Hayley Carter

Contributors

Adriano Cavalcanti : Advisor  
Hayley Carter, Ryan Perkins, Guzel Noori, Nisser Aldossary : Developers

University Portal

Capstone Project Report

# Acknowledgements

I would like to acknowledge my supervisor for this project, Professor Adriano Cavalcanti, PhD., for his guidance on the development of this project. I would also like to acknowledge Professor Arne Leitert, whose foundational courses were vital to the success of this project.

I would like to extend my gratitude to my friends, Aundrea Kiviloo, Meghan Mac, Rasmus Tindborg, and Brendan Adam for their contributions in keeping me sane throughout both this quarter and this quarantine. I would like to extend special thanks to Ray Michel, who was able to offer me technical and professional advice all throughout the project that served to keep me grounded in professional reality despite the stresses I faced. Finally, I would like to extend my utmost gratitude and appreciation for Jarrett Hatcher for the complete and unwavering support he gave me during the best and the worst times of this quarter. I don’t know where I’d be without him.

Table of Contents

[Acknowledgements 1](#_Toc66215694)

[Table of Figures 3](#_Toc66215695)

[The Project 4](#_Toc66215696)

[1.1 Introduction 4](#_Toc66215697)

[1.2 Application 4](#_Toc66215698)

[1.3 Technologies 5](#_Toc66215699)

[1.4 Motivation 5](#_Toc66215700)

[Implementation and Design 6](#_Toc66215701)

[2.1 Introduction 6](#_Toc66215702)

[2.2 Technologies 6](#_Toc66215703)

[2.3 Application 8](#_Toc66215704)

[2.4 Agile Development 10](#_Toc66215705)

[2.5 Testing 12](#_Toc66215706)

[Conclusion 13](#_Toc66215707)

[Appendix 14](#_Toc66215708)

[iAcademic User Manual 14](#_Toc66215709)

[Objective 14](#_Toc66215710)

[Instructions 14](#_Toc66215711)

[References 23](#_Toc66215712)

# Table of Figures

[Figure 1: Technologies used in this project including HTML, CSS, Java, and MySQL. 5](#_Toc66215624)

[Figure 2: The Google Cloud Platform logo. 6](#_Toc66215625)

[Figure 3: Overview of the structure of Java Servlets. 8](#_Toc66215626)

[Figure 4: Relational Model for our database 10](#_Toc66215627)

[Figure 5: A diagram of the Agile life cycle 11](#_Toc66215628)

[Figure 6: Gantt Chart for our project. 12](#_Toc66215629)

[Figure 7: Our unit tests, and the fact that they have passed. 13](#_Toc66215630)

[Figure 8: Platform selection. 14](#_Toc66215631)

[Figure 9: Sign-in page. 15](#_Toc66215632)

[Figure 10: Sign in page with an example of what a student login might look like. 16](#_Toc66215633)

[Figure 11:Class search view. 17](#_Toc66215634)

[Figure 12: Search results view 17](#_Toc66215635)

[Figure 13: Student schedule and class drop view. 18](#_Toc66215636)

[Figure 14:Faculty login with an example of what that would look like. 18](#_Toc66215637)

[Figure 15:Faculty schedule view. 19](#_Toc66215638)

[Figure 16: Course select view for instructors. 19](#_Toc66215639)

[Figure 17:Manual student add for instructors 20](#_Toc66215640)

[Figure 18: Course select view for instructors. 20](#_Toc66215641)

[Figure 19: Students in a given course as shown to the instructor of that course. 21](#_Toc66215642)

[Figure 20: Course select view for instructors. 21](#_Toc66215643)

[Figure 21: Instructors grading view. 22](#_Toc66215644)

[Figure 22: Course history view for students. 22](#_Toc66215645)

# The Project

## Introduction

The iAcademic University Portal project is a tool for university students and teachers designed to accommodate course enrollment and course history. Teachers and students alike may use the tool for various course enrollment-related tasks. The project was developed for the Google Cloud Platform to allow the developers the chance to work with such tools and technologies, seeing as they are highly relevant to the current job market [1].

## 1.2 Application



Figure 1: Technologies used in this project including HTML, CSS, Java, and MySQL.

The iAcademic University Portal project is composed of a Java application running on Google App Engine, a Google Cloud service for cloud-based applications. The front end of the application is comprised of HTML 5, CSS 3, JavaScript, and Bootstrap so that the application is useable from any device. The backend is written in Java SE Runtime 8 using a MySQL database for data storage. Due to the technologies used, this project is highly scalable for user needs. The program serves a webpage front end for users to interact with.

## Technologies



Figure 2: The Google Cloud Platform logo.

The Google Cloud platform is made of physical components like computers and hard drives as well as virtual components like virtual machines. Together, they create a system which allows resources to be allocated to applications based on their runtime and data needs [2]. This allows Google Cloud applications to be highly scalable, and as the user needs change, so too can the application. A developer can opt into only the services and components they need to provide their users with a smooth application experience.

The Google Cloud platform serves our Java application and front end and allocates resources based on the apps needs as well as the specifications we provided in our appengineweb.xml file in our program. This allowed us to request resources based on anticipated needs, and allows for the application to obtain more resources if needed. Additionally, Google Cloud has many security features built into the platform, including SSL certification and SQL injection protections.

## 1.4 Motivation

The intent behind this project was to learn how to work with cloud computing tools and services to gain marketable skills for our future careers. We chose to use the Google Cloud Platform to facilitate this intent, as it is a common cloud platform for development in the professional world. By working with tools that professionals are using to develop, we will be able to apply those skills in the workforce.

# Implementation and Design

## 2.1 Introduction

The Java back end was developed and tested on Google Cloud Platform Standard Environment using the Eclipse IDE and Java SE 8, as is required by the Google Cloud Platform. It uses libraries which allow it to use Java servlets and Java Server pages. The Java application includes Java Unit Tests for relevant methods. All libraries used are open source and acknowledged in the software where applicable.

The HTML front end was developed and tested initially using Visual Studios and tested further on deployment to the Google Cloud Platform Standard Environment. The HTML is supplemented by CSS, JavaScript, and Bootstrap to enable to website to function well on all platforms.

## 2.2 Technologies

The back end of this project is comprised of Java Servlets working with both HTML and Java Server Pages. A servlet is a Java program that runs on the program’s web or application server which can handle requests from the server. It receives a request from the server, processes that request, produces a response, and sends that response back to the server, as shown below in Figure 2.

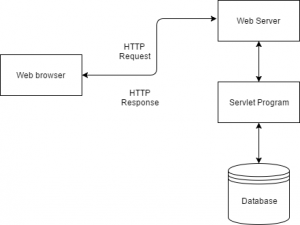


Figure 3: Overview of the structure of Java Servlets.

The servlets work in conjunction with other Java methods to deliver the majority of the website’s functionality. Servlets are portable and efficient, and they are able to share data more effectively than the common alternative of using Common Gateway Interface (CGI). Servlets are able to directly communicate with servers in a way that CGI cannot [3]. Where a servlet could not be used, a Java Server Page was implemented.

A Java Server Page is an HTML document with java code written directly into it using the <% %> tags as demonstrated in Figure 3 below.



Figure 3: Overview of the structure of Java Server Pages.

They allow for data retrieval during the loading of an HTML page, which was required for the display of schedules in our application. The advantage of using JSPs is that they communicate very easily and cleanly with Java servlets and other Java backend components.

## 2.3 Application

Most backend functionality requires a connection to the database, which is handled by our DatabaseController class. The DatabaseController object handles all functionality related to the database to allow for control over what methods have access to what. Both servlets and JSPs are able to use DatabaseController objects for their needs. When a user makes a request for information, the DatabaseController gets the information it needs from the servlet or JSP and runs the appropriate queries.

The DatabaseController accesses our database, which stored information relating to students, instructors, and courses. It is used for almost all of the application’s functionality, as much of what the application does is information retrieval and manipulation.

The structure of the database is shown in the ER diagram below.

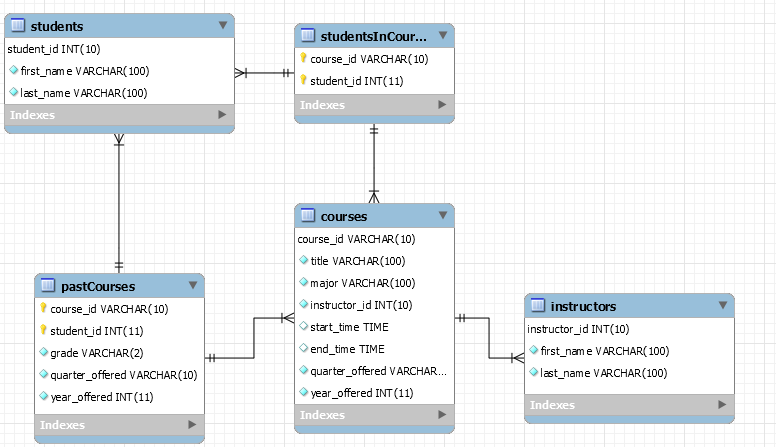


Figure : Relational Model for our database

Student and instructor information is pulled from the database and stored in a User object that the program makes use of to load information related to the specific student or instructor from the various course-related tables. When adding or dropping a course, the program uses the User object to determine whose schedule it needs to manipulate, and then goes from there to the DatabaseController object to be used to perform the correct database manipulations. The same is true when an instructor adds or drops a student to a course, and when they finalize grades.

Users interact with the application through the HTML front end, which also includes JavaScript components and Bootstrap components to facilitate use by desktop and mobile users.

## 2.4 Agile Development



Figure : A diagram of the Agile life cycle

Agile is a software development methodology which promotes coordinated and flexible task planning and management, following principles of agility. The Agile development life cycle is pictured above, and consists of requirements drafting, designing, developing, testing, and deploying. The process is broken up into sprints, and each sprint generally covers a small component or feature of the overall applications. A project is made up of as many sprints as are needed to address the feature backlog and launch a completed product, and the cycle can be repeated after launch for additional updates and maintenance.

Sprints consist of daily scum meetings to manage a set number of features to be developed in a set amount of time. Daily scrum meetings help keep developers on a schedule and allow them space to ask for assistance if they are struggling with an implementation.

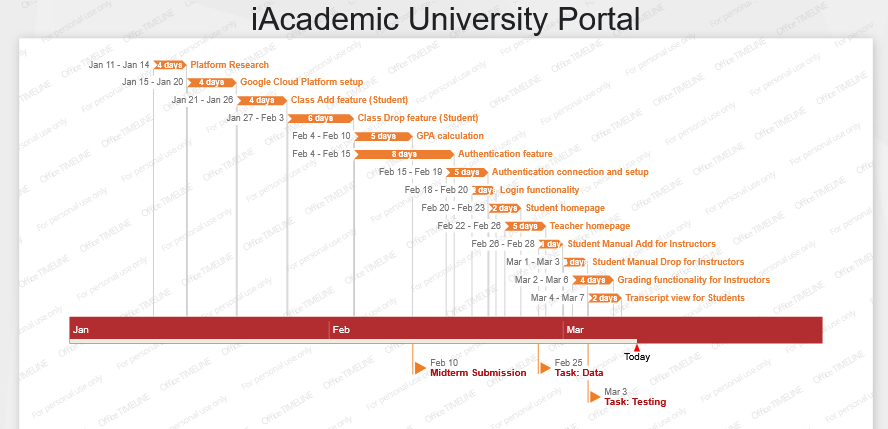


Figure : Gantt Chart for our project.

The iAcademic project was broken into sprints which generally covered multiple features, with the Authentication system being the only exception. Each sprint lasted a week, and the timeframes in which components were completed is shown in the Gantt Chart above. We found that many features took longer than initially thought due to difficulties working with the Google App Engine Environment and associated tools.

## 2.5 Testing

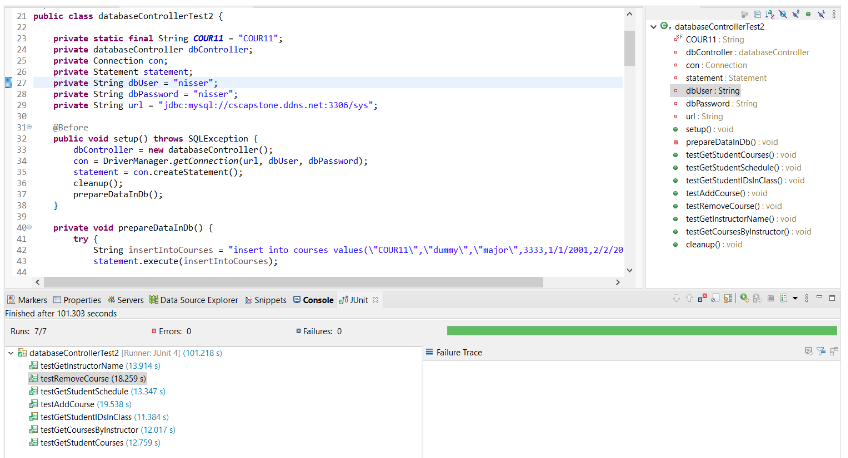


Figure : Our unit tests, and the fact that they have passed.

As the backend functionality of our app relies almost entirely on our DatabaseController class, we did rigorous unit testing of that class to ensure that it was working correctly. Pictured above is evidence of the unit tests passing. Outside of unit testing, a large amount of manual testing occurred of individual features during development. The front end components especially received intense manual testing all throughout the development process to ensure that everything functioned as intended and was intuitive to understand and use for any user.

# Conclusion

Over the course of the project, the most consistently difficult aspect was working with Google App Engine. The setup required to run the development environment was tedious and poorly-explained by Google’s available tutorials, making it necessary to seek outside help. For example, Google App Engine’s tutorial for set-up fails to mention that Google App Engine requires use of Java SE 8 as opposed to more current versions of Java, which lead to a number of odd errors that were difficult to track down and solve.

Working on the project taught us a great deal about working with Java servlets, and by the end of the project, the knowledge we gained helped us to see how our initial design decisions were not as simple as they initially appeared. By the end of the project, tasks that used to take several days were complete-able in one or two. This knowledge also inspired me to continue work on a personal project that would benefit from use of servlets and JSPs.

If we were to do this project again, we would start with the database structure and build components based on it, rather than creating the database to fit an initial prototype. This would have saved us a lot of work we had to spend in redesigning the database structure to accommodate later features.

# Appendix

## iAcademic User Manual

### Objective

This instruction manual is to navigate the user through the web pages to properly use the functionalities delivered in the website.

### Instructions

Step-by-step instructions for each page

1. The main page is to navigate the user to the right platform - a student or a faculty platform. Additionally to the platform choices in the center of the page, a user can also choose a platform option through the navigation bar on top next to the “Home” tab - “Choose platform”. There are only two platforms available which are Student and Faculty.

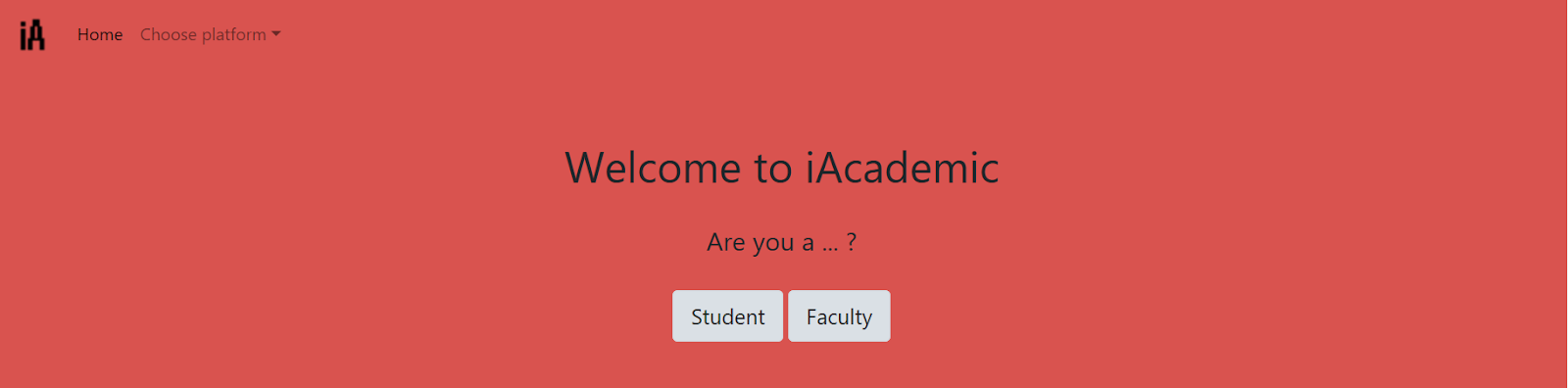


Figure : Platform selection.

1. According to the choice the user chose, the website will redirect the user to the authentication/sign in page. There are two inputs acceptable such as username and password. Username is the University ID of a student or faculty. The usernames already exist in the database, therefore there is no need to create an account or register for this web platform. Sign in page is shown below

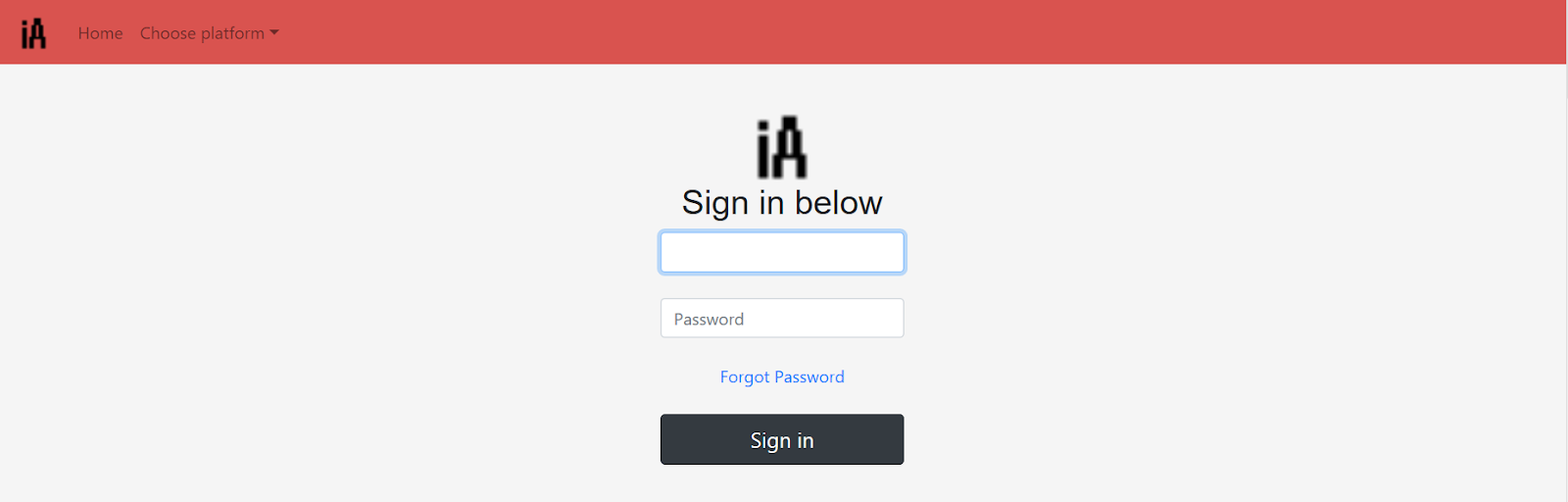


Figure : Sign-in page.

1. As you can see below in the Sign In page is the Student Login information.

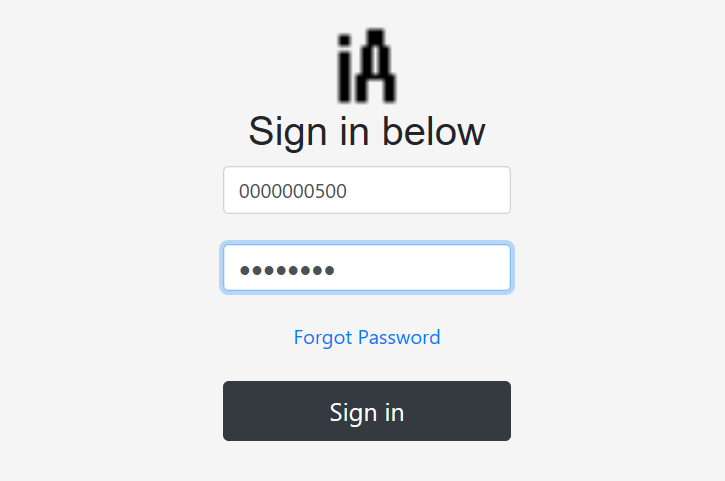


Figure : Sign in page with an example of what a student login might look like.

1. Below is the Student portal where students are able to search for classes. Users should at least fill one of the input boxes to narrow down the search. As an example shown below, the Subject name was entered which is Computer Science. There are a lot of subjects that can be entered such as History, Geography, Chemistry etc. And it is important to choose which term you are trying to enroll in. Choose a term and a year to narrow down the search and enroll for the right class and in the right term.

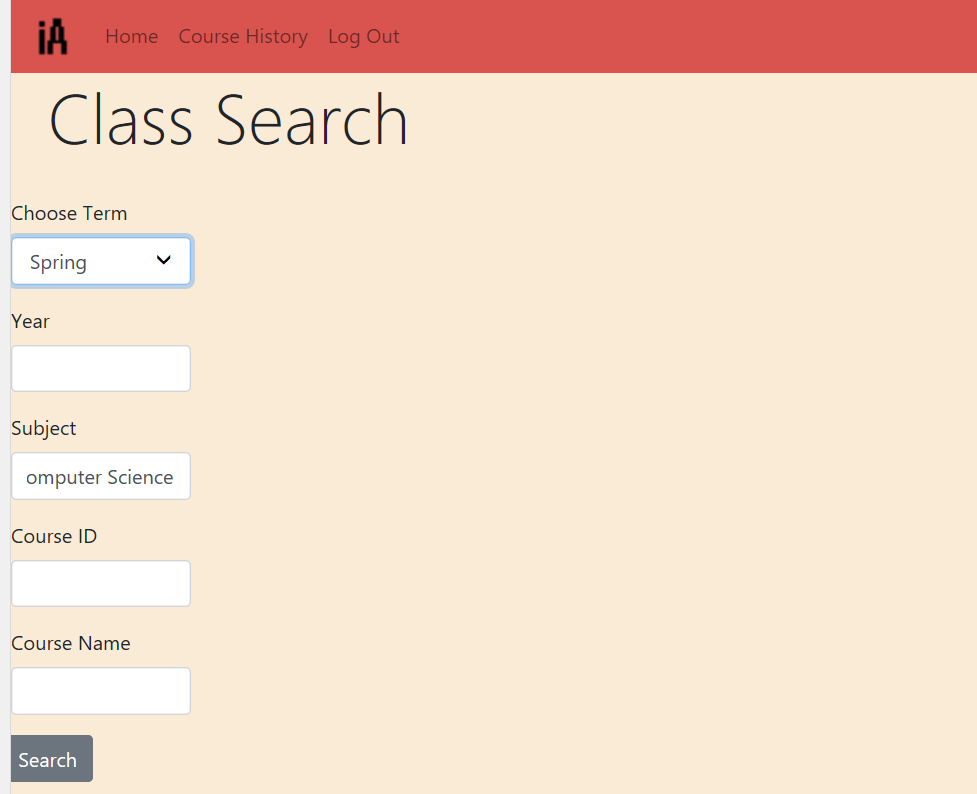


Figure :Class search view.

1. Below are the search results according to the criterias that student input. The page below shows only one Computer Science class available for Spring 2023. On the very first volume on the left, there is a checkbox available from the search result so that the students get to choose which classes to enroll in. Check the checkbox and click Enroll so that the class shows in the students Class Schedule.



Figure : Search results view

1. Below is the student’s class schedule where the student has the option of dropping the classes the same way the student added the class by using the checkbox.



Figure : Student schedule and class drop view.

1. Below is an example of Faculty login. Username is a University ID given to the Faculty and Password is encrypted.

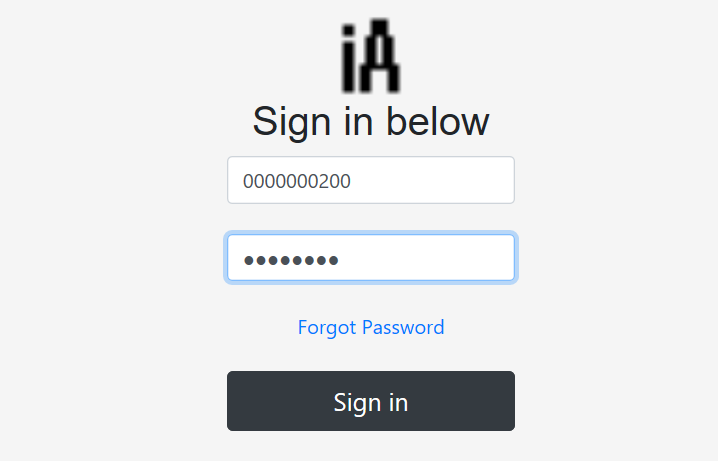


Figure :Faculty login with an example of what that would look like.

1. Below you can see the Class schedule for the faculty that signed in.



Figure :Faculty schedule view.

1. The faculty has the permission to add a student to the class manually. There is an Add Student Tab available on the navigation bar of the faculty platform. Once clicked the page will redirect the faculty to the course selection to which the faculty is trying to enroll the student in.

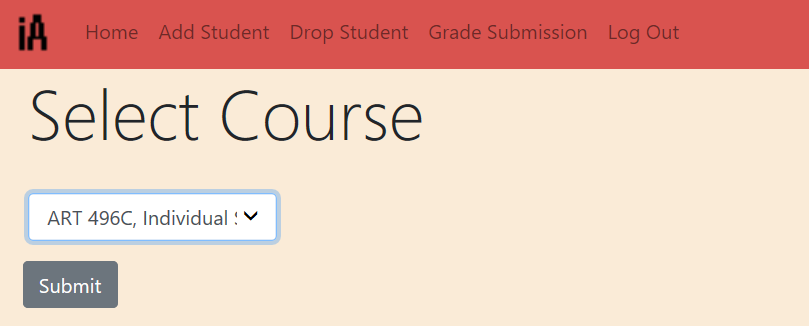


Figure : Course selection view for instructors.

1. Once the Submit button is clicked, the faculty is redirected to the another page to enter the student information that the faculty is trying to enroll the student in

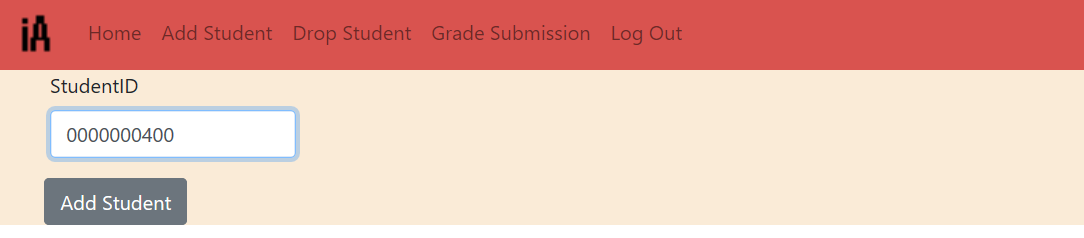


Figure :Manual student add for instructors

Once the Add Student button is clicked the student shows in the list of the students for the faculty, as well as the class is showing up in the students course schedule. The same way as the Add Student, there is an option of dropping the student from the class from the faculty’s end.

1. Below is the page that lets the faculty choose the class to drop the student from.

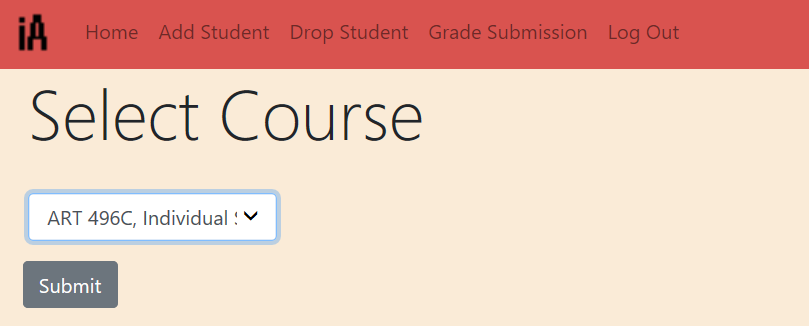


Figure : Course select view for instructors.

1. Once the Submit button is clicked, there will be a list of students available in this case only one. There is a check box available to choose the number of students and drop them from the class.

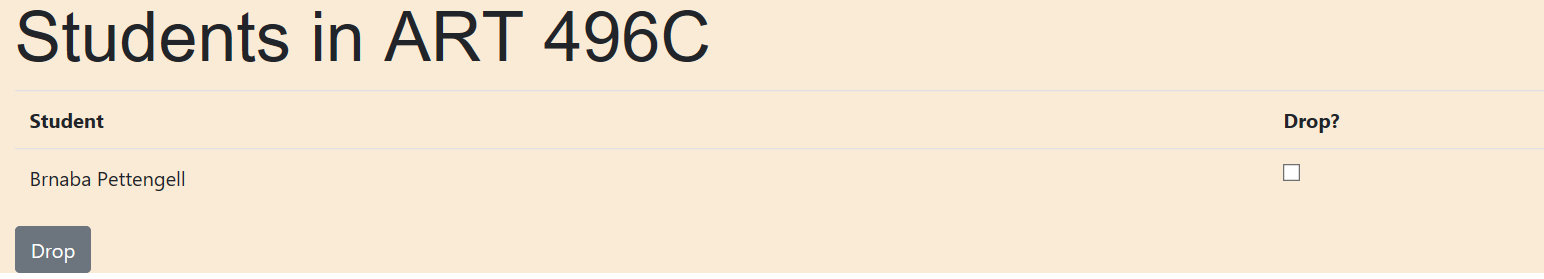


Figure : Students in a given course as shown to the instructor of that course.

1. The faculty can enter a grade for the student. First the class needs to be chosen.

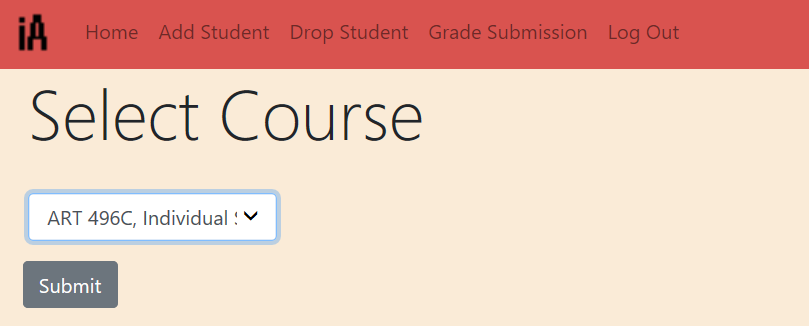


Figure : Course Select view for instructors.

1. Select a grade from the dropdown list. Press the Submit button

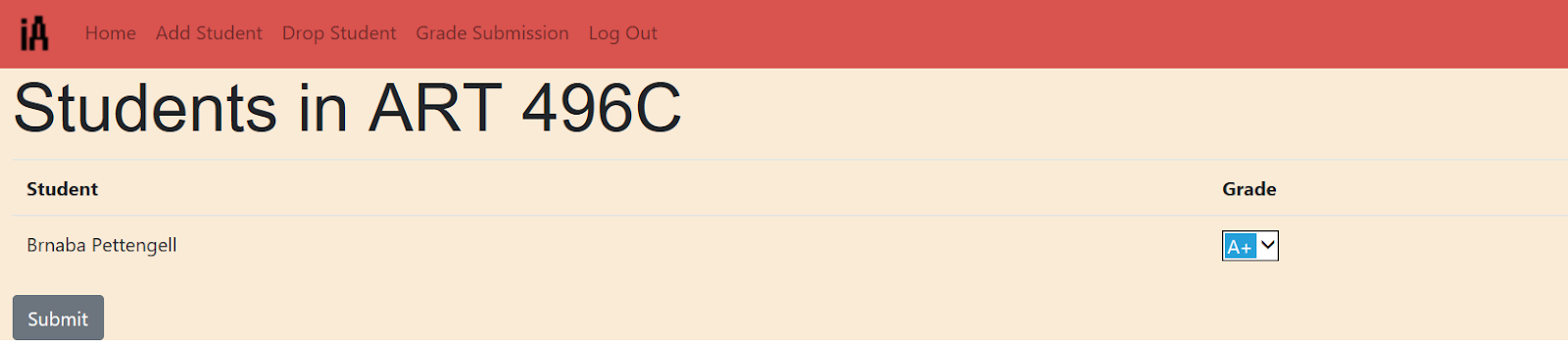


Figure : Instructors grading view.

1. Below is the Course history that is available in the Student Platform. Course history shows the classes that the student took along with the grade for each class and total GPA. Course History is available on the navigation bar at the top of the page.

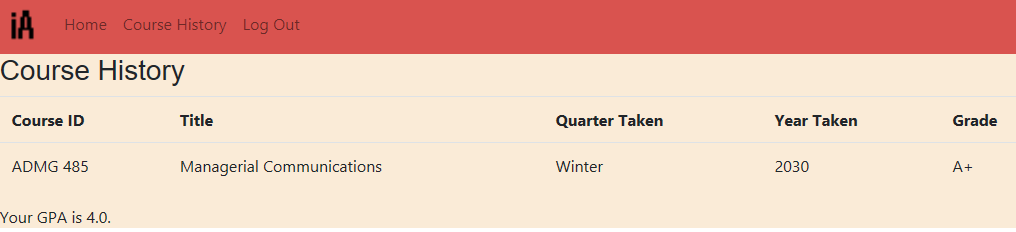


Figure : Course history view for students.

# 

# References

[1] Cirrus, Admin - Open. “Why Is Cloud Computing Important?” *Open Cirrus*, 14 Sept. 2018, opencirrus.org/cloud-computing-important/.

[2] “Google Cloud Overview  |  Overview.” *Google Cloud Overview*, Google, cloud.google.com/docs/overview.

[3] “Introduction to Java Servlets.” *GeeksforGeeks*, 23 Oct. 2019, www.geeksforgeeks.org/introduction-java-servlets/.