



University Discourse

CS 30700 (Spring 2020)

Team 1

Design Document

Drishti Agarwala

Mohana Chandiran Dhukkaram

Tae Yoon Kim

Rewati Shitole

Index

- **Purpose**
 - Functional Requirements
 - Non Functional Requirements
- **Design Outline**
 - High Level Overview
 - Sequence of Events Overview
 - Overall Sequence Diagram
- **Design Issues**
 - Functional Issues
 - Non Functional Issues
- **Design Details**
 - Class Design
 - Class Description
 - Sequence Diagram
 - Navigation Flow Map
- **UI Mockups**

Purpose

In the current university setting, there is no integrated platform that addresses all of the specific academic concerns which students have [Academic concerns include but are not limited to grade estimation, schedule feedback, course review, and course recommendation]. Other sources (such as Reddit/r/Purdue and numerous discord groups) are not always appropriate, because they do not strictly serve the purpose of answering academic queries, and are also meant for providing solutions to issues which are not academic in nature.

Through our product, we intend to assist students in finding solutions to the above-mentioned problems via a unified and focused platform. It will be a web application, which will provide users with functionalities, like searching for a specific course, posting their plan of study, asking for suggestions on it, rating a certain course, getting a grade estimation based on the past semester curves and following academic content they are interested in. We believe that this product will save students the hassle of exploring various apps by serving their purpose holistically.

Functional Requirements

1. User Account

As a user,

- a) I want to be able to create an account
- b) If time allows, I want to be able to report inappropriate content
- c) If time allows, I want to be able to have two-factor authentication

As a registered user,

- a) I want to be able to login to my account
- b) I want to be able to change my credentials
- c) I want to be able to reset my password if I forget it
- d) I want to be able to logout of my account
- e) I want to be able to delete my account

2. Feed

As a registered user,

- a) I want to be able to follow tags
- b) I want to be able to view the posts from the tags I follow on my feed

As a user,

- a) If time allows, I want to be able to view the top posts on my feed.

3. Posting Content

As a registered user,

- a) I want to be able to post text
- b) I want to be able to post images
- c) I want to be able to upload documents
- d) I want to be able to post my schedule in an interactive calendar form
- e) I want to be able to share posts as an embeddable link

4. [Searching/Viewing Posts](#)

As a user,

- a) I want to be able to search for posts and tags
- b) I want to be able to view all posts and comments
- c) I want to be able to filter my search based on University, Course, Subject or Professor(tags)

5. [Rating](#)

As a user,

- a) I want to be able to view course ratings based on professors and course difficulty

As a registered user,

- a) I want to be able to rate the course based on professor and difficulty

6. [Commenting and Editing](#)

As a registered user,

- a) I want to be able to comment on posts
- b) I want to be able to comment on comments
- c) I want to be able to edit the post
- d) I want to be able to edit the comments
- e) I want to be able to upvote and downvote the posts
- f) I want to be able to upvote and downvote the comments

7. [Curve Estimation](#)

As a user,

- a) I want to be able to view past semester curves of courses.

As a registered user,

- a) I want to be able to input my scores and past semester curves for a course

Non Functional Requirements

1) Client Requirements

As a developer,

- a) I would want the application to be able to run on a web browser
- b) I would want the application to be user-friendly and easy to understand.

2) Server Requirements

As a developer,

- a) I would like the server to be able to handle the requests from the clients correctly and as required.
- b) I would like the server to be able to query the database if the client's request needs it.
- c) I would like the server to be able to send the result of the query to the appropriate client under a given time limit.

3) Performance Requirements

As a developer,

- a) I would like the application to be able to handle enough users simultaneously.
- b) I would like the application's response time to be 5 seconds.
- c) I would like the application to be able to handle the errors well

4) Appearance Requirements

As a developer,

- a) I would like the User Interface to be very user-friendly and aesthetic

- b) I would like the user to be able to input their schedules in an interactive manner.

5) Security Requirements

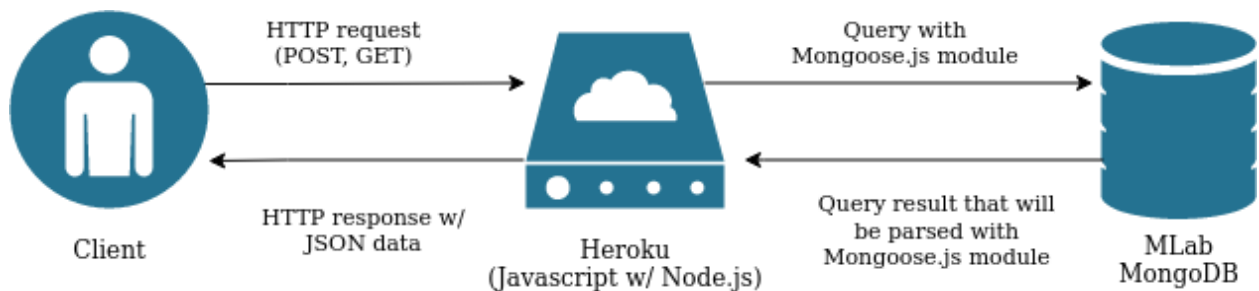
As a developer,

- a) (If time allows)I would like to store user's data in a protected manner
- b) I would like to have a unique username associated with one email id to prevent people from making multiple accounts.

Design Outline

High-Level Overview

The product will be a web application that allows users to post and search for courses, professors, universities, and more. This application will use the client-server model in which one server handles every access from any arbitrary number of users using the express module for the Node.js server. The server will accept client requests, query the database if necessary, and respond to the user's request accordingly.



1. Client

- The client provides the user with an interface powered by react.
- The client sends HTTP requests (GET, POST, PUT, PATCH, REMOVE) to the server for creating an account, posts or fetching data.
- The client receives and parses the JSON data response from the server and changes the interface accordingly.

2. Server

- Server (using Node.js) is hosted using Heroku
- The server receives and handles HTTP requests from clients.

- c. The server sends queries to the database (if necessary) to create, modify, remove or retrieve the appropriate entries.
- d. The server parses the result sent from the database and sends it in response to the client.

3. Database

- a. Database stores all data used in the web application, such as but not limited to: user information, user credentials, posts (courses, professors, universities, labs, notes, and general documents), comments (comments on a post and threads), course summaries, course ratings, curve estimation data, and tags.
- b. The database responds to queries from the server and sends the appropriate data back to the server if needed.

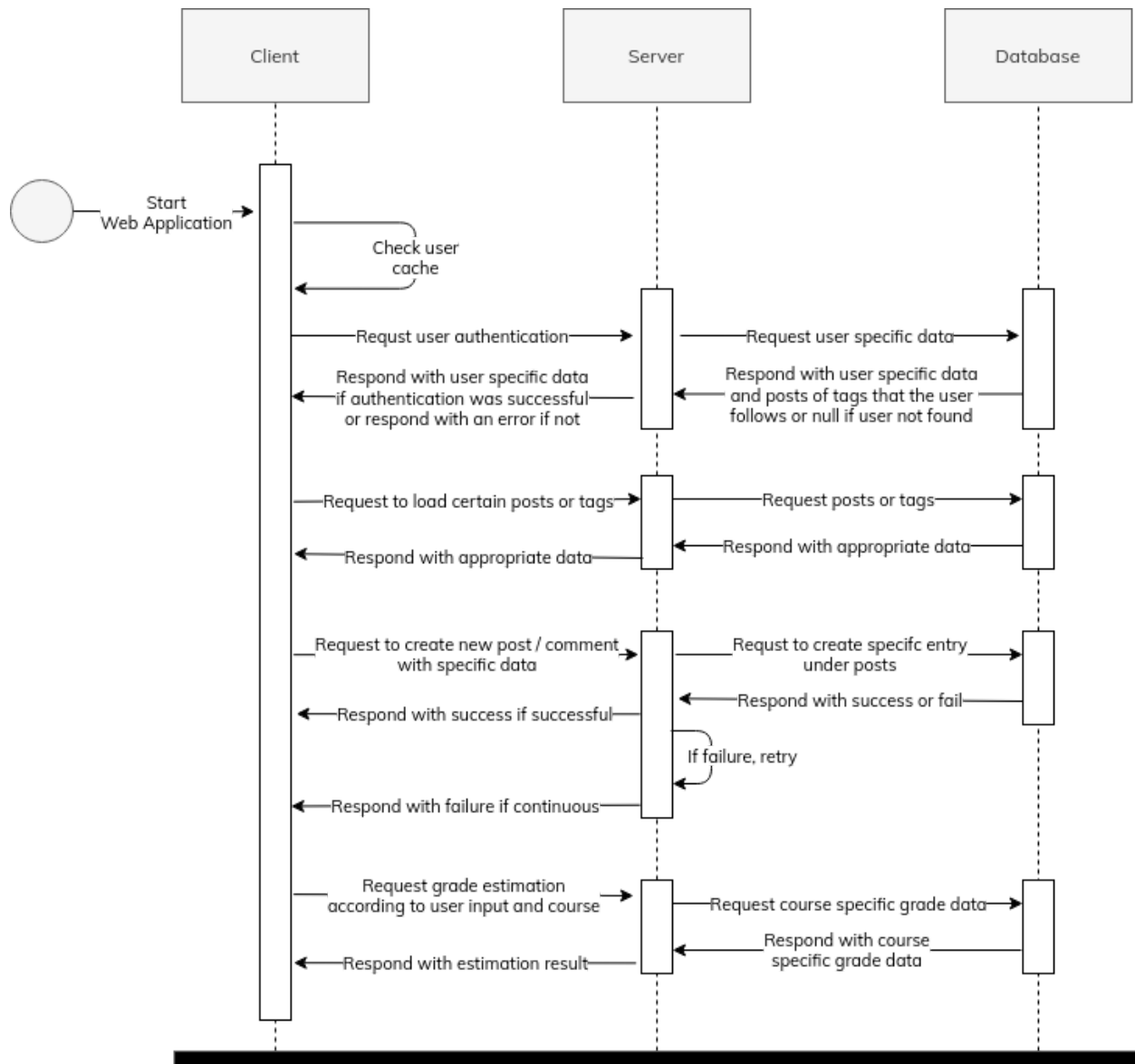
Sequence of Events Overview

The sequence diagram below depicts the interaction between clients, server and databases. When the user starts the application, the client checks the cache to see if it can execute an auto-login. If it fails, the user is asked to enter the username and the password. The login request is then sent to the server by the client. The server handles the request and forwards a query to the database. If the credentials were legitimate, the database responds to the request with the tags the user follows and the posts from those tags. If the login was unsuccessful an error is returned to the client and the user is asked to re-enter the login information. After logging in, the user can carry out several tasks like searching(for a tag or a post), creating a new post, commenting on posts or estimating curve.

Firstly, when the user searches for a tag or post, the client sends a request to the server which sends a query command to the database, requesting the appropriate data. The database responds with the requested data to the client through the server.

Secondly, to make a new post or make a comment, the client sends a request to the server with the required data of the post. The server then sends a query command to the database to create the entry. The database responds to the server with success or failure. If the database failed to create the entry, the server will send another query command to the database to try again. In case of a successful entry, the server responds to the client with success. However, in case of continuous failure, the server will respond to the client with a failure.

Overall Sequence Diagram



Design Issues

Functional Issues

- 1) Who can view and modify content(make posts, upvote, downvote, comment)

Option 1: Anyone(with or without the account)

Option 2: Only a user with an account can view or modify

Option 3: A user with an account can view as well as modify while a user without the account can only view it.

Decision: Option 3

Explanation: We made this decision as we wanted everyone to reap benefit from the resources this product provides but at the same time we do not want everyone to be able to make changes as that might be unwanted sometime. Hence only the dedicated users with a registered account can modify the available content.

- 2) What information do we need from the registered users to sign up for an account?

Option 1: Username and password

Option 2: Username, password, and email

Option 3: Username, password, email, first name, last name, phone

Decision: Option 2

Explanation: We made this decision as we do not require the user's name or phone number for providing any of the available features. On the other hand we request for email id because it is required for contacting the user when security issues might arise, like password recovery, or security alerts. Every email should be allowed to have a unique username for security reasons so that a single user doesn't make multiple accounts.

3) How can the user search for a post or a tag?

Option 1: Only be able to view the suggested tags related to their search in the drop-down

Option 2: Only be able to view the suggested posts related to their search in the drop-down

Option 3: Be able to view both posts and tags related to their search in the dropdown.

Decision: Option 1

Explanation: We made this decision because the implementation of Option 3 was intricate and we were not confident about its completion, on the other hand, Option 2 would have prevented the users from being able to search for tags. Option 1 is suitable as it allows them to search for specific tags and view all the posts related to it.

4) How can we calculate the rating of the classes?

Option 1: Just allow the users to enter one overall rating and calculate the average of all user ratings.

Option 2: Let the users fill a class rating form which would ask them to rate the class based on difficulty and professor and calculate their individual ratings.

Decision: Option 2

Explanation: We made this decision because after analyzing the existing similar services available in the market, we realized that there are not many options available for getting a holistic rating. Hence this form feature will allow us to provide a more descriptive and categorized rating.

5) How should we ask the users to input their schedules to ask for opinions?

Option 1: As an image

Option 2: In an interactive way where the user viewing can click on them and view basic details(credits and professor) of the course.

Decision: Option 2

Explanation: This platform feature was intended to be a complete package for getting recommended the best schedules and the relevant information about the courses. If the schedules are interactive, it allows the students to quickly get more information about the respective classes, and click on the tag to go to its course page.

Non Functional Issues

1) What language and framework should we use for our frontend?

Option 1: React.js

Option 2: Angular.js

Option 3: Vue.js

Decision: Option 1

Explanation: Currently, React.js is the most popular frontend and it is easy to learn compared to Angular.js and it is maintained by a renowned company.

2) What language and framework should we use for our backend?

Option 1: Node.js

Option 2: PHP

Option 3: Java

Decision: Option 1

Explanation: As similar to the explanation given before, Node.js is very popular and it is easy to learn and deploy projects as soon as one knows Javascript. Although PHP is easy to learn, it seems that the trend is shifting.

3) What database should we use to store information?

Option 1: MySQL

Option 2: MongoDB

Option 3: PostgreSQL

Decision: Option 2

Explanation: Since the project will have data structures that will contain many different key-value pairs (tags) and documents (posts), we decided to go with

MongoDB as document databases store objects that go well with Node. Also, MongoDB allows remote cloud service called MLab.

4) What cloud hosting service should we use?

Option 1: None (Local)

Option 2: Heroku

Option 3: AWS

Decision: Option 2

Explanation: We definitely would like to use this opportunity to get some experience on remote hosting services, so local hosting is not an option.

Comparing between Heroku and AWS, Heroku is more beginner friendly and suitable for projects.

5) How should we store user passwords?

Option 1: Plain text

Option 2: Hashed passwords

Option 3: Hashed passwords with salt

Decision: Option 3

Explanation: Storing the password in plain text is something no one should be doing. Hashing passwords are good but hashing the passwords that have been added salt is even better as it prevents danger to the users even if the hacker gets the password with the salt. Salt will be stored in the environment variable of the server where no one will have remote access to it.

6) What kind of hashing algorithm should we use?

Option 1: SHA-1

Option 2: SHA-2

Option 3: MD5

Decision: Option 2

Explanation: According to the previous experiences of programmers, it is believed that MD5 and SHA-1 is no longer a suitable candidate as a way to securely store sensitive information.

7) How should we store comments of a post in the database?

Option 1: Store all the comments in a collection and store its unique id in the post object and fetch it every time we display the post.

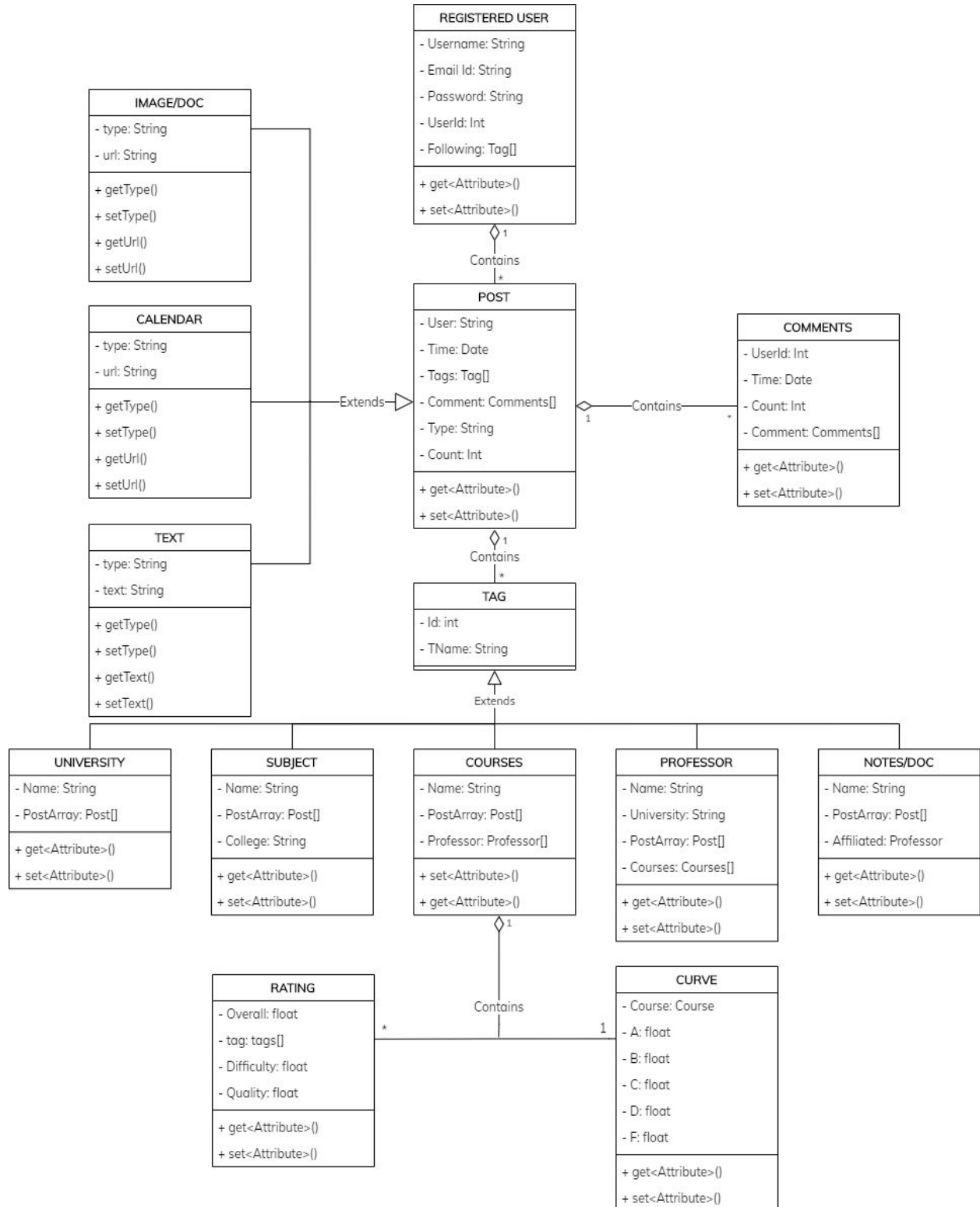
Option 2: Store the comments as objects in the post object.

Decision: Option 2

Explanation: It reduces the number of times we query the database and additionally removes the need for two different collections in the MongoDB database.

Design Details

Class Design



Class Description

Registered User

- An object of this class is created when a user creates an re account.
- Each user will be assigned a unique user ID.
- Each user will have to enter a username and a password to login.

Post

- Every post will consist of:
 1. The name of the user who posted it
 2. The time when they posted it
 3. Tags associated to the post
 4. Comments on the post
 5. The count of the votes (upvotes + downvotes)
- The post can be of multiple types: image, document, text, and calendar which will be stored in a string variable

Image/doc

- It consists of a field to tell us the file type of the image or document to be uploaded.
- It also consists of the URL which indicates the source of the image or document.

Calendar

- The calendar is an interactive mapping of the plan of study. The courses are linked as tags and so are hyperlinks to their respective course pages.
- It has a type field to determine the type of the post(Image/Doc/Calendar/Text)

Text

- The text post consists of the text that it will contain along with the properties of a regular post.
- It has a type field to determine the type of the post(Image/Doc/Calendar/Text)

Comments

- Peers can comment on a post. Individual comments can be upvoted and can also be commented on to form a thread.
- Every comment will consist of:
 1. The name of the user who posted it/Userid
 2. The time when they posted it
 3. Comments on the comment
 4. The count of the votes (upvotes + downvotes)

Tag

- Every tag will consist of a unique tag id and a tag name to uniquely identify the different types.
- The tag can be of multiple types: university, subject, courses, professor, notes/doc.

University

- University tag is a type of tag that stores university name in its name attribute.
- It also has an attribute PostArray[] which consists of all the arrays which are associated with this tag.

Courses

- Subject tag is a type of tag that stores the course name in its name attribute.
- It has an attribute PostArray[] which consists of all the arrays which are associated with this tag.
- It also has an attribute of Professor type indicating the list of professors teaching the course.

Subject

- University tag is a type of tag that stores the subject name in its name attribute.
- It has an attribute PostArray[] which consists of all the arrays which are associated with this tag.

- It also has a String variable indicating the College under which the subject falls.

Professor

- Professor tag is a type of tag that stores the professor name in its name attribute.
- It has an attribute PostArray[] which consists of all the arrays which are associated with this tag.
- It also has an attribute of Course type which indicates the courses taught by him.

Notes/doc

- The notes/doc tag is a type of tag that has the name of the course the notes/documents are relevant to.
- It also has an attribute PostArray[] which consists of all the arrays which are associated with this tag.
- Lastly it has an attribute of Professor type which indicates the professor to whose class the notes belong.

Rating

- The rating class will be used to rate different courses.
- It consists of an array of instances of Tag which will store a list of all the tags related to that course.
- A float variable called overall will be used to calculate the overall rating of the course.
- A float variable called difficulty will be used to calculate the average difficulty rating of the course
- A float variable called quality will be used to calculate the average quality rating of the course.

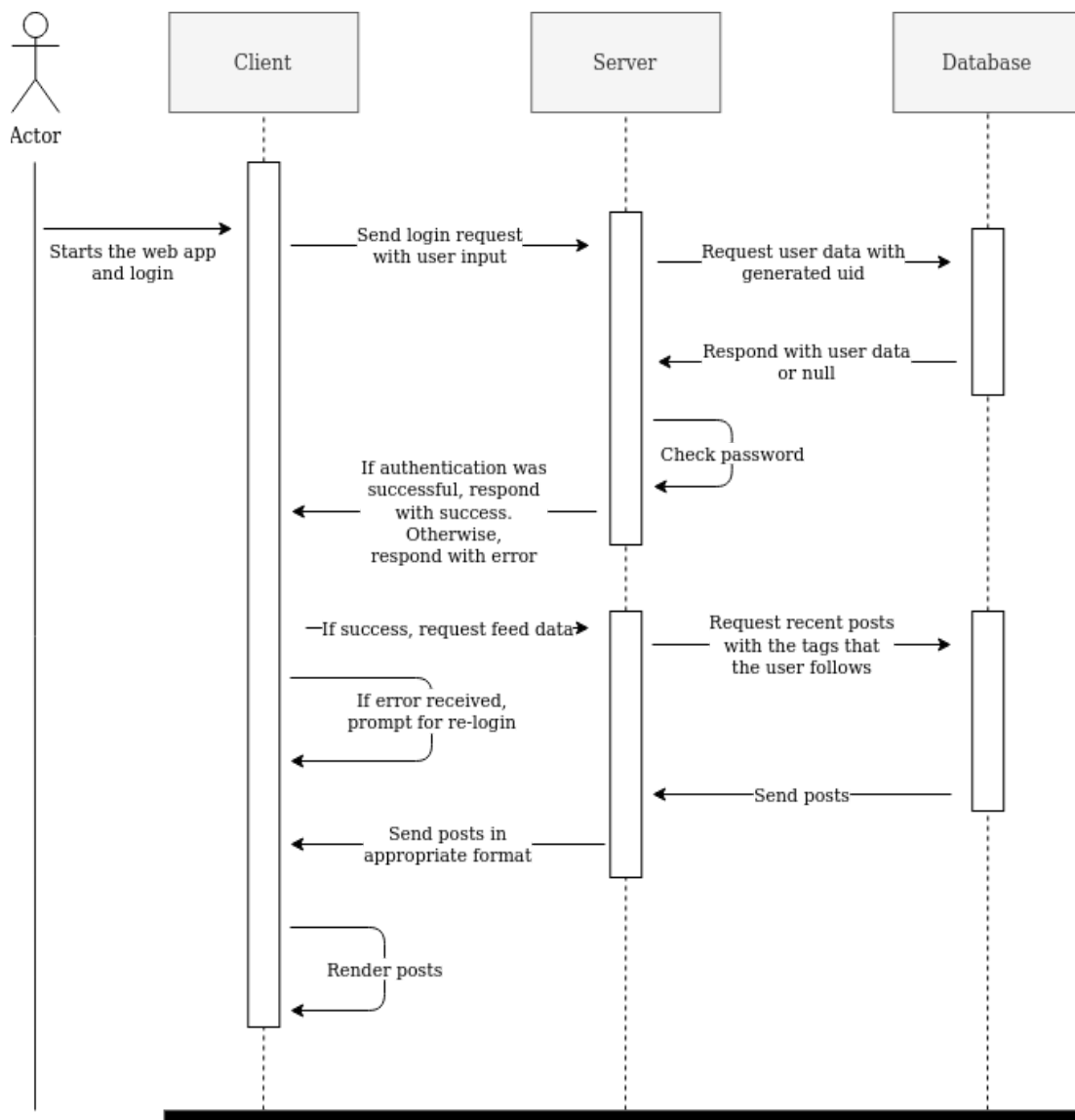
Curve

- It consists of an instance of Course class which will represent the course for which the curve is being calculated.

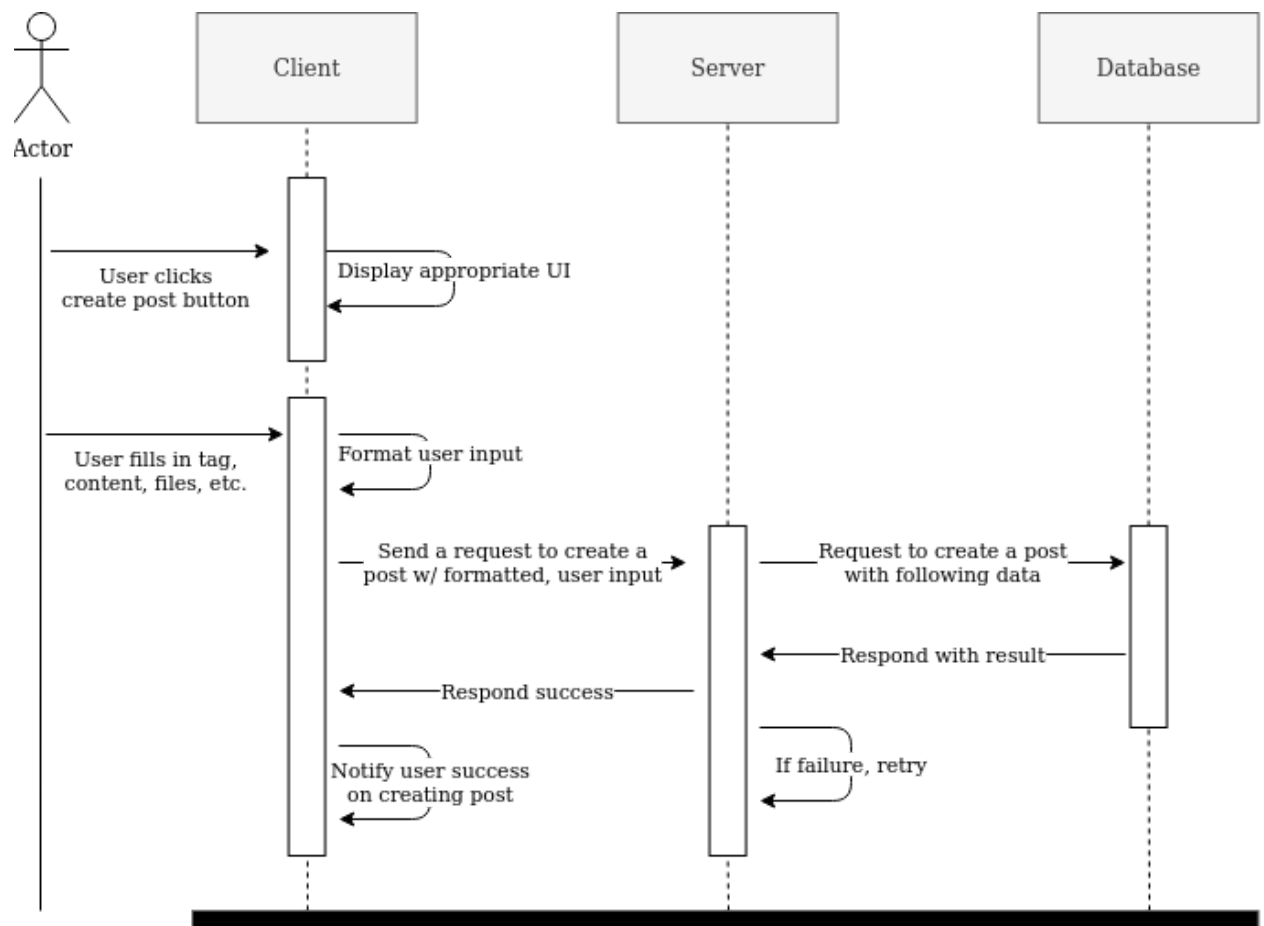
- It has float variables which will be used to calculate the ranges for different grades(A/B/C/D/F)

Detailed Sequence Diagrams

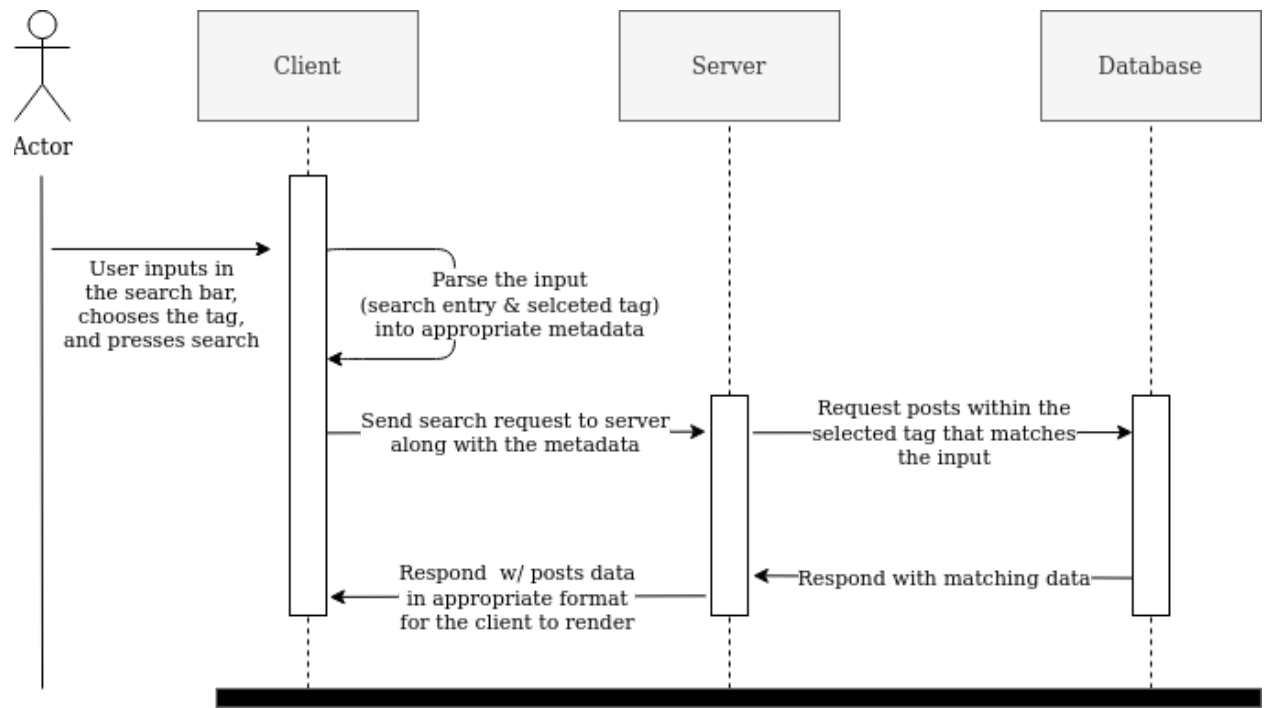
- **Login / Register:** The client sends a login request with the user input to the server. The server generates a user id and sends it to the database. The database responds to the server with user data or null if the data is not found. The server checks the password and responds with success or failure. In case of failure, a re-login is requested. Otherwise, the request for posts and user tags is sent to the database and it responds to the server with the appropriate posts and tags which is then sent to the client.



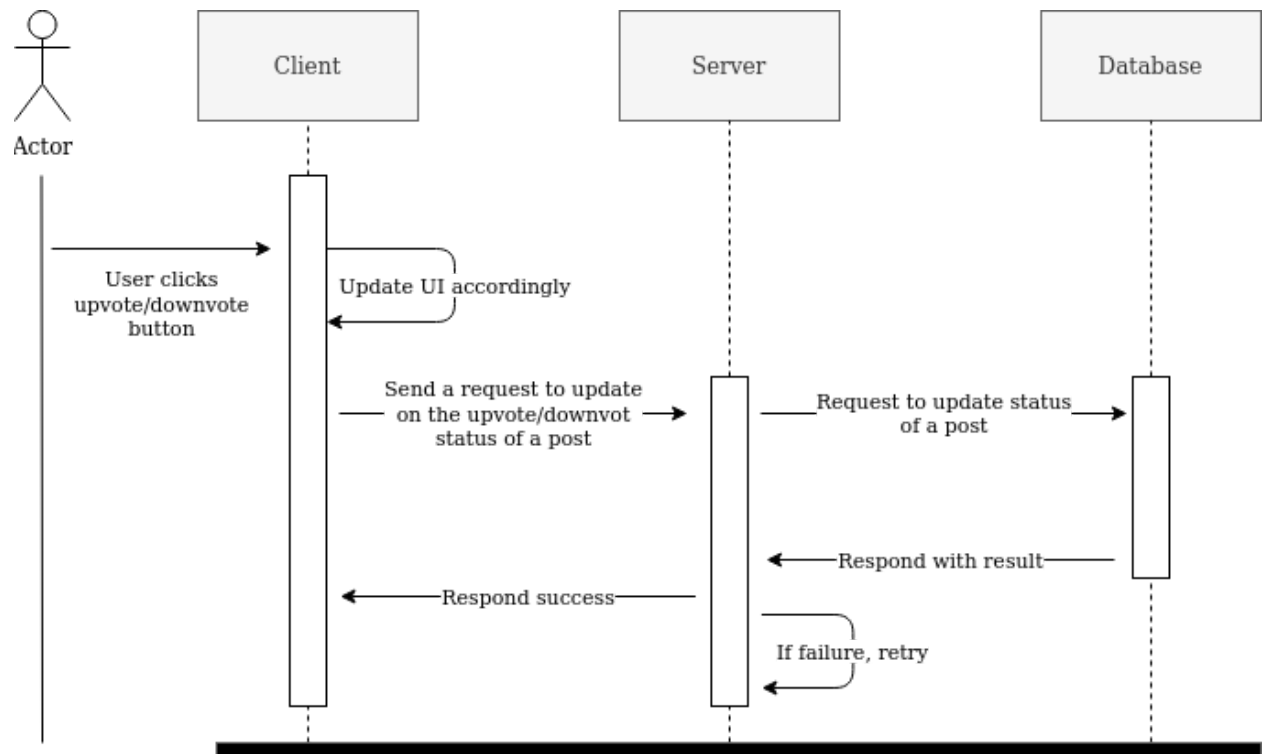
- **Create post:** The user clicks on the create post button. The client displays the appropriate UI. The user then fills in tags and content. The client formats the user input and sends a request to the server to create a post with user input. The server sends a query command to the database and the database responds with the result. If it responds with a failure, the server sends a request to retry. If the database responds with success, the server sends the post to the client and the client notifies the user with success on creating the post.



- **Search for a post or a tag:** The user searches for a tag or a keyword in the search bar. The client parses the input into appropriate metadata and sends it to the server. The server requests for posts with the selected tag from the database that matches the input. The database responds with the matching data to the server and the server responds with posts in the appropriate format for the client to render.



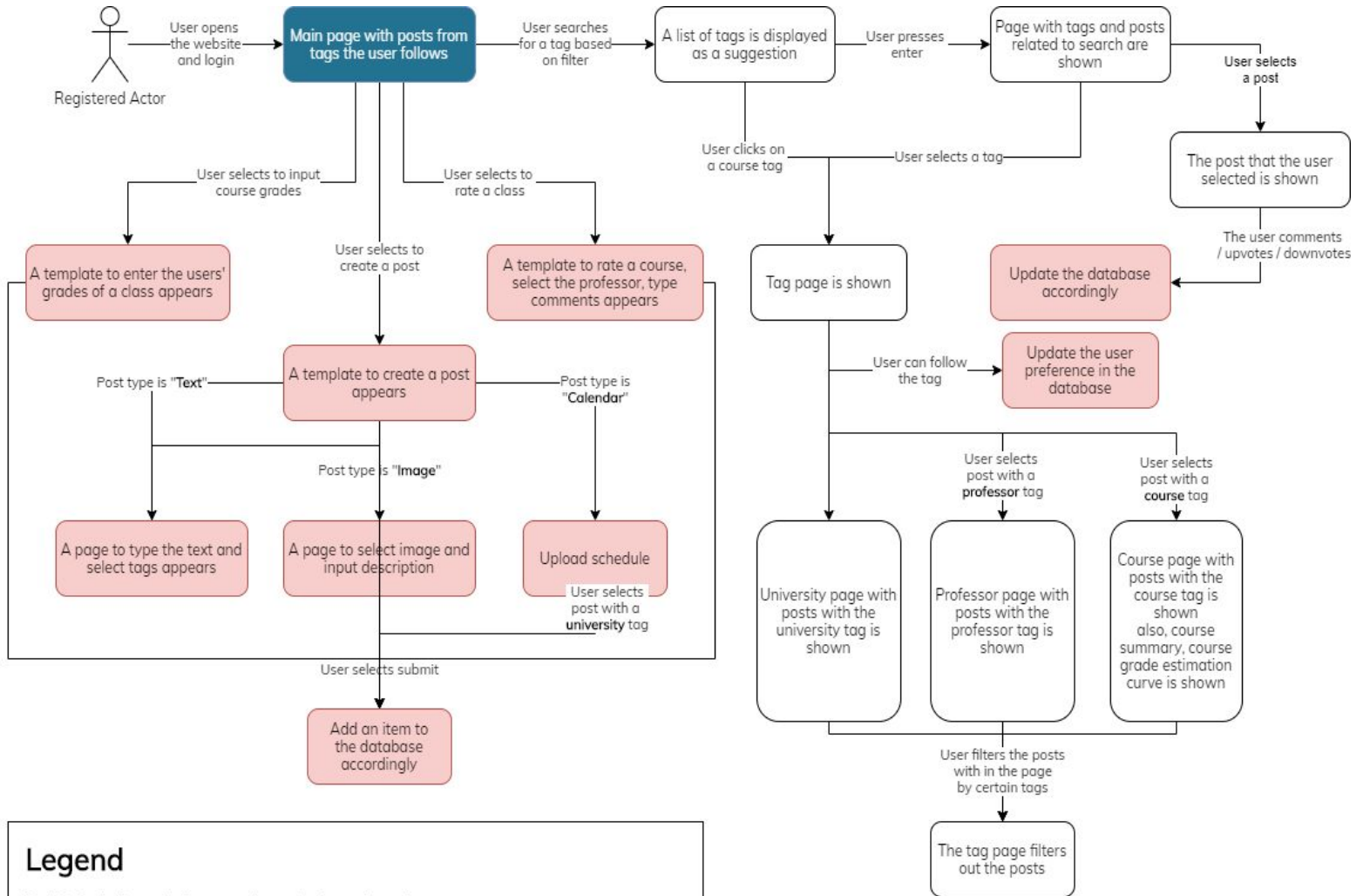
- **Upvote / Downvote a post:** The user clicks the upvote/down button. The client updates the UI to account for the upvote or downvote. The client sends a request to update the upvote/downvote status of the post to the server. The server sends a query command to the database. The database responds with success or failure. In case of a failure, the server tries again. In case of a success the server responds to the client with a success.



Navigation Flow Map

The activity diagram that follows shows the flow of our web app. When the user first opens the web app, they can view their personal feed which consists of posts from the tags/topics they follow. They can then select an activity out of the four available features, that includes being able to search for a certain tag based on a filter, being able to fill out a rating form for a course they have already taken, being able to create a post or being able to feed in data for the curve estimation feature. However, for the last three activities they will have to create an account or log in to their existing accounts. If the user opts to create a new post it can be of multiple types: image/doc/text/schedule. Once the user clicks on one of the options they are asked for the post info, that post is made and sent to the database.

ACTIVITY DIAGRAM



Legend

Red blocks indicate the features where only the registered users can access.

Blue blocks indicate the features where the unregistered and the registered users differ in what is shown on screen. For the main page, the registered user is shown posts of tags that the registered user follows. For the unregistered user, the most liked (upvoted) posts are shown.

UI MOCKUP

LOG IN / REGISTER PAGE: This page will pop up when users will try to modify the content, like create a post, write up a comment or upvote downvote other content. As for performing these activities, they need to be a registered user.



University Discourse

LOG IN

REGISTER



University Discourse intends to assist students in finding solutions to their academic concerns.



EXPLORE LINKS


About
Services
Education
Work for us

CONTACT

321 Pikes Place Parkway
West Lafayette, IN 47906
(555) 765-4321
info@company.com
Mon - Sat: 7:00 am - 6:00 pm

LOG IN / SIGN UP: These 4 pages allow the user to login / sign up

Search



University Discourse


Log in

jane@example.com





LOG IN

Forgot Username

Forgot Password

University Discourse

University Discourse intends to assist students in finding solutions to their academic concerns.



EXPLORE LINKS

About

Services

Education

Work for us

CONTACT

321 Pike Place Parkway
West Lafayette, IN 47906

(555) 765-4321


info@company.com

Mon - Sat: 7:00 am - 6:00 pm

Toggle to view your data or an error, connector adapting account

© 2018 Company Privacy Policy Terms & Conditions

Search



University Discourse


Recover your account

jane@example.com





EMAIL ME

If you are having trouble accessing your account, follow this link.

LOGIN SIGN UP

University Discourse

University Discourse intends to assist students in finding solutions to their academic concerns.



EXPLORE LINKS

About

Services

Education

Work for us

CONTACT

321 Pike Place Parkway
West Lafayette, IN 47906

(555) 765-4321


info@company.com

Mon - Sat: 7:00 am - 6:00 pm

Toggle to view your data or an error, connector adapting account

© 2018 Company Privacy Policy Terms & Conditions

Search



University Discourse


Register

the_jane@gmail.com





NEXT

Already a member? Log in

By signing up, you agree to Privacy's Terms of Service and Privacy Policy.

University Discourse

University Discourse intends to assist students in finding solutions to their academic concerns.



EXPLORE LINKS

About

Services

Education

Work for us

CONTACT

321 Pike Place Parkway
West Lafayette, IN 47906

(555) 765-4321


info@company.com

Mon - Sat: 7:00 am - 6:00 pm

Toggle to view your data or an error, connector adapting account

© 2018 Company Privacy Policy Terms & Conditions

Search



University Discourse


Register

Choose a username





Password

SIGN UP

By signing up, you agree to Privacy's Terms of Service and Privacy Policy.

University Discourse

University Discourse intends to assist students in finding solutions to their academic concerns.



EXPLORE LINKS

About

Services

Education

Work for us

CONTACT

321 Pike Place Parkway
West Lafayette, IN 47906

(555) 765-4321

info@company.com

Mon - Sat: 7:00 am - 6:00 pm

Toggle to view your data or an error, connector adapting account

© 2018 Company Privacy Policy Terms & Conditions

USER FEED: This is the feed of a registered user, which shows posts from the topics they follow and the feature which allows them to create posts of different types.

Q Search

MAKE A POST

TEXT

IMAGE

DOCUMENT

CALENDER

LINK

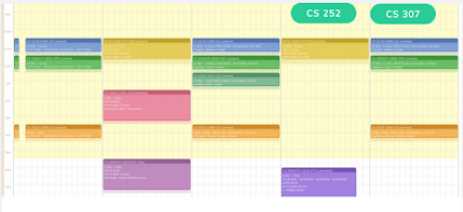
H

Fall Semester Schedule (CS Sophomore)

February 28, 2019

CS 252

CS 307



Hey guys, I just wanted to know if this schedule looks good for the upcoming Fall semester as a CS Sophomore. Any and every suggestions would be more than welcome!

♥

↗

⌵

R

Fall Semester 250 Lab

February 28, 2019

CS 250

I would suggest CS Sophomores not to take up a late evening class as the 250 lab would be demanding, and might require you to work in evenings.

♥

↗

⌵

Comments

Tags

What are your thoughts?

COMMENT >

H

February 28, 2019

Thank You so much for the suggestion! I was just making my schedule.

♥

↗

⌵

R

Spring Semester Calc 2

February 28, 2019

MA 162

Hey everyone! Would you suggest doing MA 162 in the spring with Dr. Dolittle or with Robert Downey?

♥

↗

⌵

Follow Tags

CS 252

CS 307

Econ 251

MGMT

University Discourse

University Discourse intends to assist students in finding solutions to their academic concerns.

f

t

i

y

EXPLORE LINKS

About

Services

Education

Work for us

CONTACT

321 Pikes Place Parkway
West Lafayette, IN 47906

(555) 765-4321

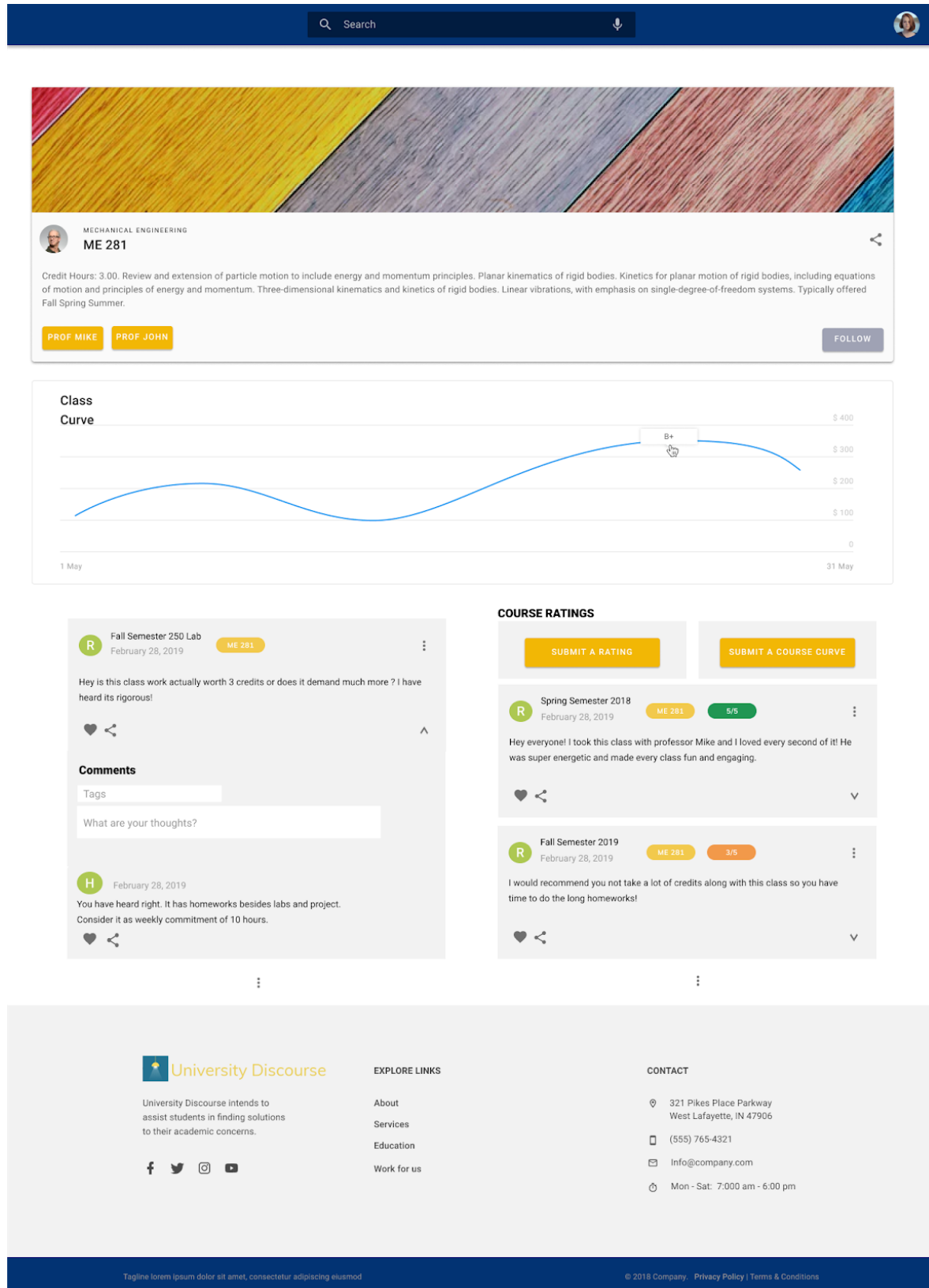
Info@company.com

Mon - Sat: 7:00 am - 6:00 pm

Tagline lorem ipsum dolor sit amet, consectetur adipiscing eiusmod

© 2018 Company. Privacy Policy | Terms & Conditions

COURSE PAGE: This is the course page that pops after the user searches for a specific course. It displays the course summary, its estimated curve based on past semesters, course ratings, rating input form, curve input-form, and relevant posts with the course tag.











SEARCH RESULTS: This is the screen that is displayed after the user enters a search input. It will consist of all the posts (along with their relevant tags and creator's name) which contain the search keyword in their post title. Eg: homework in this UI.

Homework

Search Results

Sort

Filter

Post Details	User Name	Date	Tags
<div><div>ECON 251 is very hard, need help with homework Updated 1 day ago</div></div>	<div>Tom Cruise</div> <div>on 24.05.2019</div>	<div>May 26, 2019</div> <div>6:30 PM</div>	<div>ECON 251</div> <div></div>
<div><div>Calc 3 is very hard for me, can you make a groupme for homework Updated 1 day ago</div></div>	<div>Matt Damon</div> <div>on 24.05.2019</div>	<div>May 26, 2019</div> <div>8:00 AM</div>	<div>MA 261</div> <div></div>
<div><div>Why is blackboard not letting me submit on homework Updated 1 day ago</div></div>	<div>Robert Downey</div> <div>on 24.05.2019</div>	<div>May 26, 2019</div> <div>7:30 PM</div>	<div>GENERAL</div> <div></div>
<div><div>Homework will not be this hard, I promise - I'm a TA for ME Updated 2 days ago</div></div>	<div>Christian Bale</div> <div>on 24.05.2019</div>	<div>May 25, 2019</div> <div>5:00 PM</div>	<div>ME 110</div> <div></div>
<div><div>Does anyone have homework for MGMT 201 Chapter 2? Updated 2 days ago</div></div>	<div>Henry Cavill</div> <div>on 24.05.2019</div>	<div>May 25, 2019</div> <div>4:00 PM</div>	<div>ME 110</div> <div></div>
<div><div>Did the homework for MGMT 200 get graded? Updated 3 days ago</div></div>	<div>Chris Evans</div> <div>on 23.05.2019</div>	<div>May 25, 2019</div> <div>2:00 PM</div>	<div>MGMT 1</div> <div></div>
<div><div>Homework 2, CS 240 from Fall 2018 Updated 4 days ago</div></div>	<div>Sam Smith</div> <div>on 22.05.2019</div>	<div>May 25, 2019</div> <div>11:30 AM</div>	<div>MGMT 1</div> <div></div>
<div><div>I have doubt in Homework 10, CS 252 (Q 3) Updated 6 days ago</div></div>	<div>Steve Rogers</div> <div>on 21.05.2019</div>	<div>May 24, 2019</div> <div>1:00 PM</div>	<div>NORMAL</div> <div></div>


Rows per page:

8

1-8 of 1240

<

>

University Discourse

University Discourse intends to assist students in finding solutions to their academic concerns.

f

t

ig

y

EXPLORE LINKS

About

Services

Education

Work for us

CONTACT

321 Pikes Place Parkway
West Lafayette, IN 47906

(555) 765-4321

Info@company.com

Mon - Sat: 7:00 am - 6:00 pm

Tagline lorem ipsum dolor sit amet, consectetur adipiscing elitmod

© 2018 Company. Privacy Policy | Terms & Conditions