MIE350 - Final Report

December 8th 2020

Team 12: Industrial Engineering Online Community, “UofT Indy”

Name	Student Number	Email
1. Doruk Kasimoglu	1004747875	doruk.kasimoglu@mail.utoronto.ca
2. Jina Yazdanpanah	1004204942	jina.yazdanpanah@mail.utoronto.ca
3. Kaan Kanatli	1004813883	kaan.kanatli@mail.utoronto.ca
4. Mohammad Khater	1004971872	m.khater@mail.utoronto.ca
5. Nada Al Aker	1003941185	nada.aker@mail.utoronto.ca
6. Raissa Amuruz	1004779698	raissa.amuruz@mail.utoronto.ca
7. Ramadan Qaoud	1004760760	ramadan.qaoud@mail.utoronto.ca

Table of contents

Executive Summary **3**

1.0 Overview of Web Application **4**
1.1 Purpose 4
1.2 Intended Audience 4
1.3 Content Summary 4

2.0 Discussion of Requirements and Use Cases **5**
2.1 Functional Requirements 5
2.2 Non-Functional Requirements 7
2.3 Use Cases 7
2.3.1 Use Case One: Sign up/log in/ logout/ delete account 7
2.3.2 Use Case Two: Message in a Community Discussion 8
2.3.3 Use Case Three: Provide Course/Instructor Feedback 8
2.3.4 Discussion of Use Cases 8

3.0 Design Principles and Characteristics **12**
3.1 UML Use Case Diagram 12
3.2 UML Class Diagram 13
3.3 UML State Chart Diagram 13
3.4 UML Sequence Diagram 14
3.5 UML Activity Diagram 15

4.0 Web Application Summary and Conclusion **16**

5.0 Difficulties Encountered **16**

6.0 Lessons Learned **17**

7.0 Modifications for Wireless Use **18**

8.0 References **19**

9.0 Appendices **20**

Executive Summary

The report discusses the implementation and process behind creating the website “UofT Indy”, a platform that allows the Industrial Engineering students of the University of Toronto to remain connected through their social and academic needs.

The design and implementation of this website consisted of multiple steps. It started with the creation of the functional and non-functional requirements, necessary for understanding the users needs. Our team has placed emphasis on the necessary requirements, investigating the implementation of LogIn page, Home page, Communities page and a Reviews page. In order to create a well functioning web application, these requirements were then used to develop three use cases. These use cases assisted in the understanding of the data flow involved within our system. The highlighted use cases are the Log-In process, the process to submit a review and the process to reply to a discussion board thread.These use cases were then utilized to create Data Flow Diagrams, graphically representing the exchange of data and serving as a starting point in manufacturing the overall structure of the website. After doing so, a series of UML diagrams composed of the use cases, classes and objects were created that allowed for the identification of the sequences and activities that occurred as well as understanding the system architecture. This process demonstrated the relationships between the objects, which better enforced the implementation of the back-end structure of the web application.

Once these processes were implemented, the website was developed through a combination of front end development and back end development methodologies. This included the use of a server using Apache Tomcat, Microsoft Access Database and Java for Object Oriented Programming on Eclipse Luna. The front-end development is implemented through HTML, CSS, JavaScript, jQuery, and some Bootstrap. Once the website implementation was complete, the difficulties encountered by the team members were discussed. Then the report discusses the methods applied which assisted in overcoming these difficulties. Finally, modifications of the web application are addressed that aid in the development of a better overall experience for the users. The final website consists of multiple web pages. The home page shows our project description and relevant details. The ‘Courses’ page provides an external link to detailed description of the course, as well as it’s reviews that have been submitted by users of the website. Lastly, a communities page that creates a discussion between students on topics they are interested in.

1.0 Overview of Web Application

1.1 Purpose

Our driving motivation over the course of this term project has been to create a web application to support, engage and maintain the industrial engineering community within the University of Toronto. Our team has designed an online community site where the special interest community is the industrial engineering student body. This community shares a range of common interests spanning their academic needs, extra-curricular engagement and general resources. Our web application aims to unite this community through creating dynamic online support in each of these areas. This system is needed more critically now than ever before. Due to the COVID-19 pandemic, students everywhere are isolated from their university community in a novel way. Currently, there are official university resources available and students will have access to their direct social circles. However, this platform will bridge the current gap of community wide peer-based engagement. This is invaluable to the university experience and may be otherwise inaccessible to many students at this time. With the aid of our web application, students will be able to feel supported on a peer-to-peer level, finding a place where their experience is authentically shared.

1.2 Intended Audience

Several aspects of the community can be considered to be stakeholders of this application. The primary stakeholder of this system is the Industrial engineering student body. As the web application will aim to provide a holistic coverage of a students needs, there are several subgroups of the student body which will use the web application. These include students from each year including students on PEY. The industrial engineering student body is also considered the primary user group of this web application. Currently, there is no functionality in place for industrial engineering alumni, however this can be considered a future update to the web applications functionality. The primary assumptions of this web application are that students will be willing to transition their university-related activities to the newly developed platform. Related to this assumption, is the assumption that only individuals who are in the intended user group will create an account through the web application. Currently, there is no design in order to account for student verification, within UofT or within Industrial Engineering, and thus the application operates on a basis of good faith. It is also necessary to assume that this platform will not create any conflicts with the university administration in order for it to be developed successfully.

1.3 Content Summary

In terms of the website structure overview, our platform draws inspiration from a combination of resources such as the University of Toronto website [1], “courses.skule.ca” [2], Rate My Professor[3] and Reddit[4]. To provide a brief overview of the functionality of these platforms, “courses.skule.ca” is a site that provides a detailed description on all the engineering courses at the University of Toronto, Rate My Professor is a web application which provides reviews submitted by students on any professors and Reddit is a social news and discussions website. Further information can be found through the referenced sources.

However, our website provides a unique and novel perspective of the functionality addressed by each reference. We attempt to combine the strengths showcased by each platform. We aim to provide accurate and informative academic information, similar to the official university ressources. We aim to provide real feedback from students, similar to “courses.skule.ca” and Rate My Professor. And lastly, we aim to create a

dynamic and engaging platform for students to communicate comfortably, similar to Reddit. Through this website, the main objectives we aim for the students to be able to achieve are to have access to course information, be able to submit and browse real student reviews and to engage in an interactive conversation platform with other industrial engineering students. These comprise the 4 main pillars of the web page functionality we aim to achieve.

Our team has developed a comprehensive range of requirements through iterative brainstorming. Then, based on these requirements, we have generated 3 key use cases. We initially had designed for 4 use cases but have since narrowed our scope due to a reassessment of human resources and abilities. From our core use cases, we have generated a range of UML diagrams in order to visualize and organize the back-end architecture of the system. We then implemented a server using Apache Tomcat. The back end development of this project is implemented through the use of Microsoft Access Database and Java for Object Oriented Programming on Eclipse Luna. This is then integrated to the front end with the help of an XML file that routed to Java Servlets which then accessed the rest of the Java classes. The front-end development is implemented through HTML, CSS, JavaScript, jQuery, and Bootstrap. However at one point the team had changed the HTML pages into dynamic JSP pages.

2.0 Discussion of Requirements and Use Cases

The discussion of our web application first starts with an introduction to the systems requirements. The systems requirements are considered as the basic functionalities the application should fulfill. With respect to the requirement generation, the team engaged in collaborative and iterative brainstorming sessions in order to address the needs of the potential primary user group. These user needs were then translated into system requirements. As the team is composed of individuals who are within the range of the target user group, brainstorming and idea generation was facilitated through considering the gaps in our personal experiences with the resources we currently interact with. This is all within the scope of the target purpose of maintaining a student based community, accessible to all students. As our personal interests could potentially contaminate our objective requirements generation, we also consulted with other students in the primary target audience in order to address their needs and current gaps.

2.1 Functional Requirements

Requirement Description	Importance
Home page: Includes description of webpage's purpose.	Must Have
Log-in and sign up pages: Allow users to log in by entering username and password and display a link that redirects to sign up page where user inputs the information required to create an account.	Must Have
Informational Page: Resources and How to use them based on student feedback	Could Have

Courses Page: Displays all industrial engineering courses from first year to fourth year. Course tab expands and shows links to the course's description and reviews page. Allows users to submit review scores.	Must Have
Navigation Bar: Displays the summary of the website's functionality through interactive buttons for all pages and a search bar. Redirects users to the corresponding clicked page.	Must Have
Account Button: Expands to show buttons to log out and delete accounts.	Must Have
Professors: feedback, advice, reviews of industrial engineering professors	Should Have
Tutors: postings of available students to tutor for industrial engineering courses	Should Have
Calendar: with events that clubs and indy group can post to	Could Have
Your personal profile: Each user will have a customizable profile, it will show the other groups what you're a member of and your activity on different pages	Should Have
Search bar: can search for anything including another person's profile and communities	Could Have
Miscellaneous advice page: like reddit discussion board	Could Have
Communities page: Users reply to threads on any of the communities that exist in the communities database.	Must Have

Table 1. *Functional Requirements*

Table 1 describes the functional requirements of the web community. Functional requirements are defined as the specific task related objectives of the system. From the comprehensive list above, only the "must have" requirements were implemented on the website. These requirements compose the fundamental framework for the website's functionality. This narrowed scope was implemented due to a limitation in human and logistical resources. With a larger scope and more ressources, the other levels of requirements can be explored and developed further. All of the system requirements are integral to achieve our goal of forming solidarity and cooperation among the students. These specific requirements were selected as the team believes they address each of the main pillars of functionality addressed in the contextual motivation detailed in section 1, directly addressing the gap identified. From the brainstorming session, each team member voted on the 20 most relevant requirements towards achieving these goals. Any requirement with less than 4 votes, (ie. less than 50% of the groups approval) were omitted from the scope of analysis. This resulted in the following 13 requirements.

With respect to the types of users, they are as follows: non account users and account users. This structure was chosen as the website should only be accessible to Industrial Engineering students. Therefore, non

account users will only have access to the log-in and sign up page, while account users will gain the ability to participate on all of the website's functionalities.

2.2 Non-Functional Requirements

The non- functional requirements are made up of categories which are operational, performance, and security requirements. All of the below requirements are required t in order for this website to operate and achieve its goals. The user must find the website easily accessible and convenient as well as ensuring the safety of any of the information they provide. These requirements have been generated through benchmarking to similar web applications and assessing the needs of the predicted user group. They will play an important role in the architectural phase of the software design. The system will cover all of the below properties:

Requirement	Description
Operational	The system should work on any Web Browser
Performance	The system should be available for use 24/7
	The system should support 600 users simultaneously
	Monthly website maintenance should be conducted by developers
Security	No user can access to edit other users information
	Personal information of each user is protected
	Authenticated sessions

Table 2. *Nonfunctional Requirements*

2.3 Use Cases

In order to capture the flow of data required to achieve the necessary system requirements, the team has established 3 primary use cases. These use cases represent the flow of data required in order to achieve 3 aspects of the websites main functionalities. They are described and detailed below.

2.3.1 Use Case One: Sign up/log in/ logout/ delete account

The first use case represents the user's interaction with the website in terms of their profile and entering the website community space. This use case is shown in Figure 1. The users, the industrial engineering students, cannot use the website functionalities without creating a profile with our system. As a result, one of the most pivotal requirements is allowing the user to create a profile. The student profile is the user's key to accessing the website. This use case relates to the requirements, first, by directly addressing the login and sign up functionality as well as the account button functionality. Beyond this, this use case establishes the framework for one of our preferred requirements, the personal profile functionality. This use case is fundamental to the basic operation of the webpage.

2.3.2 Use Case Two: Message in a Community Discussion

The second use case describes the interaction of the user with the community pages. This use case is shown in Figure 2. The most important and unique feature of the website is the use of threads to communicate through the discussion forums. This use case describes the flow of data required in order to post or reply to a thread. The exchange of data is central to the thread categorization and the thread content. This use case directly relates to the webpage requirements through the communities requirement. Through this use case, the interaction between the user and the community's layout is facilitated. This feature has been central to the novelty and structure of the UofTIndy webpage.

2.3.3 Use Case Three: Provide Course/Instructor Feedback

The third use case describes the interaction of the user with the instructor/course feedback pages. This is a fundamental functionality that allows the user to see and provide their insights on courses and instructors. In this use case, the flow of data is central to provide and store the specific content of a students feedback. This use case directly addresses and extends two aspects of the webpage requirements; these are the courses requirement as well as the professor's requirement. The professor's requirement is considered a second tier priority, through the “should have” label, however it is seamlessly integrated in the functionality of this use case and is thus incorporated. This use case addresses the second aspect of the website's major pillars; the functionality of a quantitative and qualitative reviewing system. The final major pillar of the website functionality is informative course content, however this is static information and does not require the flow of data, as a result it is not described in detail through a use case.

2.3.4 Discussion of Use Cases

These 3 use cases relate closely not only to the system requirements, but also to the main pillar objectives of the overall website. Through developing and visualizing the interaction of the system with external entities, such as the user and the internal database, a deeper understanding is achieved with respect to the system architecture. This technique also aids the understanding of the type of information the system must be stored as well as how it must be organized. These use cases have been simplified for clarity since the progress report stage. Additionally in order to narrow the scope, given the time constraints of this project, we have omitted a fourth use case, “Forming Club Pages”. This use case considered the additional functionality of a new user type and a new major website pillar. As this use case required further development in various aspects of the website architecture, after investigative analysis, it was determined it was not within the abilities of the team. From these use cases, there are a few major observations to be made. Firstly, the use cases have confirmed the only major actor in this system is the student account holder; it also confirmed the only sub-category of user type is a student non-account holder. Further the process of developing use cases has established the required data frames required, composing the systems complete database. Though the use cases do not capture the complete set of dataframes required, they were able to serve as a framework to expand from. Lastly, the process of use case generation has revealed several gaps in the system. In the event more time is available to fine-tune the functionality of the website, the areas of improvement identified through the Use cases are; a form of student verification upon account creation and a form of course verification upon submitting a review (ie. to ensure the student has been enrolled in this course).

Use Case Name: Sign up/ log in / log out/ delete account		**ID Number:** 1	
Short Description: This describes how the user creates a profile (signs-up) or logs in to their pre-existing profile			
Trigger: The user (student) wants to participate in the online community, and thus must initiate an account in order to view the web page **Type:** External			
Major Inputs		**Major Outputs**	
Description: - Sign up account information (username, display name, password, year, clubs involved, courses taken, current courses) - Login information (username, password) - Account cancellation request - Student validation	Source: - Student - Student - Student - SR	Description: - New account - User details - Delete account	Destination: - StudentRecords Database (SR) - SR - SR
Major Steps Performed: If the user doesn't have a pre-existing account: 1. User requests to sign-up 1.1. User provides system with username, display name, password, year, clubs involved, courses taken, current course 2. If the user requests to log in: 2.1. User Inputs Username and Password 2.2. System checks existence of user in StudentRecords 2.2.1. If student found, user taken to home page 2.2.2. If student not found, remain in home page 3. If the user wants to log out: 3.1. User requests to log out 3.2. System logs out from user's account and redirects user to login page 4. If the user requests to Delete account (assumes user is already logged in): 4.1. Account Delete		**Information for Steps:** ← Signup account information → New account ← Login information → User details ← student validation ← student validation ← Log out request → Session ends → Delete account	

Figure 1: Use Case 1 describing the Sign up/log-in/ logout/ delete account process

Use Case Name: Message in a community discussion.		**ID Number:** 2	
Short Description: This describes how the student creates a comment in the discussion forum.			
Trigger: The student has a discussion topic or the student has a question/answer for a discussion in mind. **Type:** External			
Major Inputs		**Major Outputs**	
Description: - Community Name - Thread Info (Title, Content) - Reply Info (Content)	Source: - Student - Student - Student	Description: - Community Page - New Thread Entry - New Reply Entry	Destination: - CommunitiesDB - ThreadsDB - ThreadsDB
Major Steps Performed: 1. If user wants to access a community thread 1.1. User accesses the desired community 1.2. System redirects user to the community's page 2. Inside a thread: 2.1. System requests thread's title and content 2.2. System stores thread in threads DB and displays the thread on the page 3. To reply to a thread: 3.1. System request reply content 3.2. User inputs reply 3.3. System stores reply		**Information for Steps:** ← Community Name → Community Page ← Thread Info → New Thread Entry ← Reply Info → New Reply entry	

Figure 2: Use Case 2 describing how the user sends a message in a community discussion

Use Case Name: Provide course/instructor feedback		**ID Number:** 3	
Short Description: This describes how the user will be able to provide feedback for a course or instructor.			
Trigger: The user (student) wants to share his experience or comments about a specific course or instructor for the reference of possible future students of the course/instructor. **Type:** External			
Major Inputs		**Major Outputs**	
Description: - Course code - Instructor rating - Course rating - Difficulty level - Workload level - Comments	Source: - Student - Student - Student - Student - Student - Student - Student	Description: - New instructor rating - New course rating - New difficulty level - New workload level - New comment	Destination: - ReviewsfDB - ReviewsDB - ReviewsDB - ReviewsDB - ReviewsDB
Major Steps Performed:		**Information for Steps:**	
1. On the reviews page, user inputs course code.		← Course code. → Form for the user to fill	
2. System provides a form for the user to input course code, instructor rating, course rating, difficulty, workload, and comments.		← Course code ← Instructor rating ← Course rating ← Difficulty level ← Workload level ← Comments → New instructor rating → New course rating → New difficulty level → New workload level → new comment	

Figure 3: Use Case 3 describing how the user can provide course and instructor feedback

3.0 Design Principles and Characteristics

With the 3 primary use cases of the overall system established, the team utilises Unified Modelling Language (UML) principles in order to develop and represent the relationships within the system, spanning multiple depths. This section intends to demonstrate the tangible methods and logical relationships implemented in order to achieve the contextual objectives of the primary user, as outlined through the requirements. Through these diagrams, we also reveal areas of emphasis required to successfully accomplish all system requirements.

3.1 UML Use Case Diagram

The UML Use Case Diagram in Figure 4 describes the relationship the actors have with the overall system, the use cases described in section 2 are highlighted within the diagram. As represented, the main external actors in the system are the students, differentiated on the basis of being an account holder or a non-account holder. The students which are account holders are able to interact with the systems full functionality through signing in, which grants them access to all the webpages. Conversely, the non-account holders are restricted to only having accessibility to the sign up/log-in page. These user types must convert to a "account holder" user type by creating an account in order to have access to the other aspects of the web application. This firstly addresses the system requirements through providing a difference in access between individuals with accounts versus individuals without accounts. Additionally, this addresses an essential assumption of the system. The system is defined for the use of industrial engineering students, thus access should be limited to this user group. Though there is no formal process to authenticate this aspect of a user's identity, it is assumed that only industrial engineering students will proceed with the account registration process.

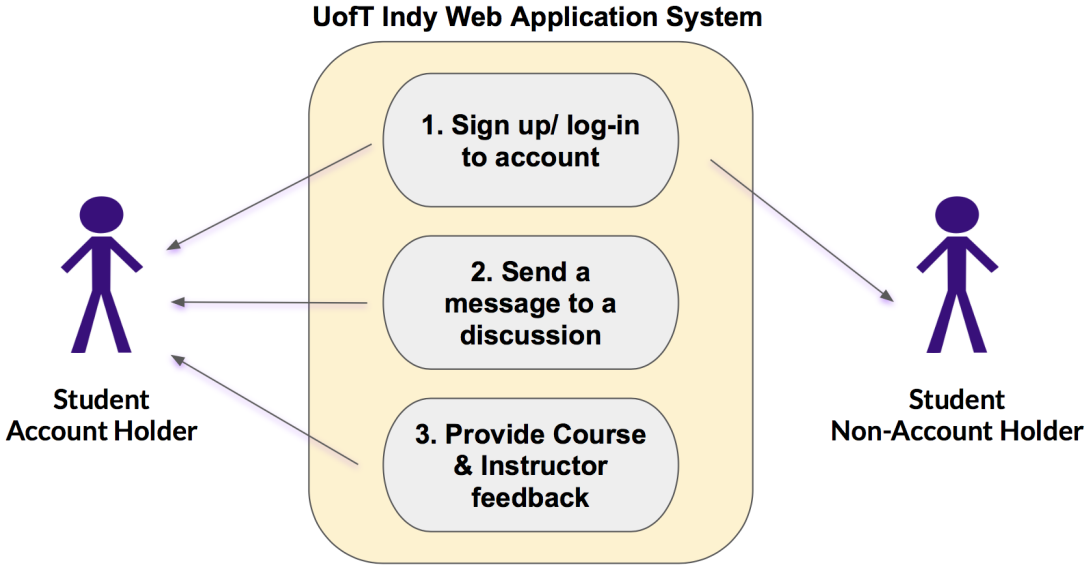


Figure 4: UML Use Case Diagram

3.2 UML Class Diagram

The UML class diagram in Figure 5 represents the back-end structure of the web application as well as the relationships between entities. This diagram primarily showcases the User objects relationship with each object in the system, such as creating threads or providing reviews and replies. Meanwhile, the Course object is stored and referenced based on the needed requests of the Review object. The composite aggregation relationships exist between the Course and Review objects as a review cannot be composed for any course unless that course exists, the same applies for the Thread and Reply objects.

With respect to the back-end structure, we have created static objects following the structure as shown and then created functions which connect the database to the servlets in a separate set of classes. Through these alternative classes we have also included other important methods for internal calculations of things such as ratings and levels of workload or difficulty and their averages.

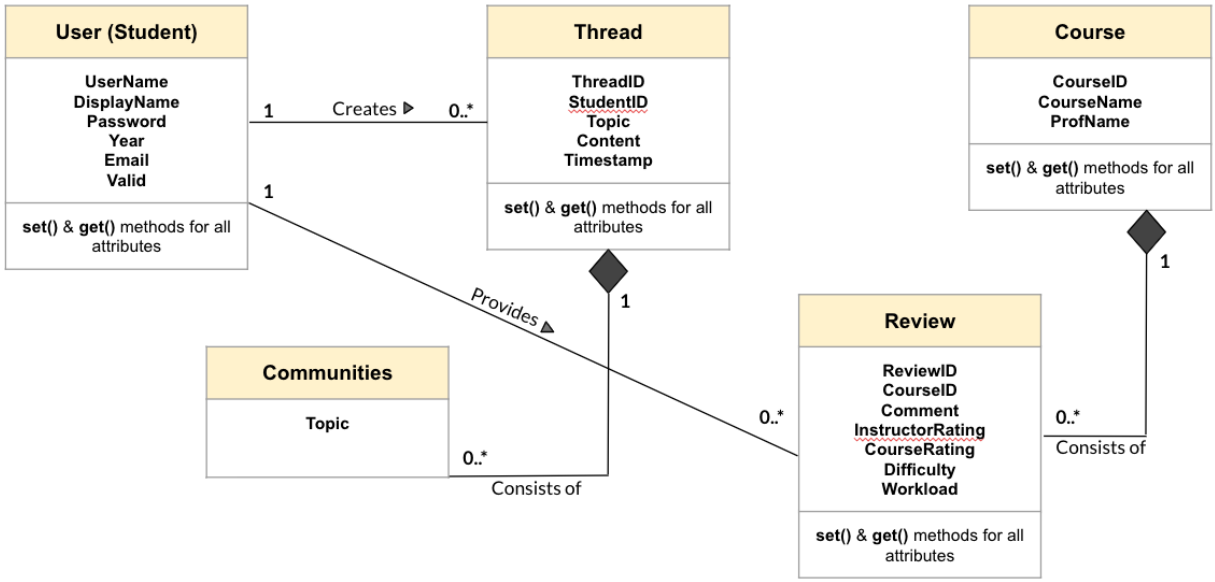


Figure 5: UML Class Diagram

3.3 UML State Chart Diagram

We have highlighted and described the Review object specifically with the use of a state chart diagram due to its importance in relation to the Course object. As described, each course has a set of reviews provided by students, therefore it is important to check the validity of each review. This process is crucial to providing a system which holds integrity and is reliable.

The UML state chart diagram is shown in Figure 6. It is used to analyze the Review object class. The different states experienced by a review are represented through different object states. These object states are shown. Once the review is submitted by a user, the object is sent into the first state, where the values are placed and waiting to be reviewed. The next state is the reviewal process of the object. The processing system requires making the decision on whether the submission is accepted or denied. The events leading to an acceptance or rejection are associated if the data falls within the acceptable data range defined. If the data

values are accepted, they are then used to update the course's averages. In the case that the review values submitted do not correspond to the acceptable range (0-5), the review is rejected and discarded.

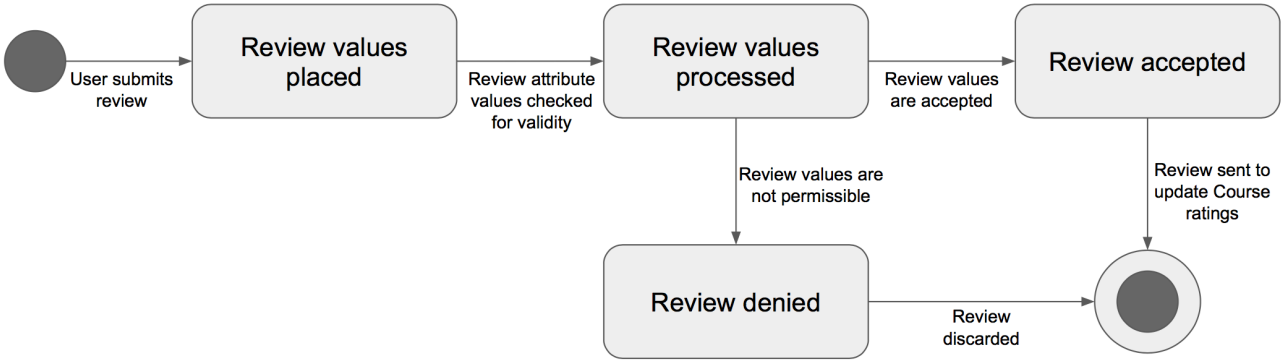


Figure 6: UML State Chart Diagram for Review object

3.4 UML Sequence Diagram

Now we analyze the third use case (Provide course/instructor feedback) through the use of a sequence diagram. This is highlighted because it is important to show the order the objects interact with each other in the scope of a review retrieval. This is in order to further demonstrate and verify the system integrity. It is also important to demonstrate the long term behaviour of our system, highlighted through the lifetimes of this sequence. This specific relationship is shown because it specifically demonstrates the impact that deleting a user account will have; specifically that it will impact other objects in the system, it is not stand alone.

The sequence diagram shown in Figure 7 corresponds to the third use case. The objects taking part in this interaction are the student (user), the result set titled reviews, and the individual instance of a course review object class. The process flow associated with this sequence diagram is as follows: the student first sends a message to the review result set to locate a specific course review. This is done by inputting the CourseID (course code). Then, the review object returns, to the student, the course review for that specific CourseID. Here the user is able to view the average ratings for courses and instructors. Finally, the user submits their own review page to store the user's reviews and comment.

The sequence diagram shows messages between the objects in sequential order. Moreover, the X on the Student and CourseReview objects on the same height level indicate that the lifetime of the object student's course reviews is dependent on the lifetime of the student user. This means that if the user decides to delete his/her account, then his/her course reviews and comments would also be deleted.

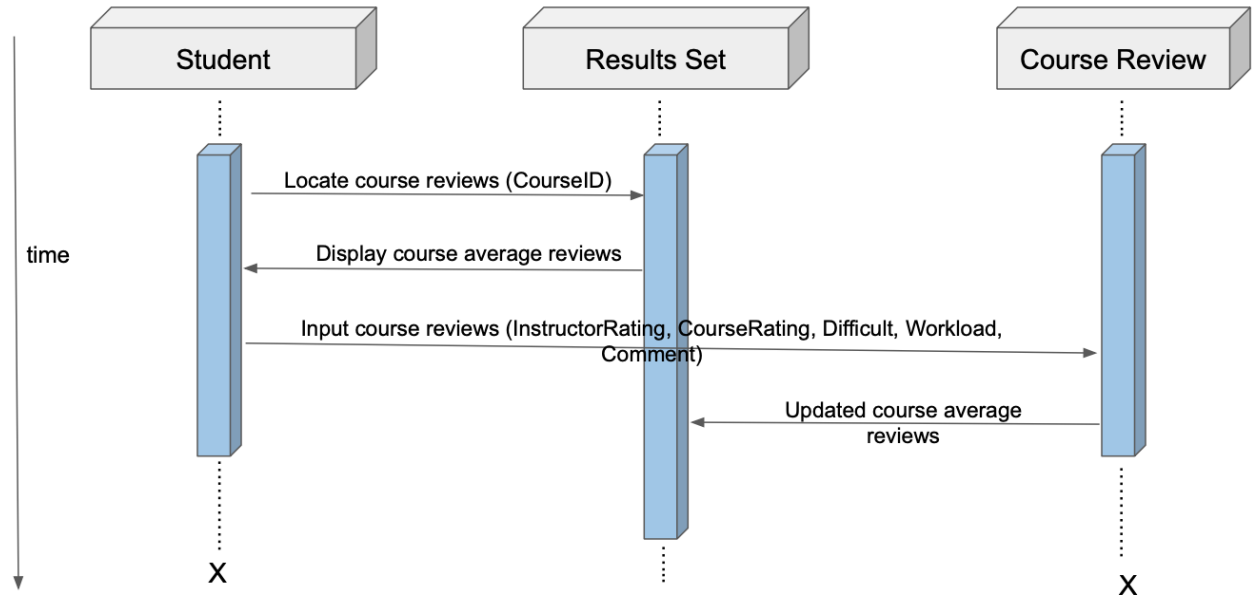


Figure 7: UML Sequence Diagram corresponding to the third use case

3.5 UML Activity Diagram

The login process is now further analyzed and showcased through the use of an activity diagram. This is important to highlight because an individual can access the web page through multiple sources. This activity diagram will showcase these multiple routes.

The activity diagram in Figure 8 corresponds to the first use case presented. This diagram was implemented using the Level 1 Data Flow Diagram shown in Appendix A. As seen by the diagram, the different access sequences are shown. In summary, the user could either create an account, log in or delete an account. In the case the user decides to log in, the next activity that could possibly occur is logging out once the session ends.

This diagram is noteworthy as it directly relates to the development process outlined by the requirements generation and use cases. This represents the sequential process this functionality represents.

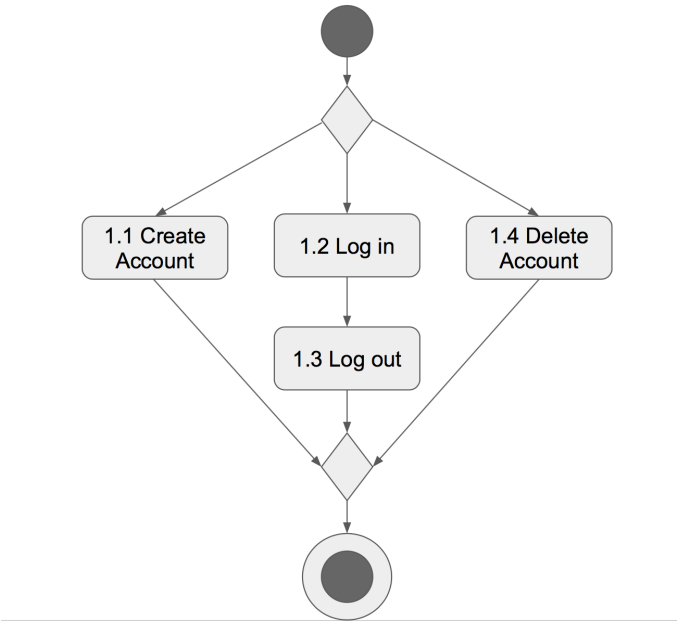


Figure 8: UML Activity Diagram corresponding to the first use case

4.0 Web Application Summary and Conclusion

To summarize, our team has developed an online web community entitled “UofTIndy”. This community aims to support, engage and maintain the industrial engineering community within the University of Toronto. Our web application aims to unite this community through creating dynamic online support in each of these areas. This platform is intended to bridge the current gap of community wide peer-based engagement. With the aid of our web application, students will be able to feel supported on a peer-to-peer level, finding a place where their experience is authentically shared. The main functional objectives we aim for the students to be able to achieve are to have access to course information, be able to submit and browse real student reviews and to engage in an interactive conversation platform with other industrial engineering students. These comprise the 4 main pillars of the web page functionality we aim to achieve. Through this document, we have detailed the core architecture of this system through outlining the requirements, highlighting the use cases, analyzing and visualizing specific functionalities through UML diagrams and discussing the contextual relevance of each aspect. The functional website has been fully implemented and has been submitted.

5.0 Difficulties Encountered

In reflection, in the development of this web application, we were faced with a selection of major challenges. In general, we struggled with task and role division, time-zone differences among the team members and meeting all the team’s initial requirements and expectations.

With respect to task and role division, as our team was composed of beginners with respect to software development methodologies and web development specifically, we had a lack of understanding of how tasks should be properly divided. This resulted in pivoting the task division as the project progressed multiple times. With respect to time-zones, our team was split between two distinct and un-compatible timezones, resulting in a difficulty to align efforts. Lastly, our scope was initially very wide and ambitious, resulting in many iterations to narrow our project scope.

To overcome these challenges,

1. We reorganized the structure of our approach and assigned one to two people to work on a specific tasks
2. We split into two work schedule teams to maximize efficiency by having a night shift for the people in North America and a morning shift for the people in the Middle East.
3. We realized that our initial goal was not realistic and decided to eliminate certain initial functionalities to narrow our scope.

The team also faced some difficulties related to the website functionality, appearance and creativity. In regards to the functionality of our website, and integrating our front-end and back-end functionalities. Moreover, the team was also challenged in the front-end development specifically as we are all beginners in this respect. Another difficulty that we struggled with was making the appearance of the website look professional, consistent and attractive. This was due to the team's lack of experience, as well as the short time constraint for learning.

One specific difficulty we encountered was the implementation of threads and replies in our communities section of the web application as there were certain complications that prevented us from executing it. Therefore, we decided on creating a discussion board for each community that allows users to send messages to each other regarding the topics they are interested in.

6.0 Lessons Learned

Reflecting on the difficulties we encountered, in order to see optimized results, we believe we should have dedicated a period of time to learn all of the front-end development skills, such as HTML, CSS and JavaScript as well as JSP pages and servlets . We would also have recommended using the same educational sources and platforms among all team members to ensure that all team members have the same level of background knowledge. This would have prevented role and task division conflicts and improved the efficiency of our team in the development of the website. It would also have been more effective to learn the web development techniques before the decision of main requirements since the team lacked information about the difficulty level of certain aspects of implementation. Since the Data flow diagrams, Use Case Diagrams and UML were done without experience in web development, a lot of iteration had to be done on the design aspects of the project. If these documents were well prepared, It would let the group be well organized and the project would have been finished in less time. Besides the development of the web application, the project taught all the team members how important it is to be scheduled and well prepared for the deadlines and important dates. This was the first time the members of the team worked in a team of 7

during university. It gave the group exposure to a professional work environment, since each member had to manage their time accordingly with less communication. Due to the Covid-19 pandemic, the team had to work in different hours and only in a virtual environment. This was a great experience since the members learned how to run a project without any human interaction. Since the world has seen that it is possible to work virtually, the professional environments may begin shifting in this direction. The team is well prepared for any upcoming project that will be done fully virtually.

7.0 Modifications for Wireless Use

With today's growing development of mobile applications and people using their phones to browse on the Internet, it is important for the team to consider how to modify our web application to work on any mobile device. The prerequisites to make a mobile-friendly website are a working knowledge of HTML and CSS. First, the team will need to override the default of the website's fixed width to adapt to the actual device's width with a scaling factor of 1. This will need to be modified in both the HTML and CSS Stylesheet. The next step is to manually resize elements that need resizing (e.g. images) to adapt to the phone's screen and leave others the way they are (e.g. words). To accomplish this, media queries (CSS feature) need to be created in order to enable web page content to adapt to different screen sizes and resolutions. The design should accommodate all wireless devices from an old Android (240 px) to an iPad landscape (1024 px) [5]. The following functionalities in the website would be limited in the mobile version if not correctly modified for responsiveness, such as the navigation bar, the thread communities, the course pages etc.

In order to assess the suitability of the mobile app and accomplish a successful design, certain usability heuristics and human factors need to be considered:

- **Aesthetic and minimalist design:** less important information should be pushed to the bottom of the page. The navigation bar should be modified to a single button that expands to show all the web's tabs.
- **Proximity compatibility (attention principles):** clickable items shouldn't be places too close together.
- **Perception principles:** resize elements to make displays legible. Make elements accessible and discriminable.
- **Interaction principles:** support responsive feedback on a touch screen.

In regards to security in our web application, the privacy of the account users is protected from no account users. Our web application checks in the StudentRecords database if the username and password input exists. If the user exists, it allows the user to access all web functionalities, otherwise the user can only access the signup and login page. The application currently only has a very simple password authentication security measure. However, in the future we would like to implement a double authentication system. This system would check if the UTorID and password provided exist in the official University of Toronto student records. Only when the student is verified to exist, the user would be redirected to the sign up page.

8.0 References

[1] *University of Toronto*. [Online]. Available: https://www.utoronto.ca/. [Accessed: 9-Dec-2020].

[2] *Skule Courses*. [Online]. Available: http://courses.skule.ca/. [Accessed: 9-Dec-2020].

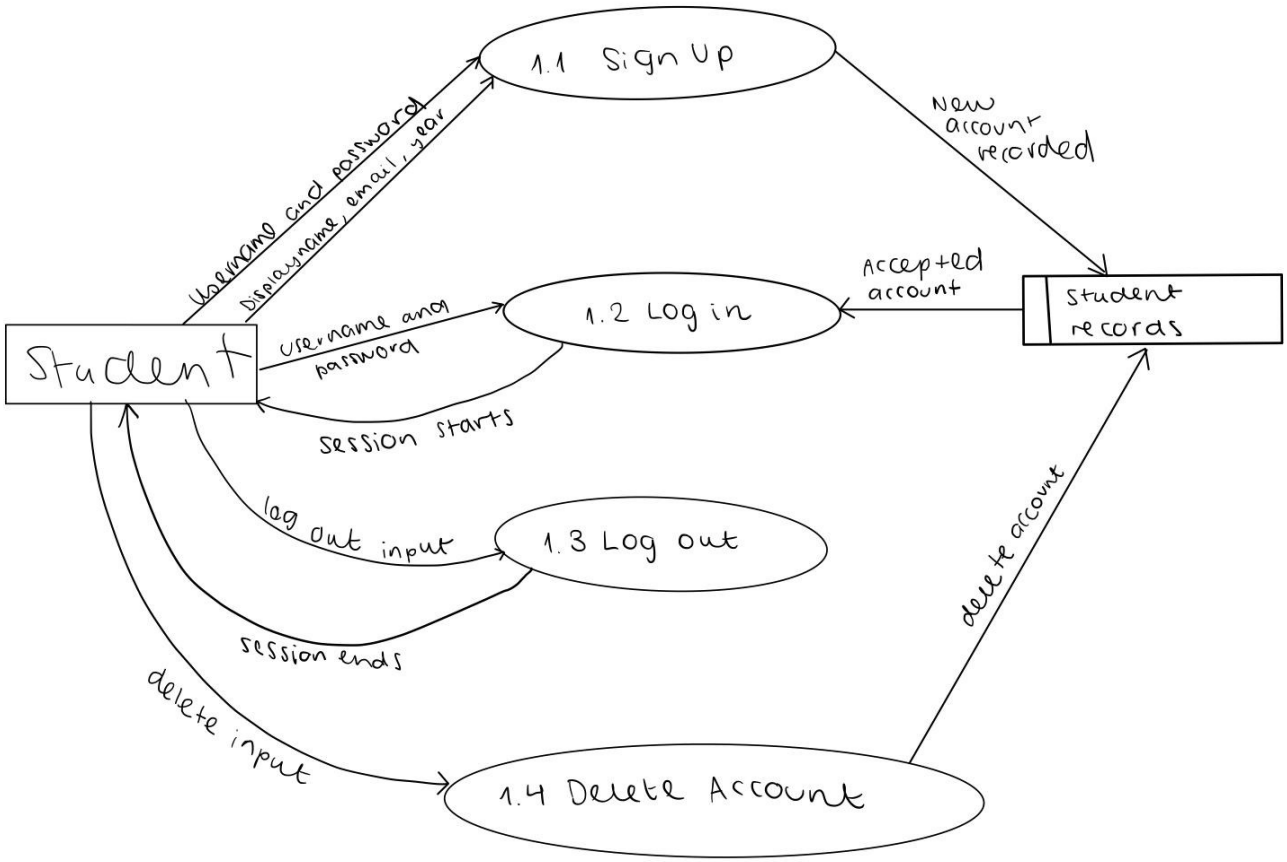
[3] “Find and rate your professor or campus.,” *Rate My Professors - Review Teachers and Professors, School Reviews, College Campus Ratings*. [Online]. Available: https://www.ratemyprofessors.com/. [Accessed: 9-Dec-2020].

[4] *reddit*. [Online]. Available: https://www.reddit.com/. [Accessed: 9-Dec-2020].

[5] C. Heng, “How to Make a Mobile-Friendly Website: Responsive Design in CSS,” *thesitewizardcom RSS*. [Online]. Available: https://www.thesitewizard.com/css/mobile-friendly-responsive-design.shtml. [Accessed: 9-Dec-2020].

9.0 Appendices

Appendix A: Level 1 Data Flow Diagram corresponding to the first use case used to create the UML Activity Diagram.



Appendix B: Project Log sheets with completion dates for each milestone

Front-End				
Task	**Assigned To**	**Milestone**	**Due Date**	**Date Achieved**
Create HTML pages	Doruk & Nada	WebApp	Nov 1st	Nov 1st
Create Nav Bar	Nada	WebApp	Nov 5th	Nov 6th
Create CSS styling	Nada & Doruk	WebApp	Nov 3rd	Nov 10th

Coordinate with BackEnd	Nada	WebApp + Server	Nov 20th	Nov 20th
Troubleshoot Back End & Front End	Nada, Mohammad, Kaan	WebApp + Server	Dec 2nd	Dec 1
Back-End				
Task	**Assigned To**	**Milestone**	**Due Date**	**Date Achieved**
Database designed	Kaan	Progress Rep	Oct 26th	Oct 15th
Server prototype	Mohammad	Server Prot.	Nov 2	Oct 30
Accounts Functionalities	Mohammad and Kaan	WebApp	Nov 18	Nov 17
Reviews Functionalities	Mohammad and Kaan	WebApp	Nov 25	Nov 23
Threads Functionalities	Mohammad and Kaan	WebApp	Nov 29	Nov 28
Connect with Front End	Mohammad and Kaan	WebApp	Dec 2	Dec 1
Design, Writing and Planning				
Task	**Assigned To**	**Milestone**	**Due Date**	**Date Achieved**
Presentation Design	Jina, Raissa, & Nada	Presentation	Nov 28th	Nov 30th
Presentation Content	Jina, Raissa, Ramadan & Nada	Presentation	Nov 28th	Nov 30th
Design WebApp Structure	Nada Raissa & Jina	WebApp	Nov 2nd	Nov 1st
Reports Skeleton	Ramadan	Reports	-	-
Progress Report	All			

Appendix C: Electronic copy of the website (emailed to the instructor)

Appendix D: Electronic copy of the final class presentation material (emailed to the instructor)

Presentation link:
https://docs.google.com/presentation/d/1yO6Cz_orrS3zeemOJtCshny9pCZgX3Bnn9qsjKG5FnQ/edit?usp=sharing

Appendix E: Detailed description of the roles and responsibilities of each team member

The team has split the group to 3 sub groups at the beginning of the project. These were the Front-End team, the Back-End team and Design & Documentation team. However as we proceeded the members have worked in every part of the project. The roles for every member can be found below.

Doruk Kasimoglu: Doruk was mostly involved in the front end team as he had previous experience. He assisted in the creation of some HTML pages and contributed to the CSS design. After the transfer of HTML pages to JSP, he was involved in implementing JSP code into the front end page as well as helping out the backend team.

Raissa Amuruz: Raissa was one of the leaders in the group in the case of project management. She was responsible with the progress of the project and she was keeping up with the team members and made sure everything was on track. She has helped the front end and back end team and made sure every product was inline with the initial project ideas.

Nada Aker: Nada had the most experience in web development so she was assigned a full-stack development role, as well as designing the project structure. She created the HTML and CSS files with the help of Doruk, and adjusted them to integrate with the back end team. After the transfer of the front end to the Eclipse IDE she helped the backend team in debugging and modifications.

Mohammad Khater: Mohammad was the lead on the back end of the project. He was the lead engineer for the servers. The implementation of the server was done by him and the management of the servlets was arranged. Beside the servers, he had an impact on the creation of the object oriented methods and the debugging process of the issues that were faced in the back-end.

Jina Yazdanpanah: Jina held the role of project manager, responsible for the general progress and organization of the group. She contributed to the overall structure of the implementation strategy and task allocations. She mostly worked in the design team and liaised to make sure everything was applied properly on the front end and back end. As well as design, she played a huge role in the documentation and the presentation of the project.

Ramadan Qaoud: Ramadan was mostly involved in the design and documentation. He was always in touch with the members in the web development part and made sure everything was documented properly and everything was inline with the initial designs. He had a huge role on the decision of the requirements and the use cases. He helped the back end team when help was needed.

Kaan Kanatli: Along with Mohammad, Kaan was responsible for the back-end as well as the database. The instructions for the database architecture were implemented by him. Additionally, with Mohammad, object

oriented methods were designed. He implemented his knowledge to make sure the system was correctly communicating with the back end.