues, including Ruby versioning issue, Heroku stack

version mismatch, and Heroku not supporting MySQL. In order to deploy our code on Heroku, we had to migrate from SQLite to PostgreSQL. However, Users of Windows were unable to install the PostgreSQL gem because it required authentication credentials. This was solved by utilizing the Windows On Linux Subsystem (WSL2) and setting up Ruby on Rails on Ubuntu subsystem. So, after resolving these issues and restructuring the project, we were able to deploy it on Heroku, with the automatic deployment feature enabled. The team encountered minor issues during the Ruby installation. On macOS, the team members had conflicting versions of Ruby installed. This was overcome by installing Ruby from scratch using the rvm tool and removing obsolete and default versions that came with the operating system.

## 9. Other Tools

The following additional gems were used in our project:

- Kaminari : Kaminari is a Gem that can be used with Rails to make paginating records super simple.
- Pagy: Pagy splits pagination into the frontend and backend so in order to set our data up correctly on the backend and show it correctly on the front end we need to do a little configuration
- Material\_icon: 'material\_icons' enables you to generate both HTML tags and inline SVG of Material Design Icons for your Ruby on Rails projects.
- Pry: Pry is a powerful tool that Ruby developers can use to debug programs and push past hurdles.
- Rubocop: Rubocop is an incredibly popular Ruby gem for linting Ruby code. With very little configuration, Rubocop scans a repository for violations of the Ruby community's style standards
- Capybara: Capybara -webkit driver (a gem) is used for true headless browser testing with JavaScript support.
- Selenium: Selenium implements the W3C WebDriver protocol to automate popular browsers. It aims to mimic the behaviour of a real user as it interacts with the application's HTML. It's primarily intended for web application testing, but any web-based task can be automated.
- database\_cleaner: database\_cleaner gem for cleaning the database. database\_cleaner is a set of gems containing strategies for cleaning your database in Ruby.

## 10. Repository contents and deployment process

Our repository can be divided into three main categories because Ruby adheres to the Model-View-Controller Architecture: the model is in charge of the application logic, the view is in charge of rendering content, and the controller establishes a connection between the model and the user (user requests). We have one of these for each of our various use cases, and we gave them names accordingly (students, subjects, users, managements, professors). Additionally, we have files for our imports, which are also a component of the Ruby architecture and are known as GemFiles. The Cucumber testing files that were used to test

our features are also available. The documentation folder, which contains all six of our iteration reports, is the last item. On our Github repository, we have also included a link to our Heroku deployment. There is no additional script needed to deploy as we have added automatic deployment to our repository.

## **11. Important Links**

- Pivotal Tracker: <https://www.pivotaltracker.com/n/projects/2596948>
- GitHub : <https://github.com/tamu-edu-students/ta-management>
- Heroku : <https://ta-management-tamu.herokuapp.com>
- Project and Presentation Video link: TODO