Intelligent Academic Planner

Daria Cook: Major in SE
Krystal Elliott: Major in SE
Xiaoyu Zhou: Major in CS

Course Instructor: Xiaocong Fan
Faculty Advisor: Meng Su

Industry Sponsor: PSU Behrend CSSE Faculty
Project Mentor: Adriano Cavalcanti
Assistant Professor of CSSE

A capstone project report submitted to the faculty of
The Computer Science and Software Engineering Department
Penn State Erie, The Behrend College

February 2017
(Version 3.5)

CSSE Technical Report Series: CSSE-BD-Class2017-003

Table of Contents

1. Abstract

The IBM Watson uses “natural language processing and machine learning” to reveal insights from large amounts of unstructured data [1]. We will use Watson to create a structured analysis of possible academic careers related to the CSSE majors at Penn State Behrend.

We seek to enable students to ask questions relating to these majors and to provide valuable feedback, promoting better decision making about academic and professional careers. Our tools will also assist advisors in preparing relevant and unique advice to each student seeking their guidance.

In this report, we arrange our goals as user and system requirements, showing the engineering process of this project. UML Diagrams are also provided for further detail and explanation of this process.

2. Report Revision History

2.1. Changes in Version 1.5

- ➔ Reformatted the document to look nicer
- ➔ Removed voice-to-text use cases, requirements, and sequence diagrams
- ➔ Modified Ask Question sequence diagram to not include voice-to-text reference
- ➔ Modified descriptions of all images
- ➔ Added more definitions to glossary
- ➔ Reworded the Abstract and Problem Statements
- ➔ Added more examples to the Exploratory Studies Techniques
- ➔ Added in more references
- ➔ Created in-text citations
- ➔ Added in picture for architectural design

2.2. Changes in Version 2.0

- ➔ Updated progress in Exploratory Studies
- ➔ Added in References
- ➔ Added in Unit Test Cases
- ➔ Added in Unit Test Execution Reports
- ➔ Added in UML Class Diagram in Structural Design
- ➔ Added in UML State Diagram in Behavioral Design
- ➔ Added steps for installing test framework
- ➔ Added steps for running test cases
- ➔ Added in challenges for system development
- ➔ Added alternative UI designs

2.3. Changes in Version 2.5

- ➔ Modified format of references
- ➔ Updated description of UML Diagram
- ➔ Added description of requirements trace table
- ➔ Modified abstract slightly
- ➔ Exploratory Studies expanded to explain purpose
- ➔ References mentioned in Exploratory Studies
- ➔ Implementation Alternatives & Decision Rationale expanded
- ➔ Architectural Design image discussed
- ➔ Relevant Packages/Products expanded

2.4. Changes in Version 3.0

- ➔ Updated Use Cases and Requirements to use appendices
- ➔ Updated Test Cases and Test Execution Reports to use appendices

- ➔ Added in Test Suite section under System Testing
- ➔ Changed order of subsections in System Testing
- ➔ Fixed grammar in Problem Statement
- ➔ Made a correction in Business Background
- ➔ Exploratory studies updated to reflect new strategies in question creation
- ➔ Broader Impacts updated to reflect target audience
- ➔ Relevant Packages/Products corrected for greater clarity
- ➔ State Diagram in Section 6 updated to reflect new web flow
- ➔ User Interface pictures updated to reflect current system
- ➔ Old user interface images and captions moved to alternative designs
- ➔ Update preprocessing algorithm
- ➔ Added in a Coding Convention
- ➔ Modified System Development Instructions
- ➔ Added in further system extension options
- ➔ Added in an additional challenge in system development
- ➔ Added in an open issue and solution

2.5. Changes in Version 3.5

- ➔ Modified table descriptions for use cases, requirements, test cases, and test execution reports to be more user friendly and less boring.
- ➔ Modified Objectives to include local, institutional, and global impacts
- ➔ Added in End User Manual section
- ➔ Modified design patterns

3. Problem Statement

3.1. Business Background

IBM Watson's services, provided by BlueMix's APIs, and other services from 3rd parties or developers can be utilized to conduct textual analysis and output a numerical scale of performance factor. Web Experience Management (WEM) can be trained to answer many open-ended questions. The question we are trying to answer is whether we can create a system to assist with the advisement process, through use of BlueMix and WEM.

3.2. Needs

To increase the effectiveness of the IBM Watson services, a larger domain is needed. Additionally, students are often unsure of which fields would coincide with their interests and talents.

3.3. Objectives

The objective of this project is to use the IBM Watson cognitive services to create a tool that can provide consultation to students, who are interested in computing majors for their academic queries, while factoring this in assessment of their personality. The tool should be able to create an assessment based upon the student's self-description and other documents they submit to the system (for example, a transcript), and the types of questions the user asks.

Locally, we hope to impact those interested in Penn State Behrend's CSSE programs, by offering guidance concerning the decisions pertaining to their future career. On an organizational level, we believe that our tool could be scaled to assist other programs in universities or institutes providing counsel, such as a juvenile detention center, where perhaps a user is not yet aware of the opportunities that exist for them. Globally, we hope to further the field of knowledge that combines inquiries with a user's personal information for a custom tailored answer retrieval system.

4. Requirements

4.1. User Requirements

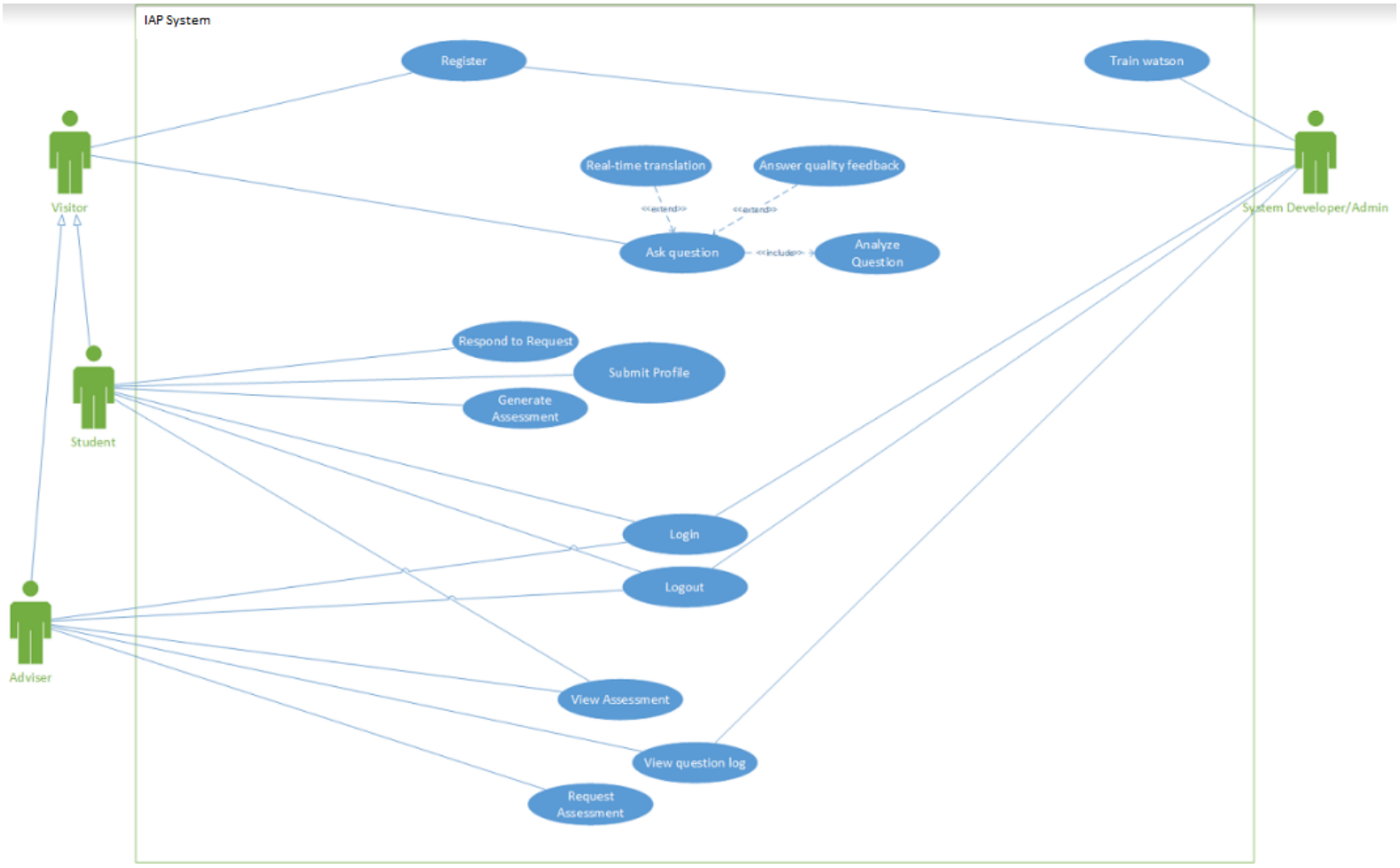
Glossary of Relevant Domain Technology

- **Watson** - An IBM supercomputer that combines artificial intelligence (AI) and sophisticated analytical software for optimal performance as a “question answering” machine.
- **Big Data Analysis** - The process of examining large datasets to uncover hidden patterns, unknown correlations, customer preferences
- **Textual Analysis** - A research method that requires the researcher to closely analyze the content of communication rather than the structure of the content.
- **Web Experience Management** - A process of managing the all-round experience of the web user across various touch points in the journey through an organization's web presence.
- **Use Case Diagram** - A representation of all of the functionalities the system is expected to have and what functionalities a specific user has access to.
- **Use Case** - Communication between a user and the system to perform a specific functionality that is represented in the Use Case Diagram.
- **Sequence Diagram** - A diagram that explains the expected flow of the system once the functionality has been implemented.
- **Data Crawling** - In this context, data crawling refers to the collection of specific data from our own resources, such as our database.
- **Natural Language Processing** - The field of study concerned with the interactions between computers and natural human languages.
- **Machine Learning** - A branch of artificial intelligence in which a computer generates rules underlying or based on raw data that has been fed into it.
- **Supervised Learning** - The machine learning task of inferring a function from labeled training data.
- **Unsupervised Learning** - A type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.

User Groups

- Visitors
- Students
- Advisors
- System Developers

Functional Requirements



Project Scope

This is the Use Case Diagram of our system (Intelligent Academic Planner (IAP) System). Student and Advisor share the options that Visitor has (represented by the inheritance arrows). Each blue bubble represents a functionality that is present in the system, and will be gone into detail in each use case. The functionality of Real-time translation and Answer quality feedback are sometimes used in the Ask question functionality, and Analyze question is always used. This is represented by the “extend” arrows for situational functionality, and “include” arrows for full-time functionality done by the system.

User Scenarios

For a detailed description of how each Use Case performs its tasks, see Appendix U Tables 4.1 - 4.15

A **visitor** has two major use cases: Register and Ask Question. As we aim to make this system usable by anyone, users are not required to log in to ask a question. Thus, we require that the ask question functionality be available to Visitors, who are users that have yet to register.

A **student** has the ability to login, logout, respond to a request, submit a profile, and generate an assessment, as well as the same use cases a visitor has access to (ask question and register). This enables a registered student to access personality assessments and have questions catered to their personality. In addition, they can create their own personal profile.

An **advisor** has the ability to login, logout, view an assessment, view question log, and request an assessment. In addition, they have access to the same use cases as visitors (ask question and register). This allows an advisor to advise students based on questions they've asked and assessments they've received.

A **developer** has special access to tools for training Watson, displayed as the "Train Watson" use case. In addition, they have the ability to view a log of asked questions just like the adviser.

Note: If you are not interested in details about the individual use cases, skip to the next section. The following contains in-depth information regarding each use case for clarity.

Table 4.2: Respond to Request: This use case explains the communication between the user and system to perform the functionality of responding to a request. Only students have access to this functionality, and it is performed when a student receives a request from an Advisor. This allows the user to either accept or deny an advisor's request for their assessment.

Table 4.3: Register: This use case explains the communication between the user and system to perform the functionality of registering. Anyone that accesses the program has access to this functionality. It is triggered when the user requests to register. This allows the user to login.

Table 4.4: View Assessment: This use case explains the communication between the user and the system to perform the functionality of viewing an assessment. This function is only available to logged in users, specifically students and advisors, and is performed when the user requests to view an assessment on their profile. This allows the user to view an assessment that they have generated.

Table 4.5: View Question Log: This use case explains the communication between user and system to perform the functionality of viewing the question log. This functionality is only available to advisors and system developers. It is triggered when the user requests to view the question log. This allows the user to view a list of questions that have been asked.

Table 4.6: Train Watson: This use case explains the communication between user and system to perform the functionality of training Watson. This functionality is only available to the system developer and is performed when the user requests to train Watson. This allows the user to view data to assist with the training of Watson in the future.

Table 4.7: Real-time Translation: This use case explains the communication between user and system to perform the functionality of real-time translation. This functionality is available to all users and is performed when the user requests a translation of their question. This allows the user to translate text from well-known languages to English. This is called occasionally when the user asks a question, and is present in the Ask Question use case.

Table 4.8: Answer Quality Feedback: This use case explains the communication between user and system to perform the functionality of sending answer quality feedback. This functionality is available to all users and is performed when the user requests to submit feedback on their response. This allows the user to submit feedback on the response to their question. This is called occasionally when the user asks a question, and is present in the Ask Question use case.

Table 4.9: Ask Question: This use case explains the communication between user and system to perform the functionality of asking a question. This functionality is available to all users and is performed when the user requests to ask a question. This allows the user to submit a question and receive a response. This use case occasionally uses the functionality of Real-time Translation and Answer Quality Feedback use cases, and always uses the functionality of the Analyze Question use case.

Table 4.10: Analyze Question: This use case explains the communication between user and system to perform the functionality of analyzing a question. This functionality is performed solely by the system. This allows the system to perform a textual analysis on the question that has been asked. This use case is always used in the Ask Question use case.

Table 4.11: Submit Profile: This use case explains the communication between user and system to perform the functionality of submitting a profile. This functionality is available only to students and is performed when the user requests to update their profile. This allows the user to update their profile information.

Table 4.12: Generate Assessment: This use case explains the communication between user and system to perform the functionality of generating an assessment. This functionality is available only to students and is performed when the user requests to generate an assessment. This allows the user to generate an assessment.

Table 4.13: Log In: This use case explains the communication between user and system to perform the functionality of logging in. This functionality is available only to registered users and is performed when the user requests to login. This allows the user to log in to their own session and view their personal information.

Table 4.14: Log Out: This use case explains the communication between user and system to perform the functionality of logging out. This functionality is available only to logged in users and is performed when the user requests to log out. This logs the user out of their personal session.

Table 4.15: Request Assessment: This use case explains the communication between user and system to perform the functionality of requesting an assessment. This functionality is available only to advisors and is performed when the user requests to ask a student for their assessment. This allows the user to send a request to a student for their assessment.

List of User Functional Requirements

For a detailed description of each requirement, see Appendix R Tables 4.16 - 4.42

- A user can log in
  - log-in should occur within 5 seconds
- A user can logout
  - log out should occur within 5 seconds
- A user can ask the system questions
  - System should conduct a textual analysis
  - System should be able to handle input from multiple languages
  - System should recommend majors suitable for the user based on the personality assessment
  - System should gather data unique to the user
  - System should recommend courses based on recommend major
- The system should respond with multiple answers to a question
  - System should show a minimum of 1 related answer
- A user can create a profile page
  - System should allow 100-600 words of academic and professional interests
  - System should allow 100 words of self-description
  - System should create a personality assessment based on profile information
  - System should summarize profile and personality data to be used by advisors on request
- A user can register
  - Register should occur within 5 second
- A user can view a log of asked questions
  - Only developers and advisors can see this log
- A user can provide information to improve the accuracy of the system
  - A user can provide answer quality feedback
- A user's session should be managed
  - A user should log out after 1 hour of inactivity
- A user's profile should be secure

    - The system should only display information the user knows is being displayed
- A user should receive a quick response to a question
    - System should provide an answer within 5 seconds of asking the question.

Note: If you are not interested in details about the individual requirements, skip to the next section. The following contains in-depth information regarding each requirement for clarity.

Table 4.16: Log In Requirement: This user requirement requests that we include functionality in the system for users to log in. High priority was given to this requirement since making individual sessions is required to begin work on the profile.

Table 4.17: Log Out Requirement: This user requirement requests that we include functionality in the system for users to log out. High priority was given to this requirement since a user should be able to log out if they can login, and log in has High priority.

Table 4.18: Ask Question Requirement: This user requirement requests that we include functionality in the system for users to ask the system questions. Highest priority was given to this requirement since our first priority is to allow Watson to answer questions both accurately and uniquely. In addition, since any user can ask a question, logging in and registering is not a requirement to begin work on this.

Table 4.19: Multiple Answer Requirement: This user requirement requests that we include functionality in the system for users to receive multiple responses to a question. This means that if the user asks a question that does not have a clear answer, Watson should ask a question to clarify what is being asked to make the answer clearer. Low priority was given to this requirement since it requires we first implement asking a question, which was given highest priority, and in some cases the second question may require personal information to ask, requiring the profile to be complete which is medium priority.

Table 4.20: Create Profile Requirement: This user requirement requests that we include functionality in the system for users to create a profile. Medium priority was given to this requirement since it requires we first implement registering, logging in, and logging out (all High priority) before we can set up user-specific profiles.

Table 4.21: Register Requirement: This user requirement requests that we include functionality in the system for users to register. High priority was given to this requirement since making individual sessions is required to begin work on the profile.

Table 4.22: View Question Log Requirement: This user requirement requests that we include functionality in the system for users to view a log of asked questions. Medium priority was given to this requirement because it can be easily implemented after completing the ask a question requirement, and because it will assist with increasing the accuracy of Watson.

Table 4.23: System Feedback Requirement: This user requirement requests that we include functionality in the system for users to improve the accuracy of the system by providing feedback. Medium priority was given to this requirement because it can be easily implemented after completing the ask a question requirement, and because it will assist with increasing the accuracy of Watson.

Non-Functional Requirements

Product: Performance Requirements

Table 4.24: Quick Answer Requirement: This user requirement requests that when we create the functionality for asking a question, the system should respond quickly. Low priority was given to this requirement because we are focusing first on implementing features and then focusing on quality and performance of the features.

Product: Dependability/Reliability/Security

Table 4.25: Secure Profile Requirement: This user requirement requests that when we create the functionality for creating a profile, the system should ensure that the profile is secure. High priority was given to this requirement because it should be done while creating the functionality of the profile and we were encouraged to keep security in mind.

Organizational: Development Requirements

Table 4.26: Manage Session Requirement: This user requirement requests that when we create the functionality for registering, logging in, and logging out, the system should ensure that the session is managed. High priority was given to this requirement because it should be done while creating the functionality of logging in, logging out, and registering.

4.2. System Requirements

Functional Requirements

List of System Functional Requirements

Table 4.27: Log In within 5 Seconds Requirement: This system requirement requests that when we create the functionality for logging in, the system should log the user in within 5 seconds. Low priority was given to this requirement because we are focusing first on implementing features and then focusing on quality and performance of the features.

Table 4.28: Log Out within 5 Seconds Requirement: This system requirement requests that when we create the functionality for logging out, the system should log the user out within 5 seconds. Low priority was given to this requirement because we are focusing first on implementing features and then focusing on quality and performance of the features.

Table 4.29: Textual Analysis Requirement: This system requirement requests that when we create the functionality for asking a question, the system should conduct a textual analysis. High priority was given to this requirement because this is one of the key requirements we are focusing on.

Table 4.30: Real-Time Translation Requirement: This system requirement requests that when we create the functionality for asking a question, the system should be able to handle input from multiple well known languages. Lowest priority was given to this requirement because it is a bonus feature that we only intend to implement if time provides.

Table 4.31: Recommend Majors Requirement: This system requirement requests that when we create the functionality for asking a question, the system should be able to recommend majors suitable for the user based on their profile information. Medium priority was given to this requirement because it requires the profile to be complete.

Table 4.32: Unique Data Requirement: This system requirement requests that when we create the functionality for asking a question, the system should be able to gather data unique to each user. Medium priority was given to this requirement because it requires register, logging in, and logging out to be complete.

Table 4.33: Recommend Courses Requirement: This system requirement requests that when we create the functionality for asking a question, the system should be able to recommend courses based on their recommend major. Medium priority was given to this requirement because it requires the profile to be complete and the system should already be able to recommend majors.

Table 4.34: At Least One Answer Requirement: This system requirement requests that when we create the functionality for responding with a question, the system should ask at least one question in response. Low priority was given to this requirement because creating the functionality for responding with a question is also Low priority.

Table 4.35: Academic Profile Requirement: This system requirement requests that when we create the functionality for creating a profile, the system should allow the user to enter between 100 and 600 words of academic and professional interests. Highest priority was given to this requirement because this is what will be to determine questions and answers related to the user.

Table 4.36: Self-Description Requirement: This system requirement requests that when we create the functionality for creating a profile, the system should allow the user to enter 100 words of self-description. Medium priority was given to this requirement because this will be used to determine question and answers, but will be taken into account after professional and academic interests.

Table 4.37: View Personality Assessment Requirement: This system requirement requests that when we create the functionality for creating a profile, the system should allow the user to view personality assessments. High priority was given to this requirement because this is one of the main resources advisors will use.

Table 4.38: Generate Personality Assessment Requirement: This system requirement requests that when we create the functionality for creating a profile, the system should allow the user to generate personality assessments. Medium priority was given to this requirement because it requires the 100 words of self-description to be complete.

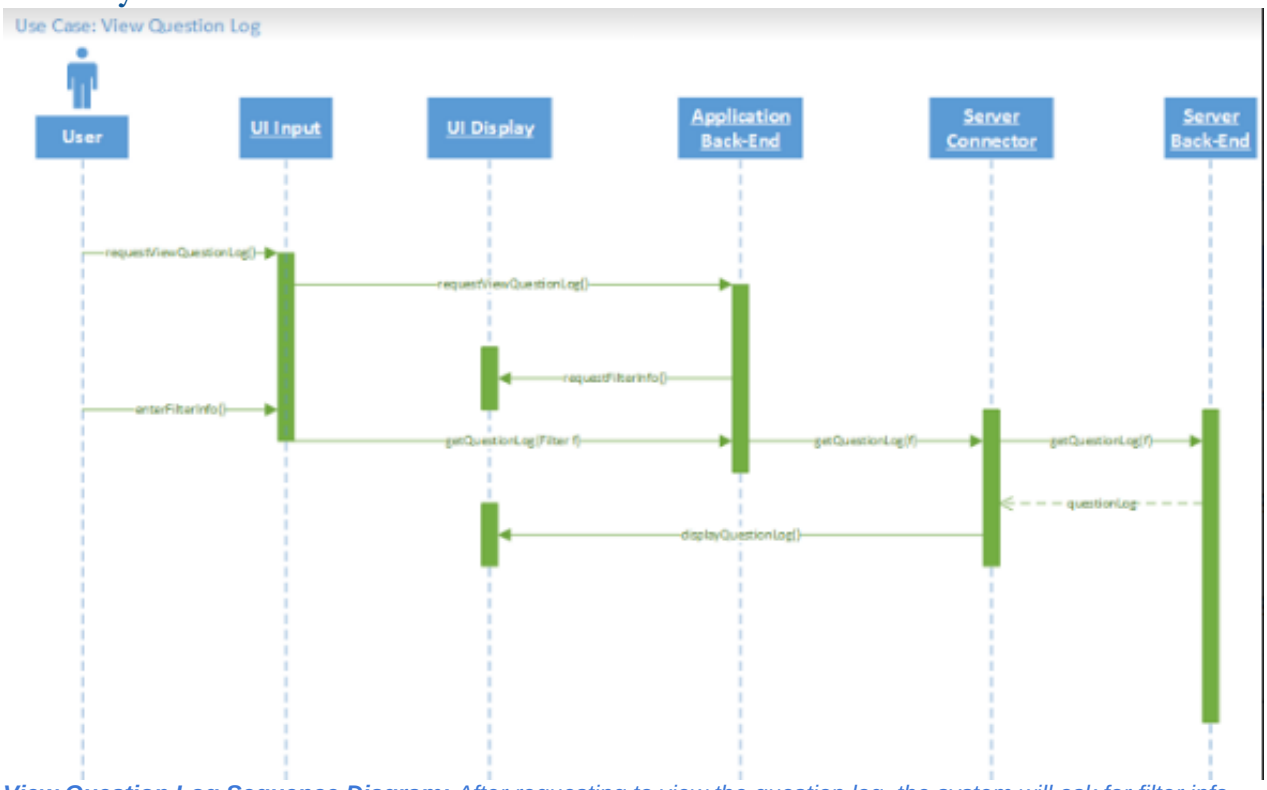
Table 4.39: Summarize Data Requirement: This system requirement requests that when we create the functionality for creating a profile, the system should summarize data for advisors. Medium priority was given to this requirement because it requires the profile to be fully complete before a summarization can be created.

Table 4.40: Register Within 5 Seconds Requirement: This system requirement requests that when we create the functionality for registering, the system should register the user within 5 seconds. Medium priority was given to this requirement because we are focusing first on implementing features and then focusing on quality and performance of the features but this will be one of the first features users' encounter, so it has a slightly higher priority than login and logout performance requirements.

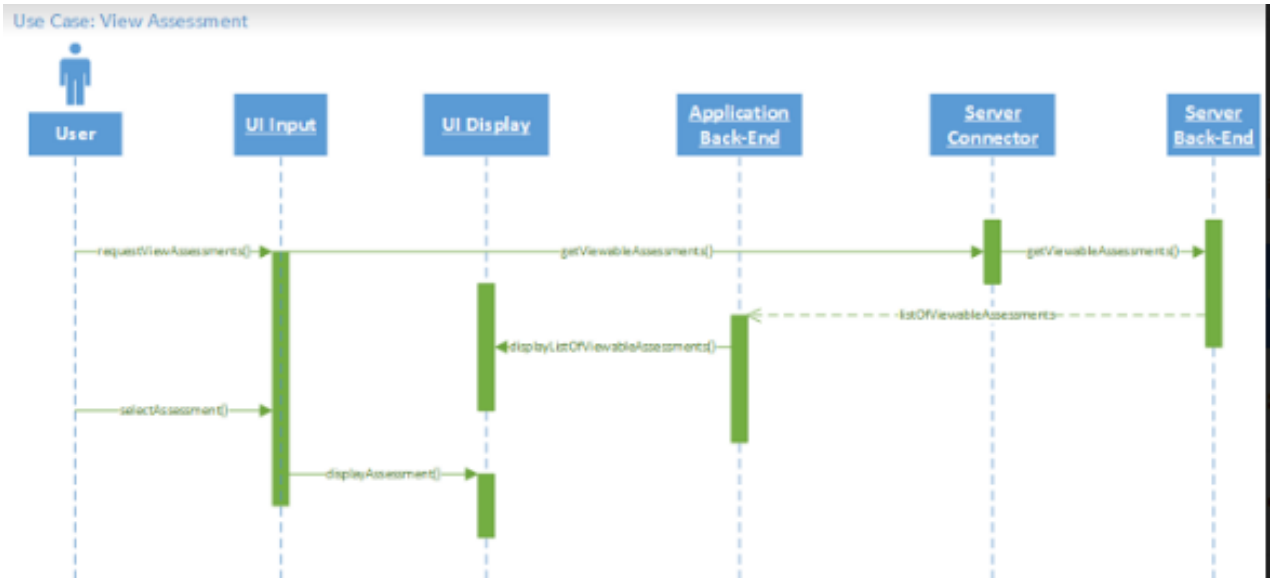
Table 4.41: Question Log View Limitation Requirement: This system requirement requests that when we create the functionality for viewing the question log, the system should only allow advisors and system developers to view it. Medium priority was given to this requirement because the functionality of the question log is also medium priority.

Table 4.42: Ask For Feedback Requirement: This system requirement requests that when we create the functionality for providing feedback, the system should ask for the feedback after asking a question. Medium priority was given to this requirement because the functionality of providing feedback is also medium priority.

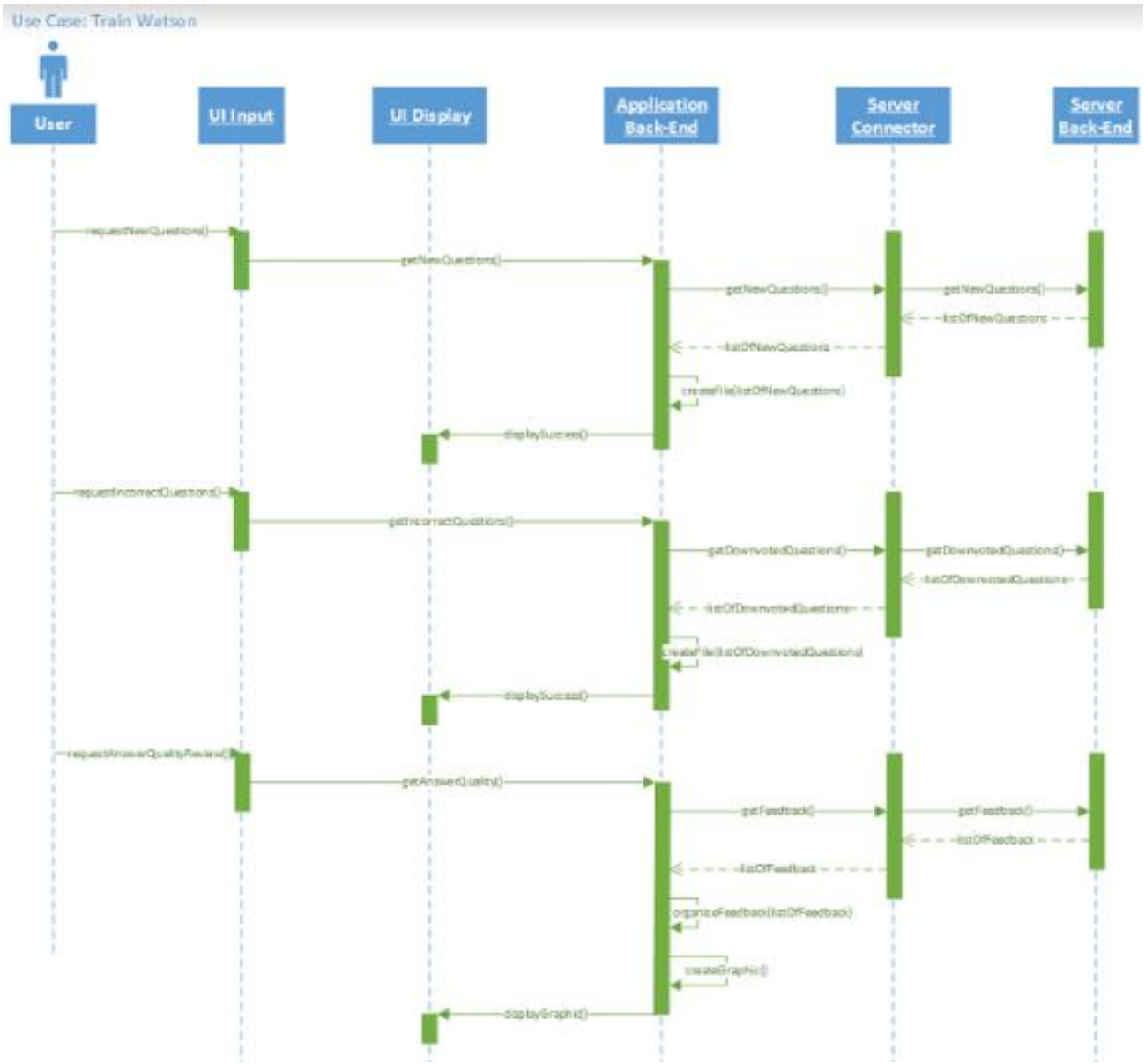
System Behavior



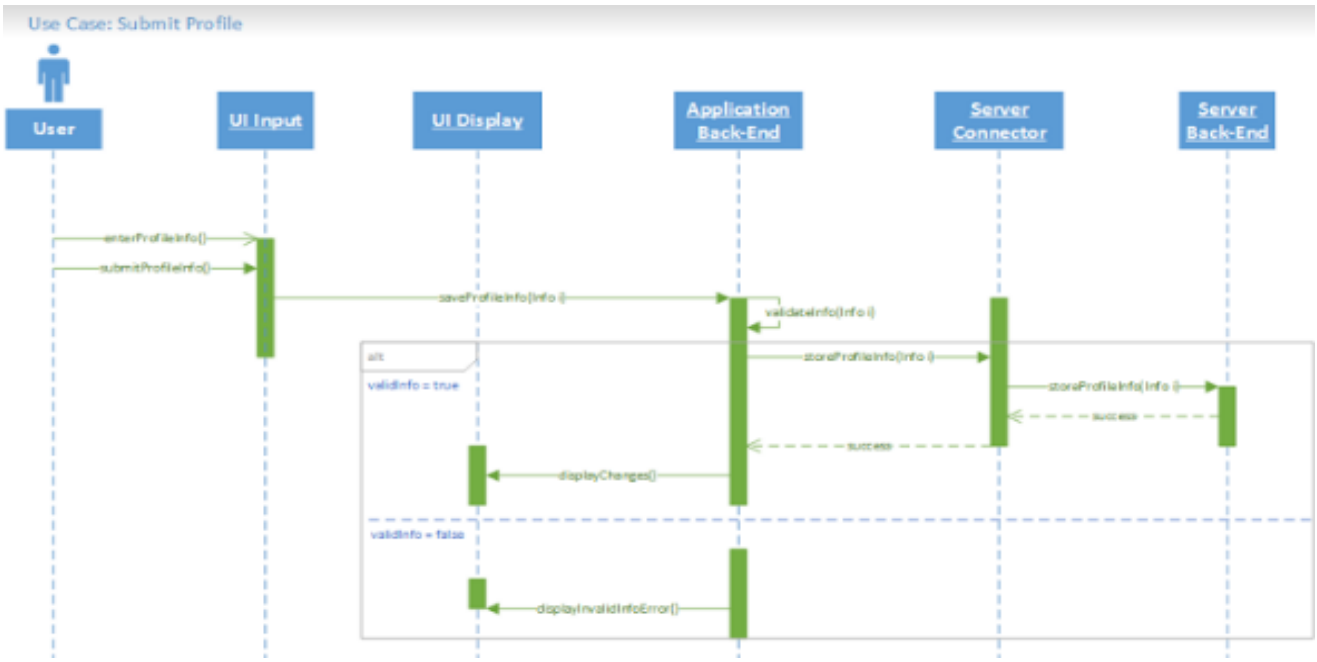
View Question Log Sequence Diagram: *After requesting to view the question log, the system will ask for filter info and then display the question log for viewing to the user after receiving the questions from the server's back-end.*



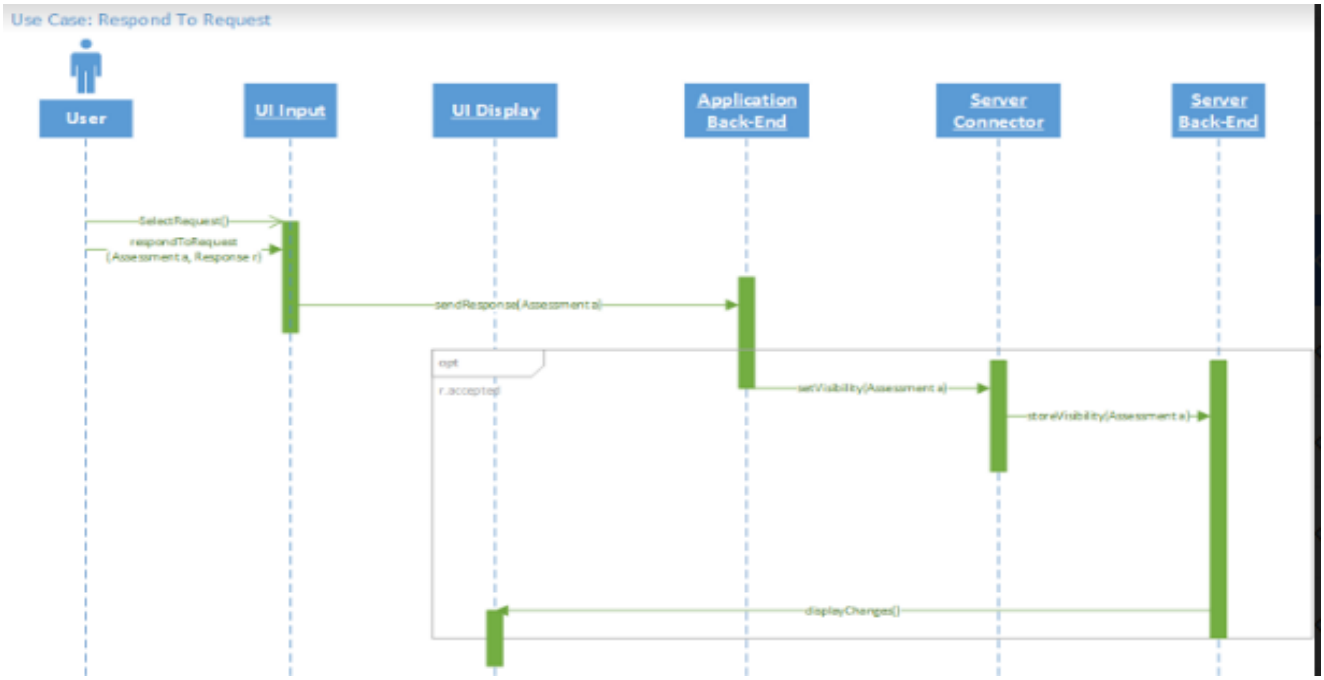
View Assessment Sequence Diagram: *After requesting to view an assessment, the system will display a list of viewable assessments. Once the user selects which assessment they wish to view, the system will display the assessment information on the screen.*



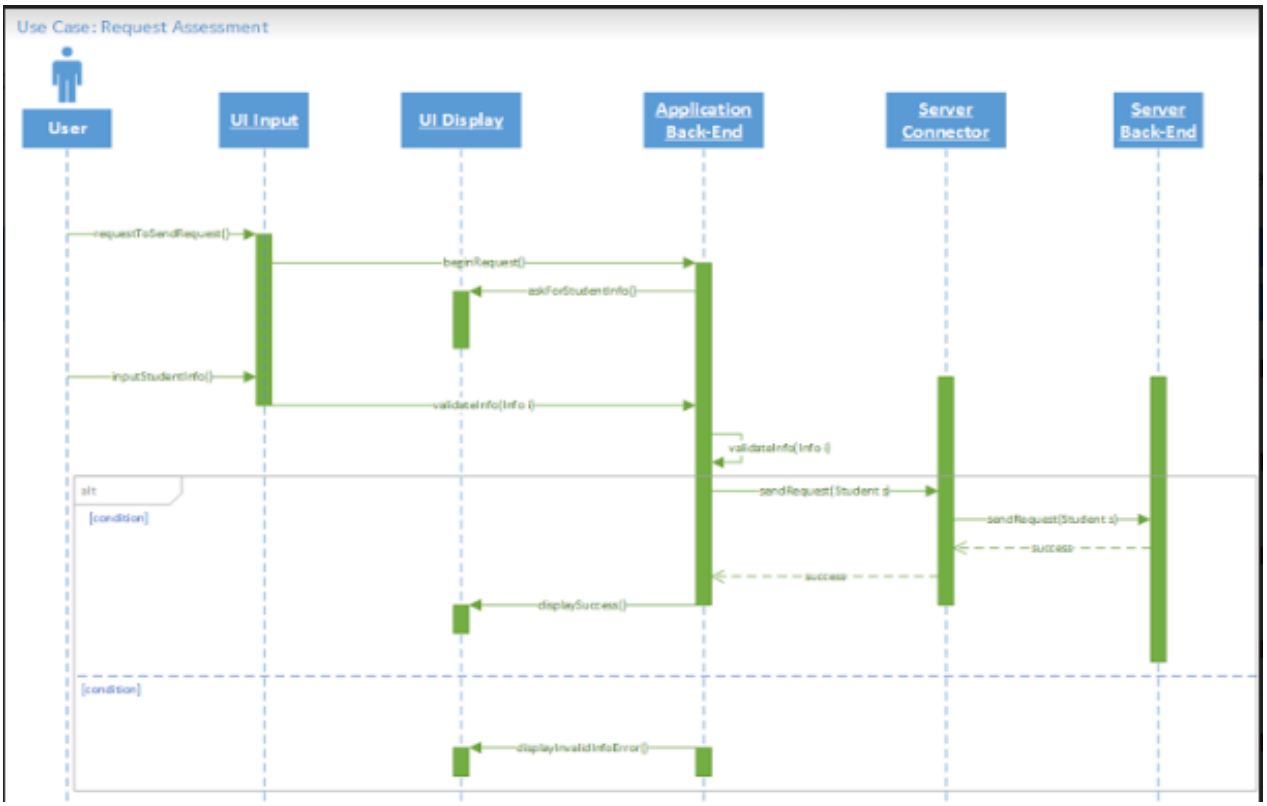
Train Watson Sequence Diagram: *After the user either requests new questions, incorrect questions, or quality feedback, the system responds by creating a file with a list of the questions matching the filter or a graphic (such as a bar graph) of the quality feedback.*



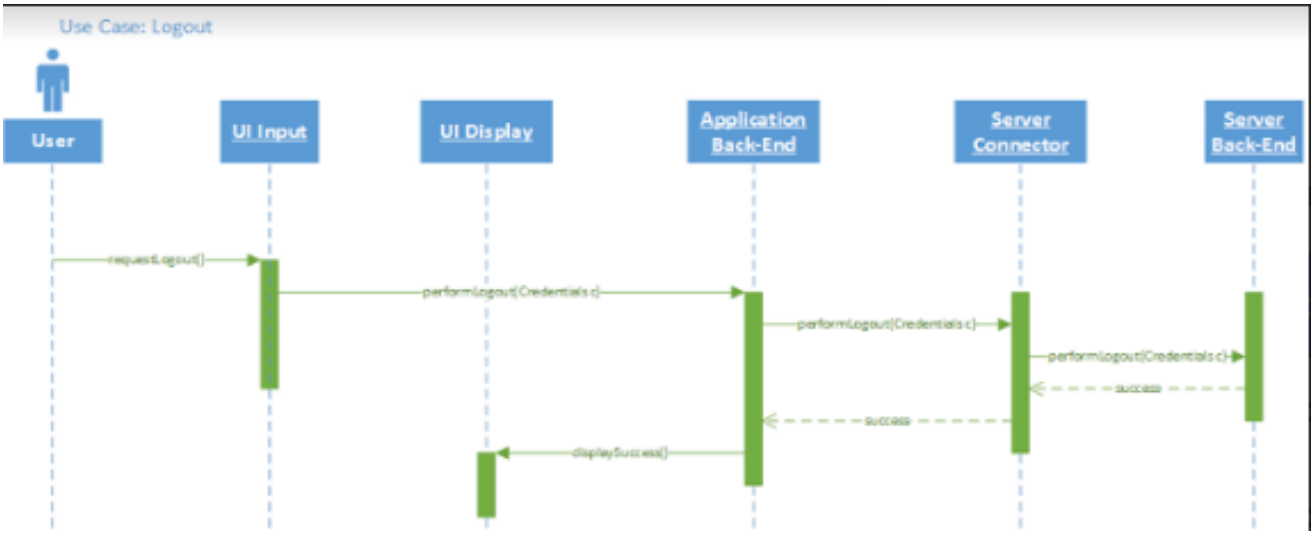
Submit Profile Sequence Diagram: *After a user inputs their profile information and requests to save, the system validates the information to ensure that there is no invalid or dangerous input. If it passes validation, the information is saved and a message stating that it was saved is displayed. Otherwise, a message displaying that there was a problem is displayed.*



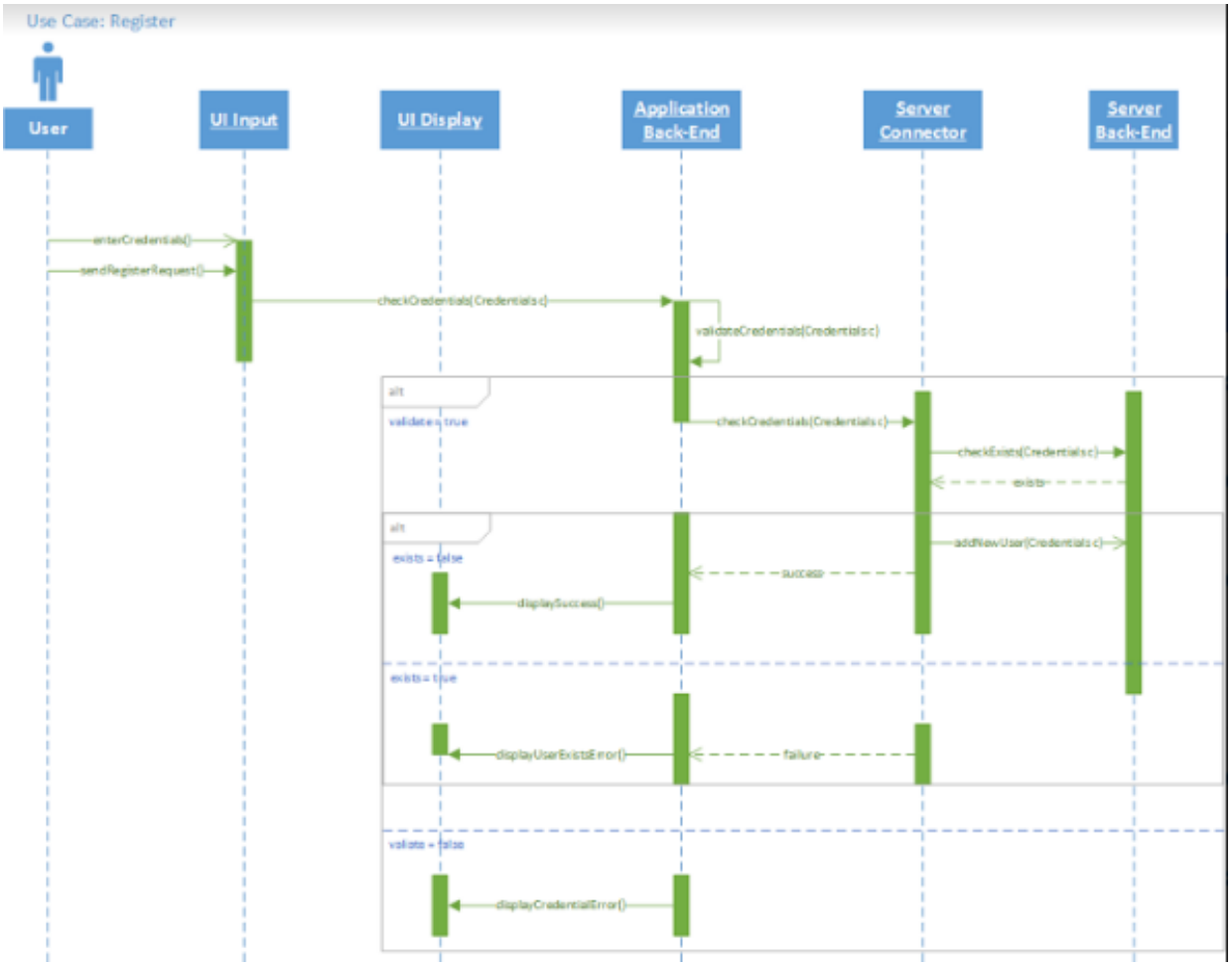
Respond to Request Sequence Diagram: *After the user selects the request they wish to respond to and submits their response, the system sends a message back to the requester, sets the visibility of the selected requested assessment*



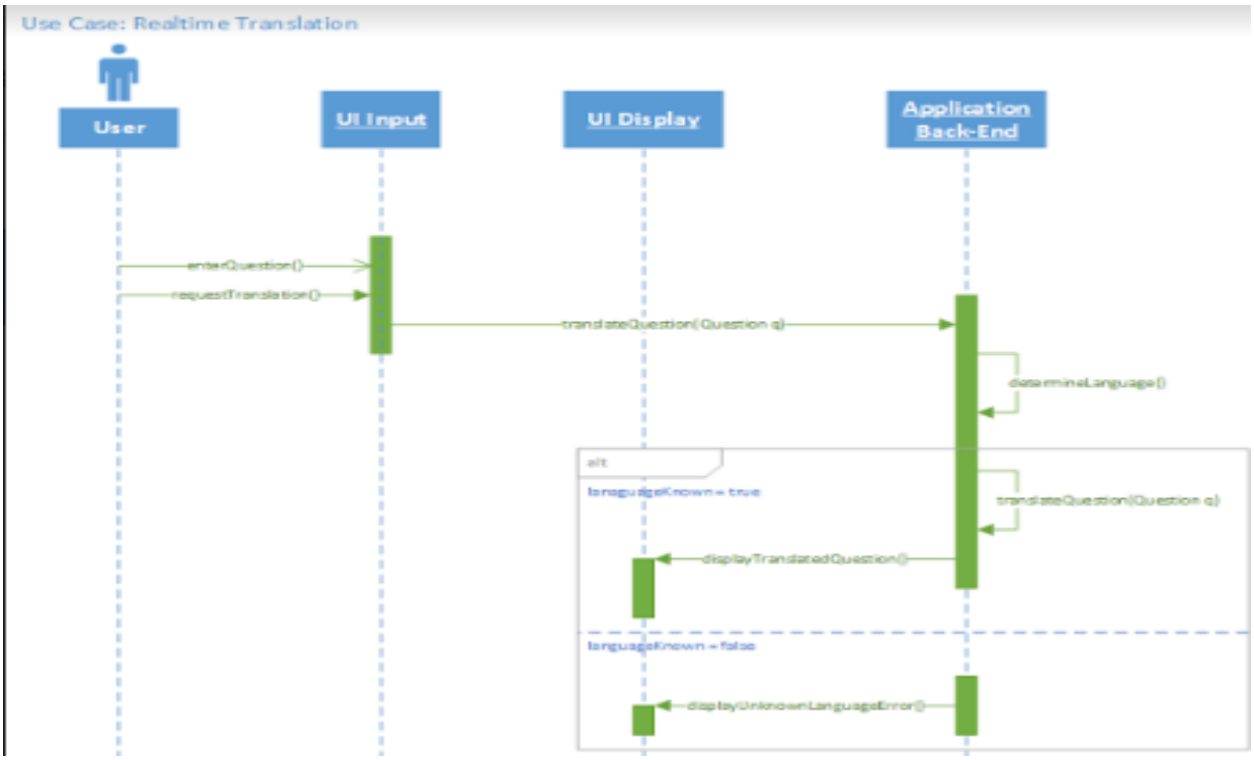
Request Assessment Sequence Diagram: *After the user requests to send a request, the system will ask for the student information. Once student information is inputted, it is validated to see if the student exists. If the student exists, the request is sent to the student and a success message is displayed. Otherwise, a message is displayed that the student could not be located.*



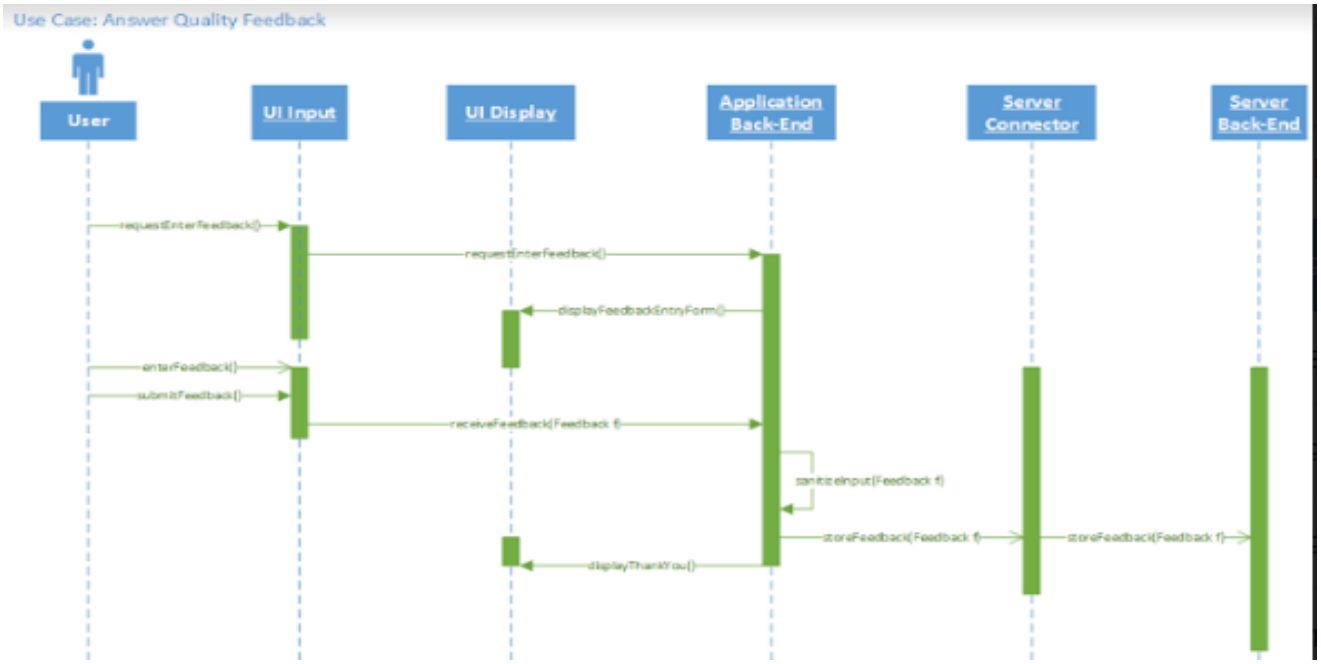
Logout Sequence Diagram: *After a user requests to logout, the system logs the user out of the server and displays success.*



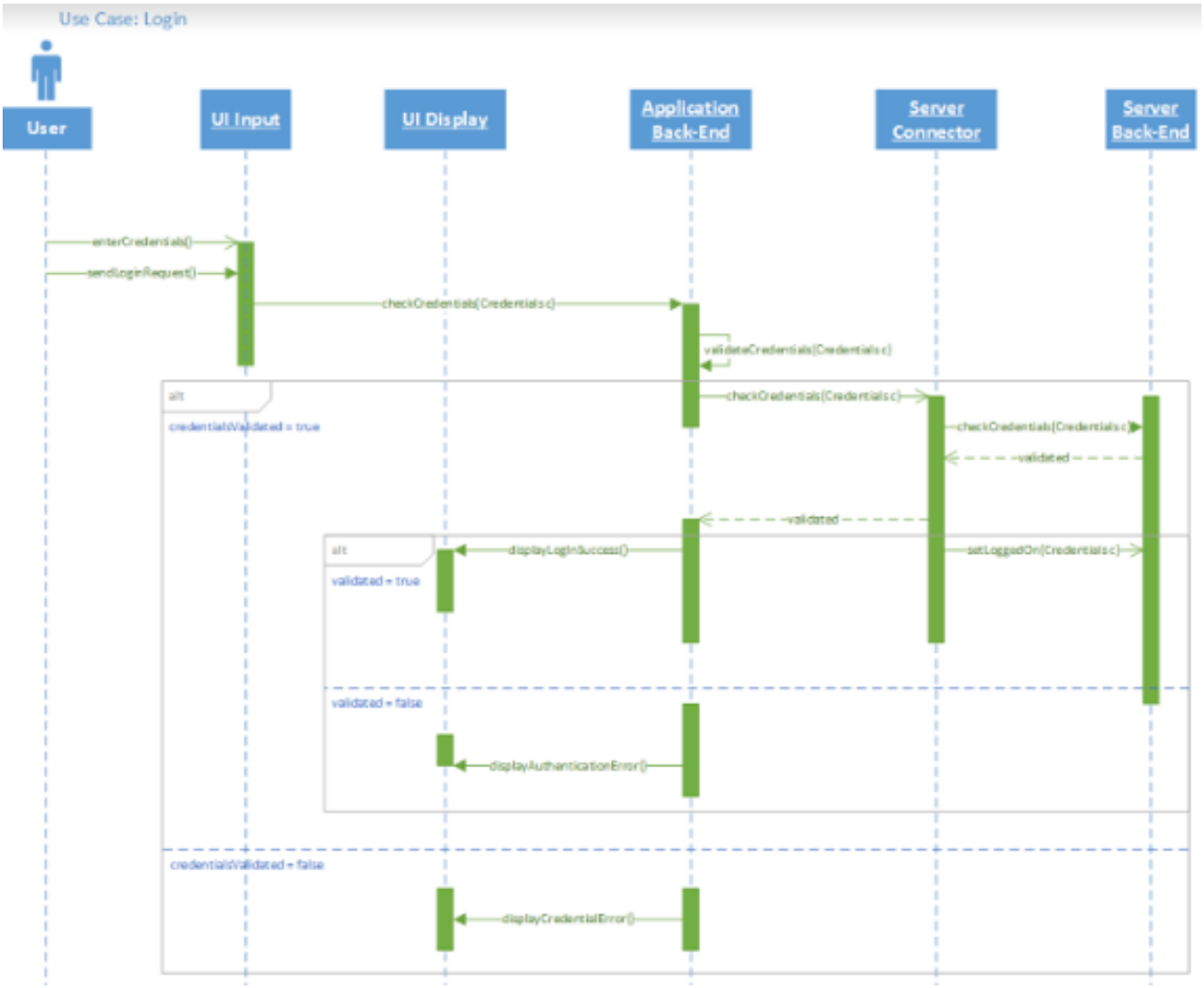
Register Sequence Diagram: *After the user enters credentials and requests to register, the system checks the credentials to see if they meet the requirements (email format, correct number of characters, etc.). If valid, the system checks if a user with the same email already exists. If a user does not exist, the system adds a new user to the database and displays success. If a user with the same email already exists or the credentials do not meet requirements, an error message is displayed.*



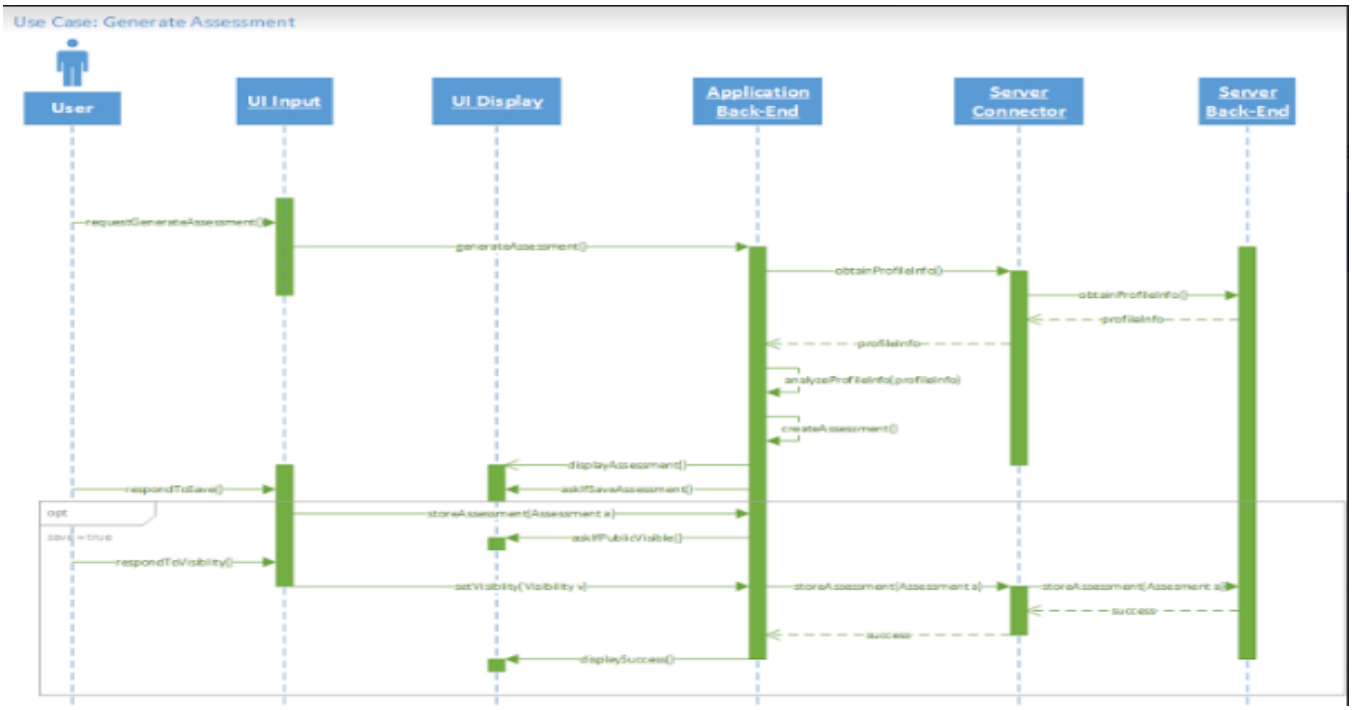
Real-time Translation Sequence Diagram: *After the user enters the question and requests a translation, the system determines the language of the question. If it can be translated, the translated text is then displayed. Otherwise, an error message is shown.*



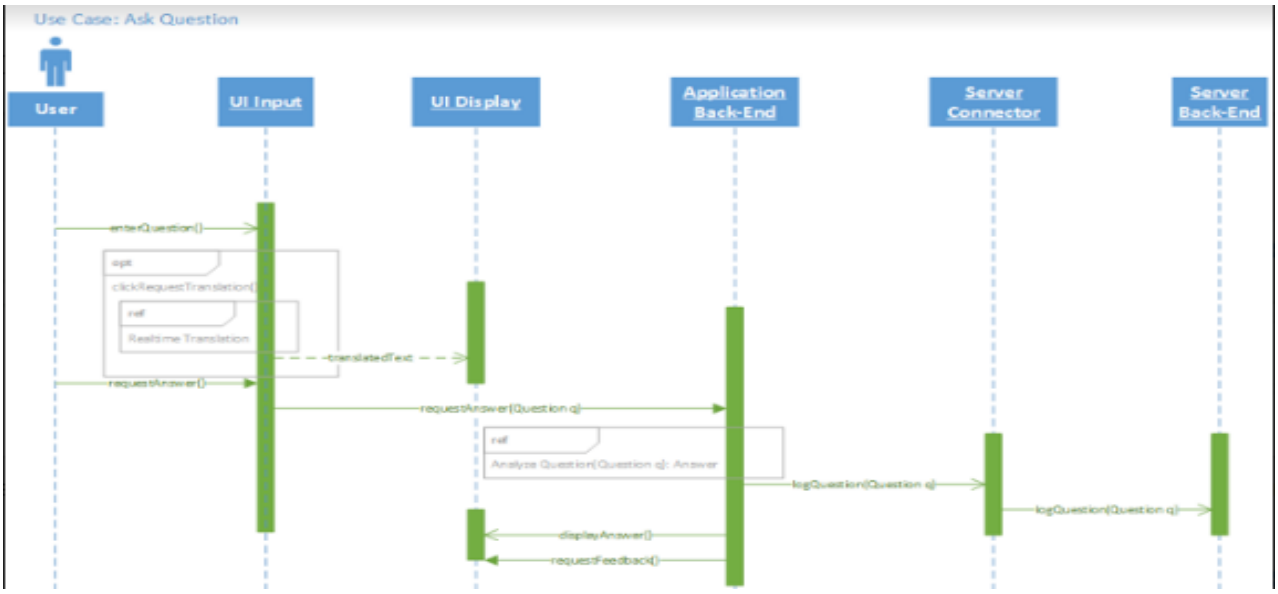
Answer Quality Feedback Sequence Diagram: *After a user requests to enter feedback, an entry form is displayed for the user to input the feedback. Once submitted, the system checks to make sure the input is not dangerous, and then stores it in the database, and finally displays a thank you message to the user.*



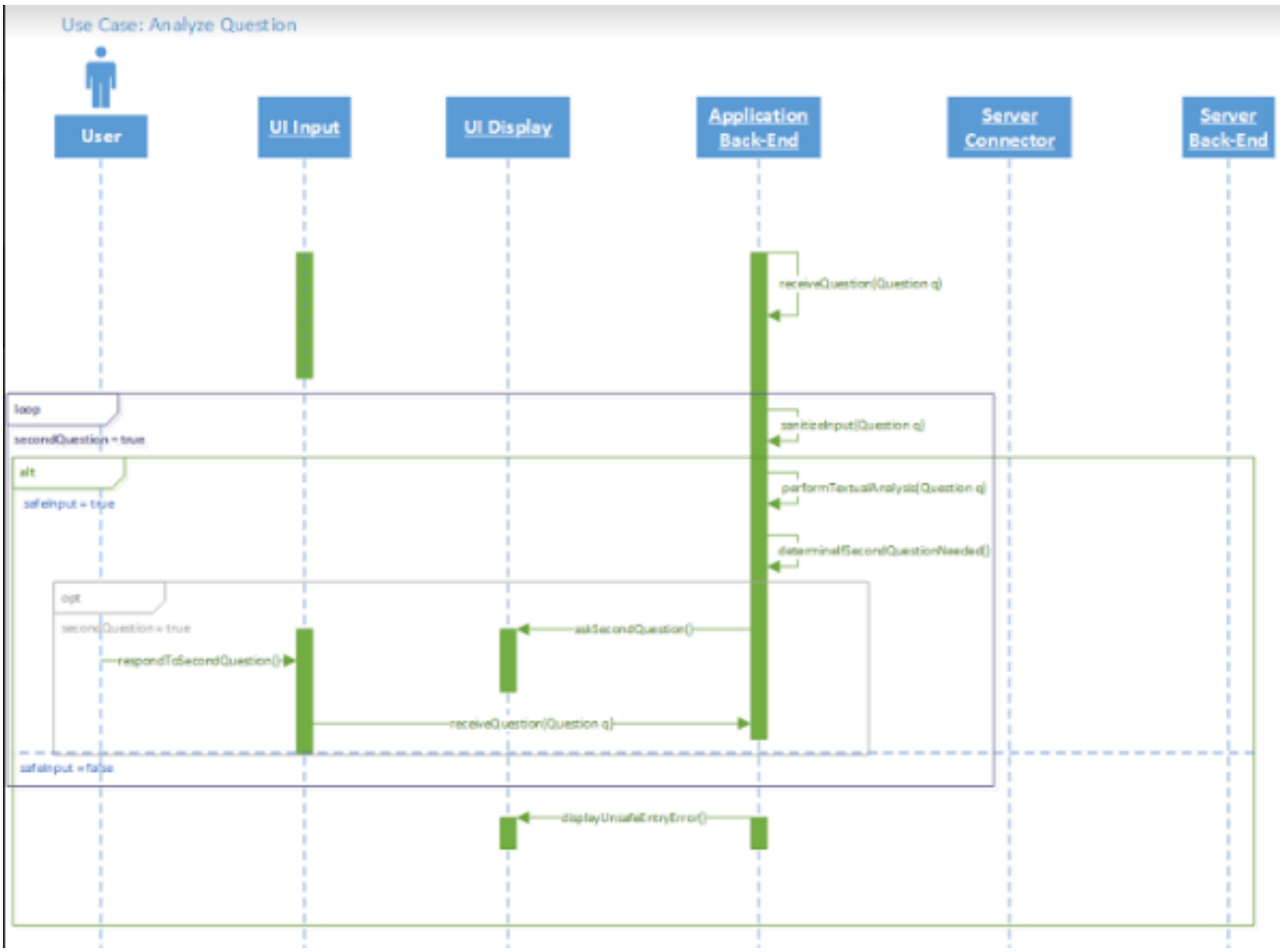
Login Sequence Diagram: *After the user enters their credentials and sends a login request, the system checks the credentials to see if they meet requirements first, and then checks to see if they are valid. If the credentials meet requirements and are valid, the user is logged on and a success message is displayed. Otherwise, an error message is displayed.*



Generate Assessment Sequence Diagram: *When the user requests to generate an assessment, the system obtains their profile information and analyzes it. Once analyzed, an assessment is created and displayed to the user. The system then asks if the user would like to save the assessment, if they do want to save then the system asks if they want it publicly visible. The publicly visible answer (true or false) is sent back with the assessment to the server to be saved.*



Ask Question Sequence Diagram: *When the user enters a question, they can request translation before requesting their answer. Once an answer is requested, the system calls to Analyze Questions and performs the functionality and returns the answer. The question is then logged and the answer is displayed. After the answer is displayed, the system asks for feedback.*



Analyze Question Sequence Diagram: *The system receives a question from the Ask Question functionality. First, the system checks to see if the question is dangerous. If it is not dangerous, a textual analysis is performed. After the textual analysis, the system determines if it needs to ask a second question. If a second question is needed, it displays it to the user and then awaits input. The process above is repeated until another question no longer needs to be asked.*

Data Requirements

- ➔ **UC001** - Input: User whose assessment is being requested.
- ➔ **UC004** - Input: User email address and real name for registration and login.
- ➔ **UC004** - Input: User password for registration and login.
- ➔ **UC006** - Input: Filter information for useful question log (e.g. major, minor, courses).
- ➔ **UC007** - Output: List of newly generated questions exported to a text file.
- ➔ **UC007** - Output: List of unanswered frequently asked questions.
- ➔ **UC009** - Input: Untranslated text.
- ➔ **UC009** - Output: Translated text.
- ➔ **UC010** - Input: User feedback pertaining to the relevance of the responses to their questions.
- ➔ **UC011** - Input: User questions.
- ➔ **UC011** - Output: Responses to user questions.
- ➔ **UC013** - Input: User profile.
- ➔ **UC015** - Output: User assessment based on their profile and search history.
- ➔ **UC019** - Input: The advisor who is requesting the student's assessment.

Non-Functional Requirements

Product: Performance Requirements

Table 4.43: Answer Question Within 5 Seconds Requirement: This system requirement requests that when we are ensuring that an answer is providing quickly, we should test that it answers within 5 seconds. Low priority was given to this requirement because we are focusing first on implementing features and then focusing on quality and performance of the features.

Product: Dependability/Reliability/Security

Table 4.44: Secure Profile Requirement: This system requirement requests that when we are ensuring that a profile is secure, we should test that it only displays information that the user has set to be public. High priority was given to this requirement because it should be done while creating the functionality of the profile and we were encouraged to keep security in mind.

Organizational: Development Requirements

Table 4.45: Manage Session Requirement: This system requirement requests that when we are ensuring that a user's session is managed, we should test that it logs the user out after 1 hour of inactivity. Low priority was given to this requirement because while it is connected to logging in, logging out, and registering, it is not critical to other functionality of the system.

4.3. Requirements Trace Table

See Table 4.46: Summary of Requirements in Appendix R.

5. Exploratory Studies

5.1. Relevant Techniques

Our team distributed a survey, containing both qualitative and quantitative questions, to 150 students in an introductory Computer Science course. Our intention was to receive feedback relevant to the thoughts and decision-making processes of a student who has recently completed high school and is considering a degree and career in Computer Science, Software Engineering, or Computer Engineering. We also accepted the input of upper level classmen with the belief that their input could provide important feedback about the information they have gathered and decisions they would have made, in hindsight.

- ➔ Qualitative Data: Open-ended questions "can lead to the discovery of new initiatives or problems that should be addressed." [8]
  - ◆ If your major/minor has changed, what was it and why did it change?
  - ◆ What are the best and worst features of your field of study?
  - ◆ Why did you choose to study at Behrend?
  - ◆ What questions/concerns did you have when deciding on your major and school?
- ➔ Quantitative Data: Closed-ended questions "allows researchers to categorize respondents into groups based on the options they have selected." [17]
  - ◆ What year are you? (Freshman, Upper)
  - ◆ What is your major and minor?

While many of our questions were based on the responses of the surveys, some students did not return their survey, some students did not ask five questions, and many students repeated questions asked by their peers.

$$(150\ surveys\ \times \frac{5\ questions}{1\ survey} - duplicate\ questions)\ \leq 750\ unique\ questions$$

This is a generous calculation of how many questions we were able to create based on the surveys.

Many additional questions were created through research conducted on the internet. Watson will need to be trained with at least 1000 question-answer pairs to ensure depth and accuracy of system knowledge. Additionally, Watson will be trained with variations of each question, to improve accuracy, should a user of our system phrase a question in a different manner than the question originally asked. References 18-52 were used to generate both questions and answers for our domain.

- ➔ The following are examples of question-answer pairs:
  - ◆ Q: Does Penn State Behrend assist students in finding internships and employment opportunities?

    A: Penn State Behrend holds a career fair once a semester where a hundred or so different companies come and you can speak with them. In addition, once you post your resume to Nittany Lion Career Network, the career center will start sending your resume out to potential employers,

who will then contact you without you having contacted them yourself. Kinda nifty, in my opinion.

- Q: Do Computer Engineers get hired by companies like Google, Microsoft, Amazon, and Facebook?

  A: Yes. All four of those companies require computer engineers, but they have much more rigorous interviews than most companies. You can get hired, you just need to be very well prepared for the interviews.

Our system will use keyword-concept extraction to update the frequency of each question being asked. This will continue recording into the database, for future analytical use. Our system will use machine learning algorithms to format, structure, and extract data in order to update question-answer tracking and add new useful information, if is not currently in our knowledgebase. Similarly, by crawling data from user feedback, we can track users' questions to check whether they are related to our knowledge domain, and either update the related information or add new data from the user input.

- ➔ Data Crawling
  - Question Log: update frequency of each asked question, extract new keyword and concepts to add to our knowledge domain.
  - Suggested Feedback: answers for unclear questions - updating better answer to our knowledge domain.
  - Analysis Data: data gleaned from user's input.

Our system should be able to interpret questions delivered in Natural Language. We will use tags and information extraction to ensure that the user's question is being addressed and understood properly. For example, our system will understand that the word "what" is indicative of a question. Our system should also be able to comprehend "questions" asked in a non-standard manner. For example, an input of "Penn State retention rate" should be interpreted as a question rather than a statement.

- ➔ Natural Language Processing (NLP) [9]
  - Generative Models for Parsing
    - Parse Trees
      - Part-of-Speech
      - Useful Relationships
    - Context-Free Grammar
  - Log-linear Taggers
    - Information Extraction
    - Named Entity Recognition
    - Relationships between Entities
    - Named Entity Extraction as Tagging

Our documents must be curated to ensure that the information our system is providing is reliable and accurate. This can done by feeding a variety of informational documents to our system and allowing the system to verify the information's validity through comparison of the sources.

- ➔ Document Curating [10]
  - Exact match search

- ◆ Wildcard search
- ◆ Proximity search

We plan to use a basic exhaustive search and complete training with fixed training data (our knowledge base). Our system will use a natural language classifier to determine what information is useful. This can be substituted by the Conversation API from Watson.

- ➔ Machine Learning Algorithms [11]
  - ◆ Supervised Learning
    - Linear Regression
    - Decision Tree (classification)
  - ◆ Unsupervised Learning
    - Gensim (python)
      Use to compare string similarity
    - NLTK (python)
      For data mining purpose
    - MonkeyLearn (Web)
- ➔ Use web service to build automated training model

5.2. Relevant Packages/Products

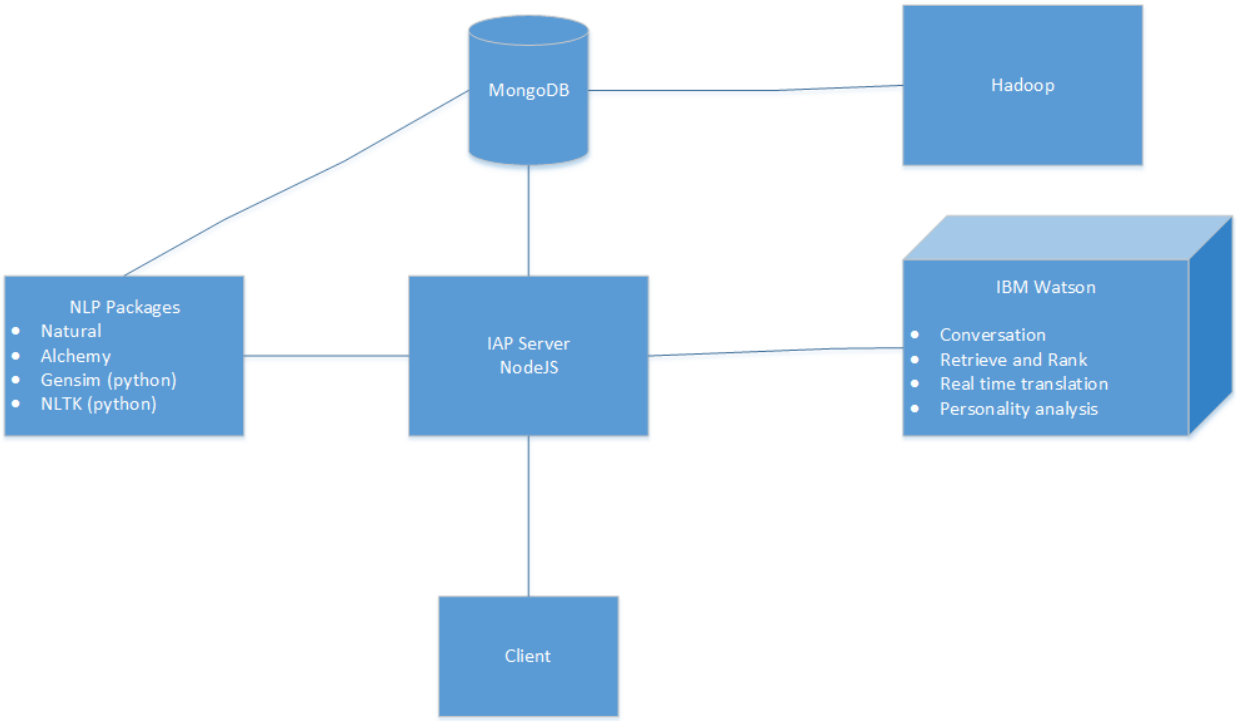
Watson services are accessed by APIs usually programmed in the CURL, Java, or NodeJS languages. Since this project is a web application, NodeJS has more native support on the server side, and faster performance in general execution and implementation. A NodeJS server will call Bluemix RESTful Watson APIs on server usage and query MongoDB natively in the format of JavaScript. The application will have NLP handling and analysis functionalities embedded. The development team could also use Hadoop for analysis, since it is especially good for managing Big Data. Since the intention is to maximize usage of Watson services, the best option for the team is to use Alchemy. MongoDB will store the data of three perspectives: user account, user question log, and analysis of all kinds of records that are needed by administration.

- ➔ NodeJS[12]
- ◆ Server application type
- ◆ We decided to use this because most of the Bluemix APIs are designed for Node.JS handling and Node.JS is an advanced, efficient server application.
  - ➔ Watson Platform
- ◆ Conversation[5]
  - Mainly used to classify the question input from user and determine if the question is in our domain or not.
- ◆ Retrieve and rank[4]:
  - Core functionality of the project
  - Uses machine training and NLP(Natural language process) to study the training dataset and provide feedback on the answer with an accuracy measurement.

- Real time translation[15]
  - Allows instant translation between common languages.
  - Displays desired language based on the user's account preference.
- Personality analysis[16]
  - Assessment functionality to generate personality analysis for user. This will allow us to give more specific answers to a user based on their personality.
  - ➔ MongoDB[14]
- Main non-relational DB for the application, since we will handle a lot of unstructured data.
- MongoDB has good features to store and sort.
  - ➔ Natural language processing
- Monkey learn[2]
  - A mature public accessible API provider to do NLP analysis
- Natural[9]
  - Equivalent NodeJS library of NLTK(Natural language toolkit) in python. It mainly focuses on customized NLP classification.
- Alchemy[3]
  - Similar to Monkey learn, but is managed by IBM. It will provide feedback on the detailed analysis of a sentence, so we can analyze parts of sentences to determine the intent of the question.

➔ Hadoop

- The core of Apache Hadoop consists of a storage part, known as Hadoop Distributed File System (HDFS), and a processing part called MapReduce. Hadoop splits files into large blocks and distributes them across nodes in a cluster.



Relevant packages/products flow diagram

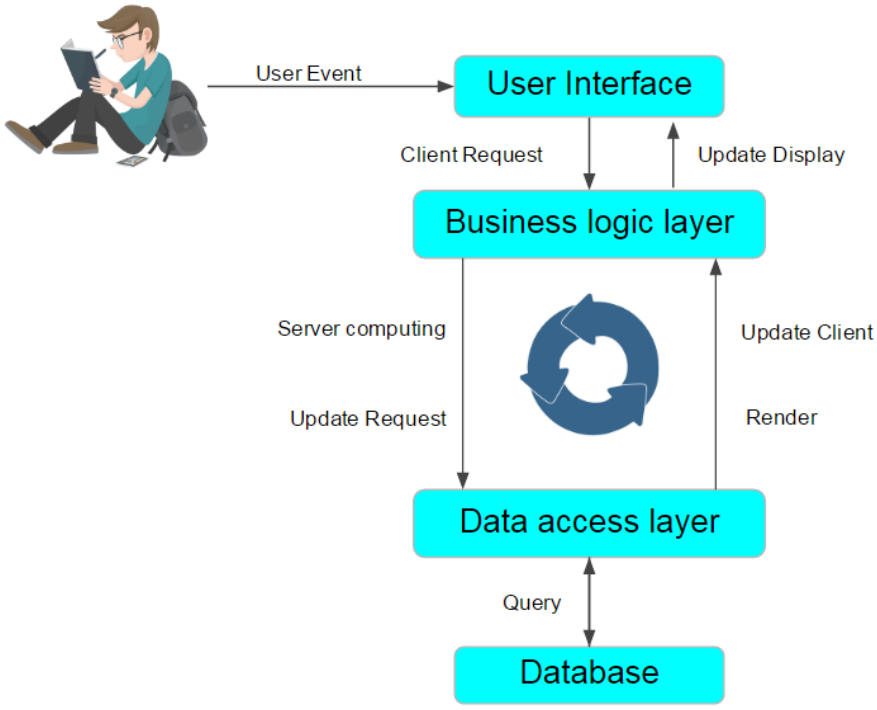
5.3. Broader Impacts

This project can be expanded to include other majors within Behrend or branches of Penn State University. On a grand scale, this type of tool would be beneficial for high school juniors and seniors, as well as college freshman and anyone participating in the role of student advisement.

6. System Design

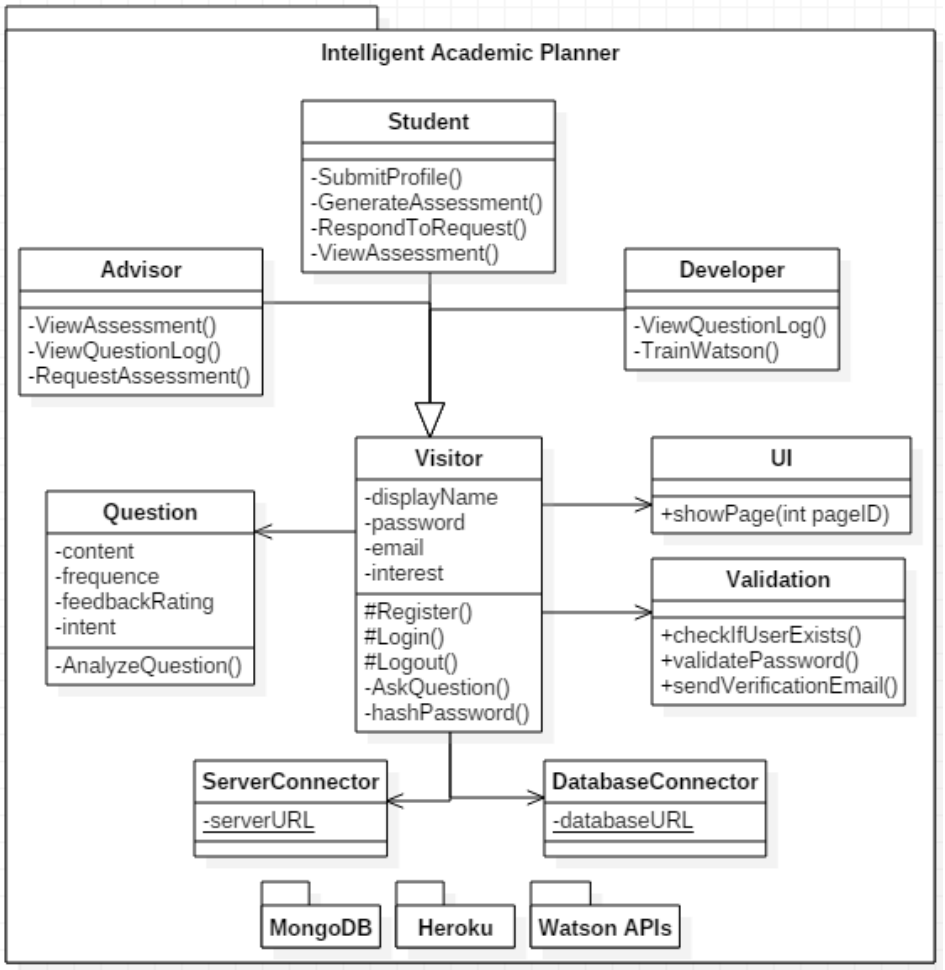
6.1. Architectural Design

- ➔ Layered Architecture
  - ◆ Client-Server-Database model
  - ◆ Server handles all computation and updates
  - ◆ Server query with DB
  - ◆ Feedback data to client side



User initializes a “submit question” event on the web browser, then the server will handle the question through the logic layer. The database can then be queried and, if there are no exceptions, the data will be sent to Watson through Bluemix RESTful APIs. Finally, the response information will be returned in the reverse direction.

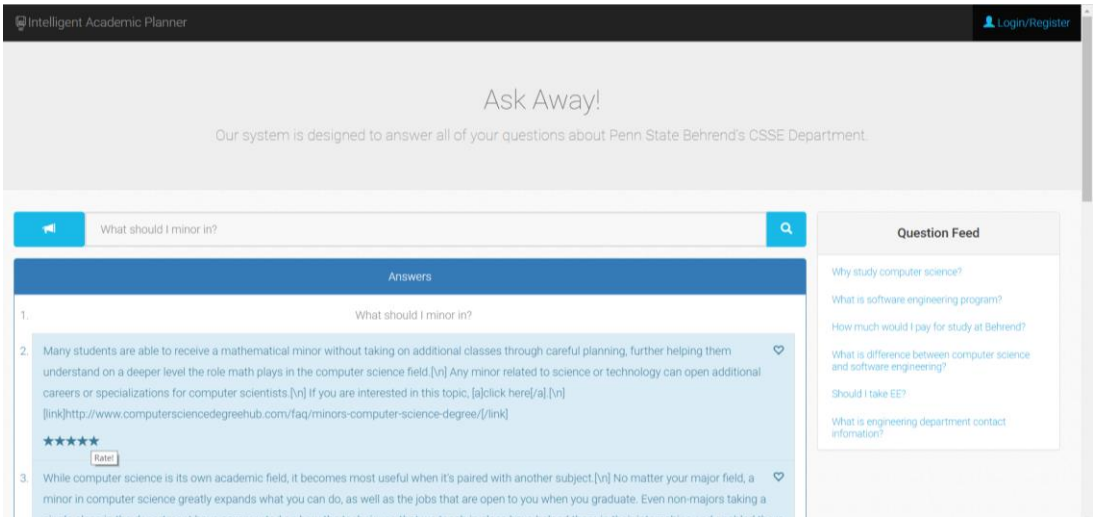
6.2. Structural Design



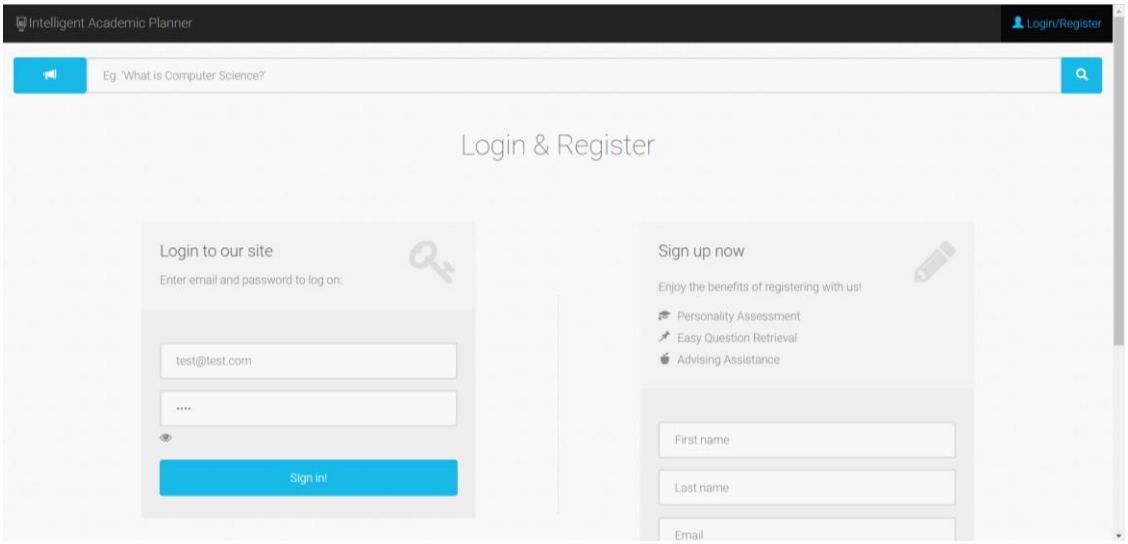
This is the class diagram of our system. We are using Heroku as our server and MongoDB as our database. All visitors have the ability to ask questions, and questions are stored on our server. The Validation class is used to check for safe and correct inputs, improving security of our system.

6.3. User Interface Design

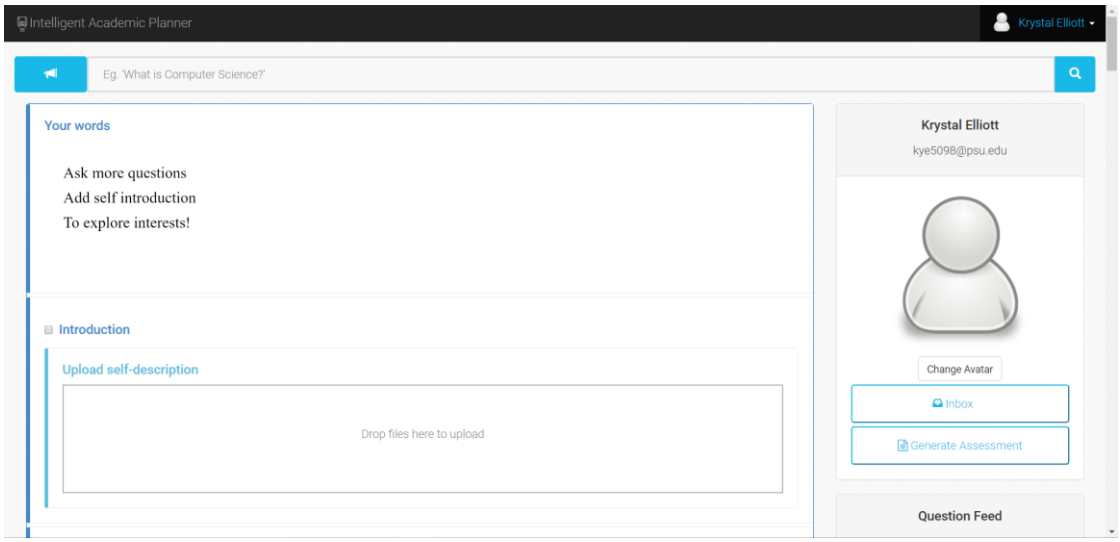
The user interface is split into several web pages: Ask Questions, Login & Register, and User Profile. The design is responsive and can be viewed on both desktop and mobile devices. A navigation bar is present on each page to ensure easy navigation.



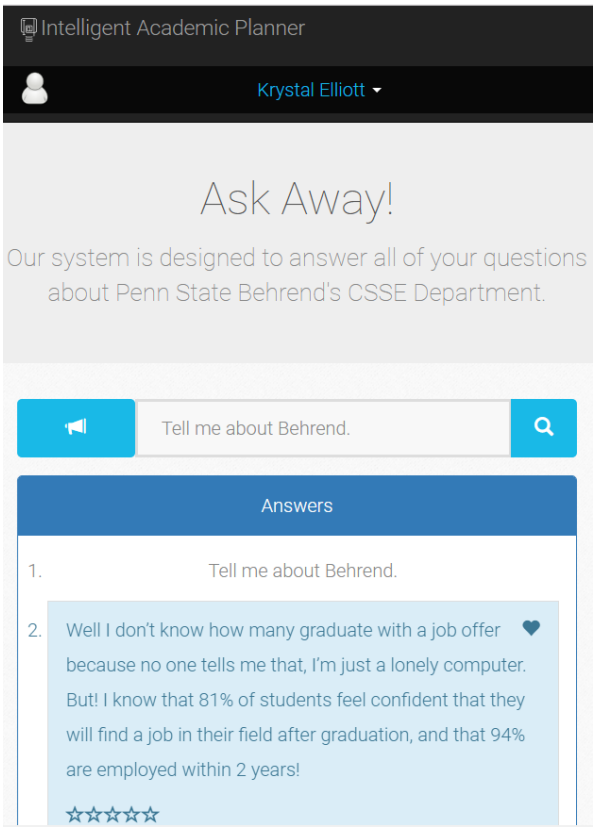
Ask Questions: This is the landing page of our site. A user can ask a question in the search bar, using voice-to-text or textual input. The answers are displayed in list form, with the ability to "Favorite" a preferred answer and give the system feedback through a 5 star rating system. A question feed column is placed along the side of the page to inform users of recently asked questions and to allow them to view their responses.



Login & Register: A search bar is at the top of the page to allow a user to ask a question from any page. The user must register with their name, an email, and password. The user must also choose an account type of Student, Advisor, or Developer. A user can register with their Facebook, LinkedIn, or Google account.

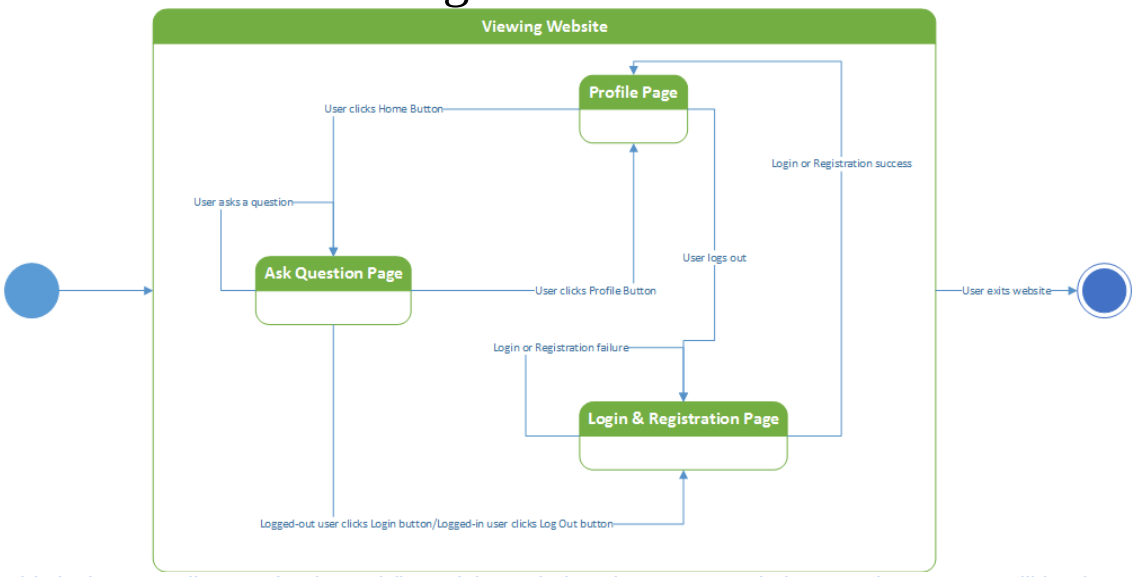


Profile: Only available after login, display information about the user, including: a conglomeration of words describing the user's interests, a profile picture, their favorited questions, and an introduction written by the user about the user. This page also features the Question Feed and Search Bar.



Responsive Design: This is an example of the view of the Ask Question page on a mobile device.

6.4. Behavioral Design



This is the state diagram for the webflow of the website where users ask the questions. Users will begin on the Ask Question page. If the user clicks the login/register button, they will be taken to the login/register page. When login or registration completes, they will be taken to their profile page. From any location in the system, if they click on the home button, they will be taken to the Ask Question page and if they click on the logout button, they will be taken to the login page.

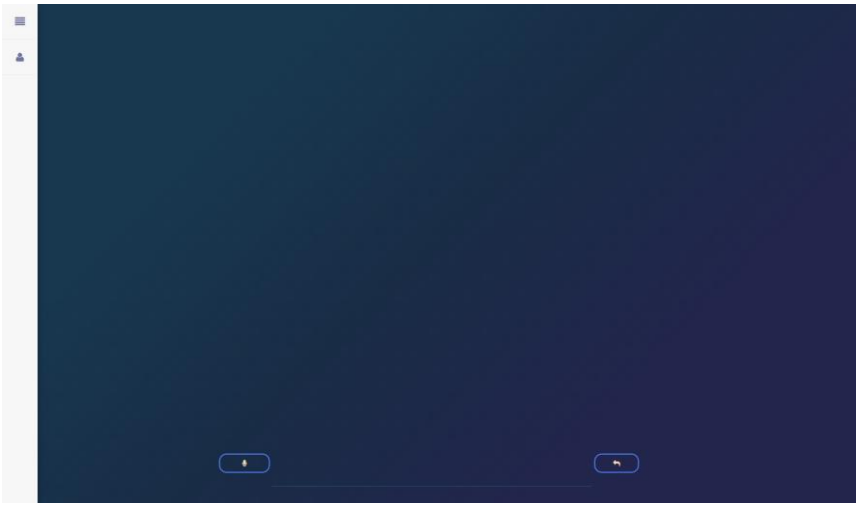
6.5. Design Alternatives & Decision Rationale

UI Alternative Designs

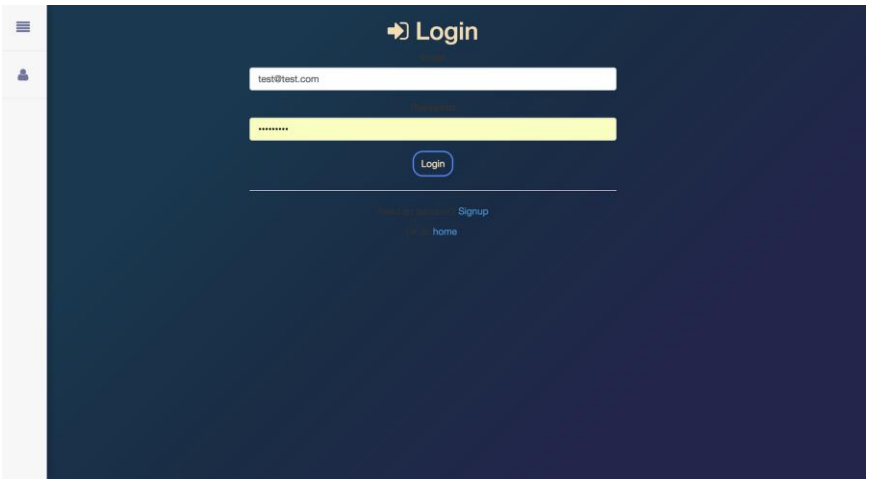
These are alternative user interfaces.

Version 1.0

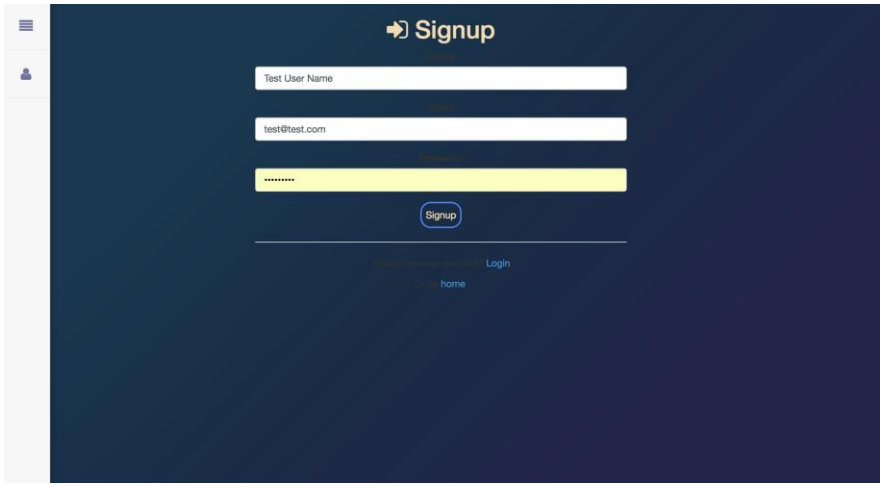
The view will be divided by 2 main sections, left side navigation of functionalities and right section of display/interactions. The user enters the question on the line at the bottom and either hits enter or the arrow button to submit the question. As the question is being answered, a loading icon appears. The user can login or view their profile using the buttons on the left side.



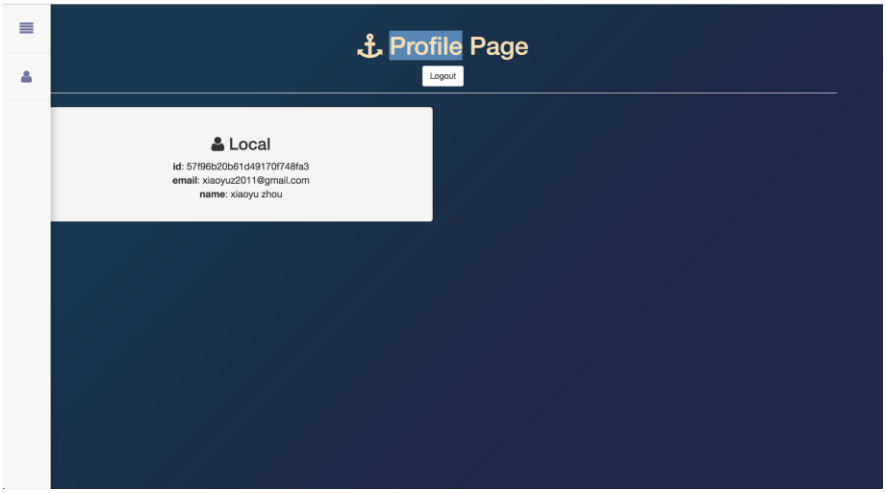
***Ask Question**: use mobile compatible view design template*



***Login**: require register email and password*



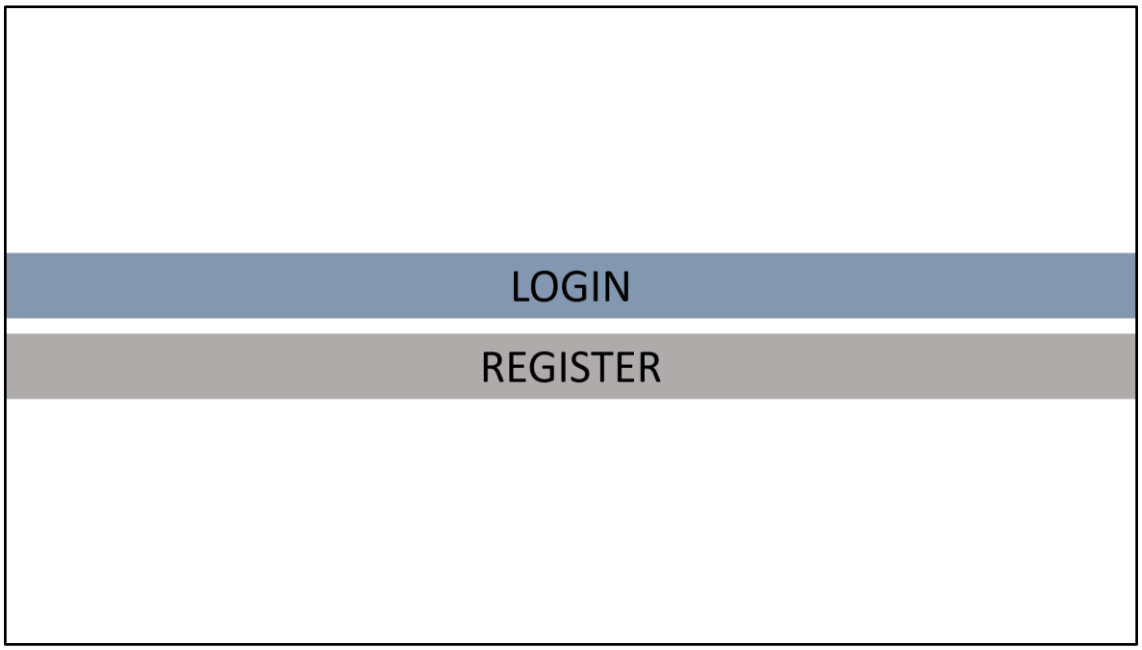
***Register Account**: require a display name, register email, and password*



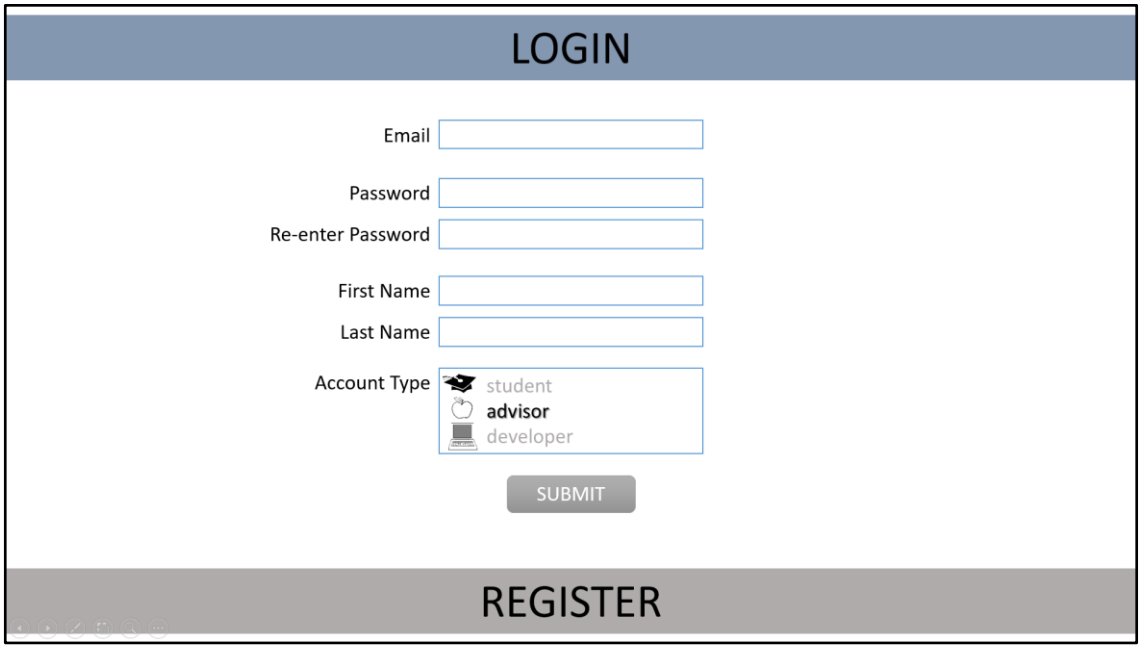
***Profile**: only available after login, display regular account information*

Version 2.0

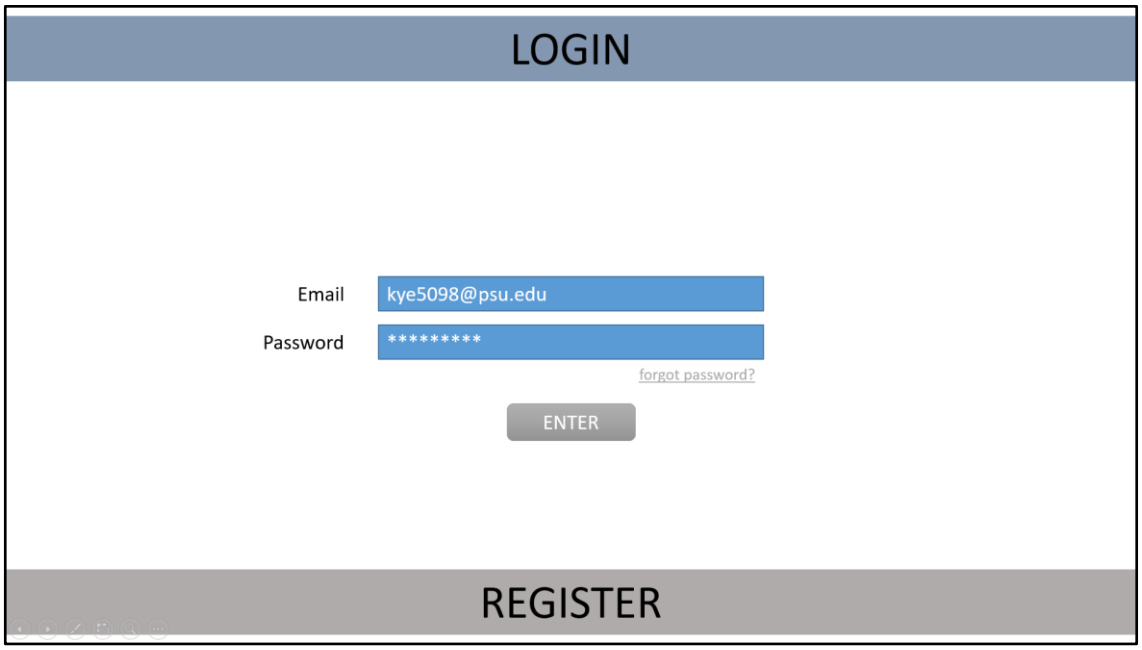
The color scheme is much brighter and it includes additional screens, which have not been included in the original UI design. This design is fairly simplistic and would allow for a user to access the project via their mobile device, as well.



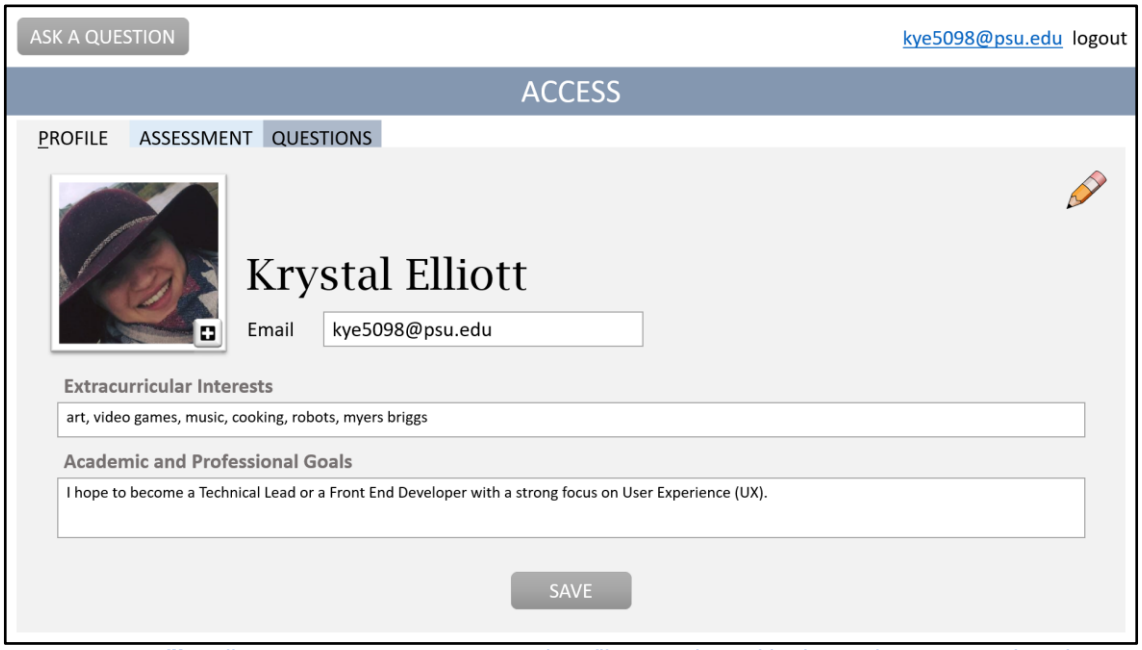
Landing Page: *offers options of logging in and registering*



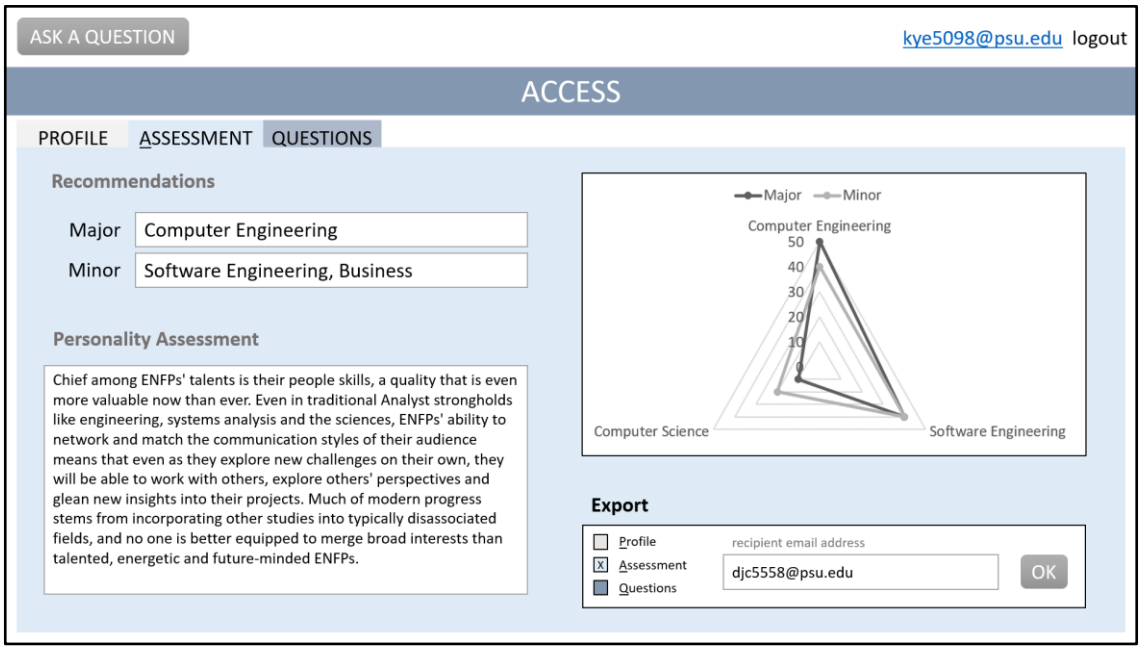
Registration: *collects user data to create an account*



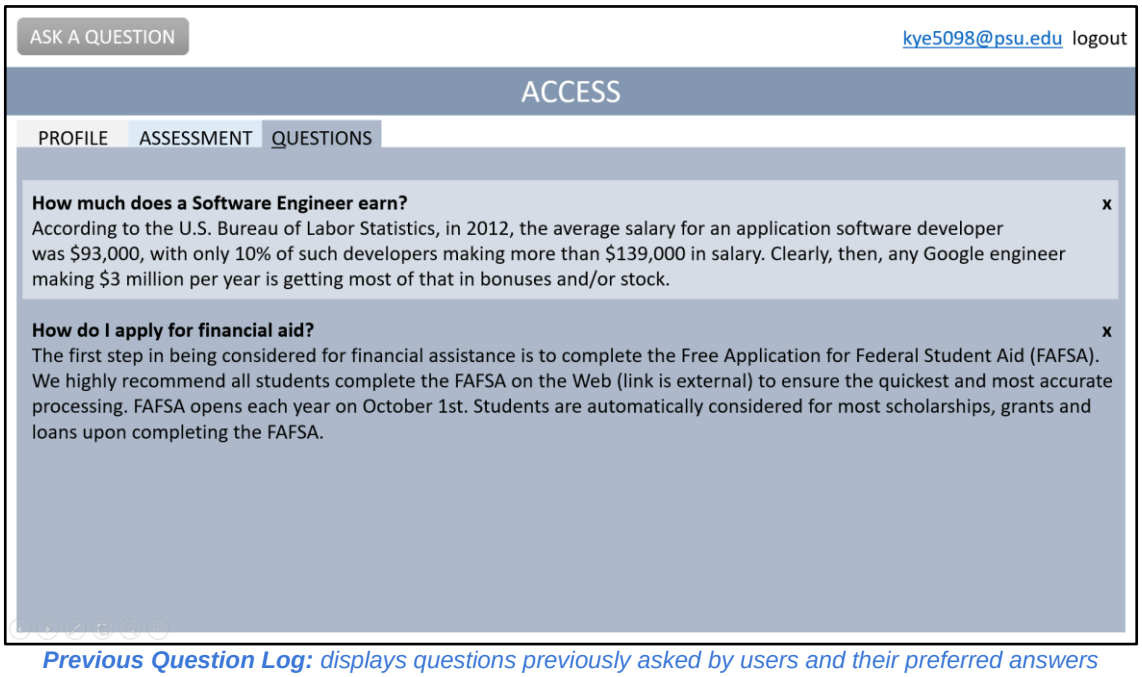
Login: *allows user to login*



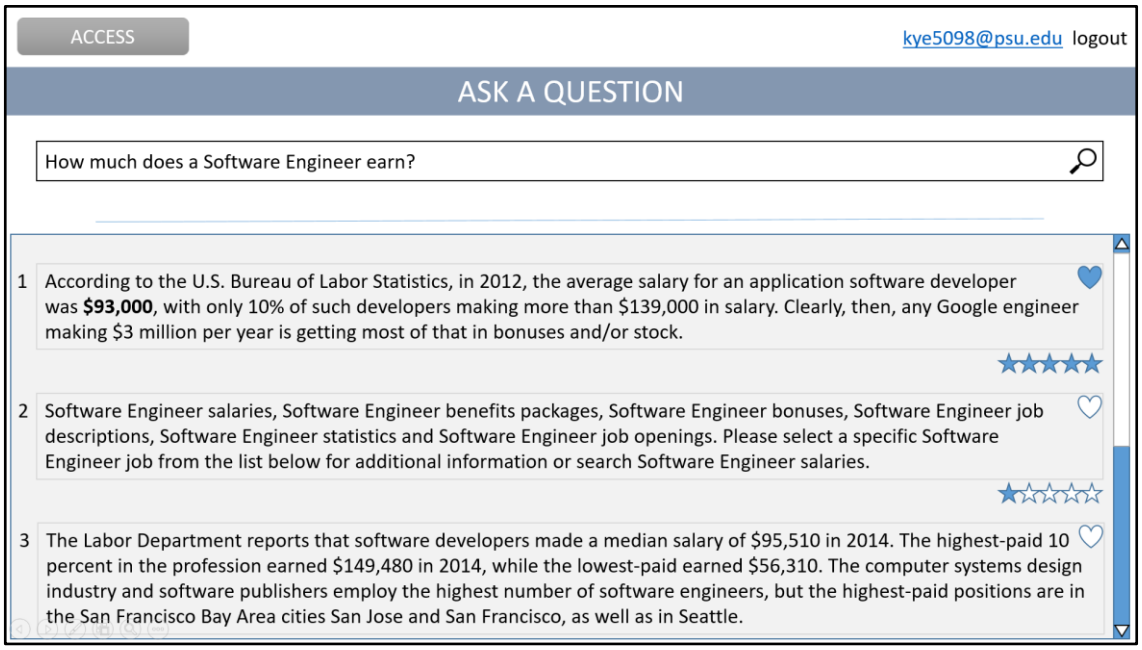
User Profile: *allows user to create a personal profile, complete with picture, interests, and goals*



Assessment: *allows user to view their personalized assessment and to send it, partially or in full, to an email address.*

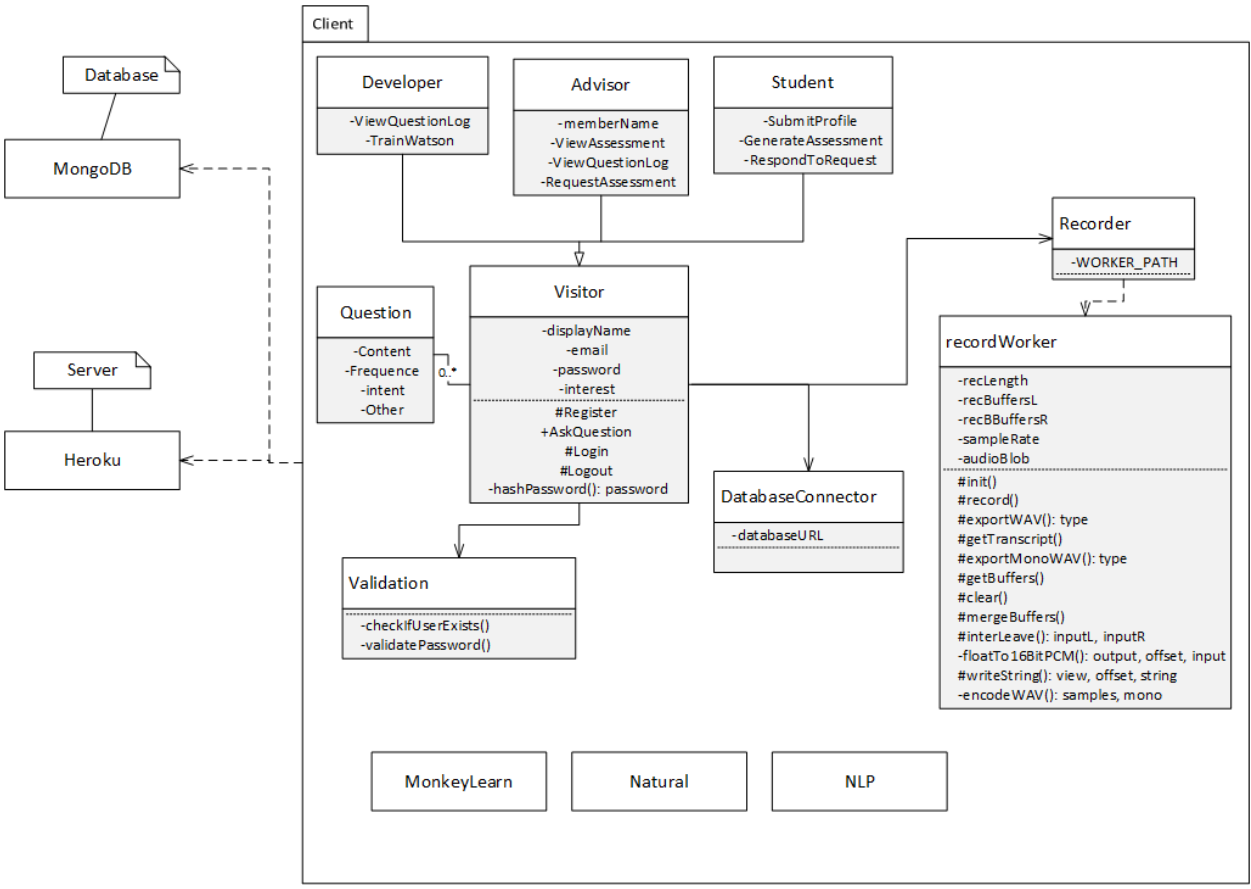


Previous Question Log: *displays questions previously asked by users and their preferred answers*



Ask Question Interface: *allows user to ask questions, rate the received answers, and select one preferred answer*

Structural Alternative Designs



➔ This design was not chosen because it places a lot of emphasis on the recorder and recordWorker. In addition, they are used for the voice-to-text functionality that we decided to not include.

Architectural Alternative Designs

➔ No alternative considered.

Behavioural Alternative Designs

➔ No alternative considered.

`

7. System Implementation

7.1. Programming Languages & Tools

- ➔ BlueMix (conversation, retrieve and rank, alchemyAPI,…)[3][4][5]
  - ◆ Input will be parsed by the server, then go into Conversation service to be guided by default classifier, then, if no exception, the question will be answered by retrieve and rank. In parallel, the question and answer will be analyzed by Alchemy, and placed into the database.
- ➔ MongoDB[14]
  - ◆ The MongoDB is mainly used to maintain user information, question and answer information, and system analysis.
- ➔ Node.js[12]
  - ◆ Server application type, programing language in JavaScript style. For production driven development, all libraries will update to the newest version. And programming style will be in industrial standard.

7.2. Coding Conventions

- ➔ Use ES6 standard for JavaScript

7.3. Code Version Control

- ➔ GitHub[13]

7.4. Implementation Alternatives & Decision Rationale

- ➔ PHP for Node.Js
  - ◆ NodeJS also has the advantage of an easy setup in a local development environment, which is preferred by the development team. Allen has had some experience developing with NodeJS in the past, so he will be able to guide his teammates, should they have questions or need guidance.
- ➔ MySQL for MongoDB
  - ◆ Although MySQL is more familiar to all of us, traditional databases are not efficient and flexible enough to handle unstructured and big data which in our project we will constantly deal with.
- ➔ Project Oxford for Bluemix
  - ◆ Since Dr. Su is offering a cognitive system course this semester which Allen is taking, and because Allen already did some projects with BlueMix, it's better for us to select BlueMix as our main tool.
- ➔ Mobile platform for web platform
  - ◆ None of the team members have experience in mobile development, so for the sake of quality, we will not attempt to do any mobile native development.

7.5. Analysis of Key Algorithms

➔ ask_question(input){
humanlizeString(input);
getInterests(input);
formatAIReadable(input);
updateUserInterest(input);
}

➔ formatAIReadable(input){
Foreach interest in input do:
interestWeight <- getConfidenceOfInterest(input);
input.append(interest * interestWeight)
Return input;
}

8. System Testing

8.1. Test Automation Framework

Steps for Installing Mocha

- ➔ Install the new package using the command **npm install --global mocha --save**
- ➔ Modify package.json's scripts to include **"test": "mocha"**
- ➔ Create test file directory with command **mkdir test**
- ➔ Edit test script with command **$EDITOR test/test.js**

Steps for Running Test Cases

- ➔ Type command **npm test**

8.2. Test Case Design

For a detailed description of each test case, see Appendix T Tables 8.2.1 - 8.2.25

Test Suites

Table 8.2.1: Test Suite TS-001: Unit Tests: Summary of all Unit Tests. Unit tests check for functionality of specific parts of the code. In these tests, we make sure that the expected result occurs under the specified conditions. For instance, if an invalid password were to be entered, we expect that a person would not log in. This would pass a test.

Unit Test Cases

Note: If you are not interested in details about the individual test cases, skip to the next section. The following contains in-depth information regarding each test case for clarity.

Table 8.2.2 Test Case TC-001: Testing if a user can properly log in. We expect that when a correct username and password is inputted, the user should be logged in.

Table 8.2.3 Test Case TC-002: Testing if a user can properly log out. We expect that when a user requests to log out, they should be logged out with no inputs needed.

Table 8.2.4 Test Case TC-003: Testing if a user can properly register. We expect that when a user inputs an untaken email and password they should be entered into the database properly.

Table 8.2.5 Test Case TC-004: Testing if a user can properly ask a question. We expect if a question is asked then the question should be logged to the database for records and a proper answer should be displayed.

Table 8.2.6 Test Case TC-005: Testing if a user can properly provide feedback. We expect when a user enters feedback, the feedback should be stored and a thank you message should be displayed to the user.

Table 8.2.7 Test Case TC-006: Testing if a user can properly translate text. We expect that if the user enters non-English text, the system should display the text in English.

Table 8.2.8 Test Case TC-007: Testing if the system can properly analyze a question. We expect that if a user asks a question that is ambiguous, the system should respond with another question.

Table 8.2.9 Test Case TC-008: Testing if a user can properly submit a profile. We expect that when new profile information is entered, the information should be stored to the database.

Table 8.2.10 Test Case TC-009: Testing if a user can properly generate an assessment. We expect that if descriptive text is given, the user requests to save their assessment, and they say yes to sharing with advisors then the assessment should be saved to the database with true visibility.

Table 8.2.11 Test Case TC-010: Testing if a user can properly respond to a request. We expect that if the user responds yes, the assessment that was requested should be visible by the person that requested it.

Table 8.2.12 Test Case TC-011: Testing if a user can properly view an assessment. We expect that the assessment should be displayed when the user requests it to be.

Table 8.2.13 Test Case TC-012: Testing if a user can properly train Watson. We expect that when a new question and answer pair is trained, the system should give the proper answer when the question is asked.

Table 8.2.14 Test Case TC-013: Testing if a user can properly log in. We expect that when a correct username but an incorrect password is inputted, the user should not be logged in.

Table 8.2.15 Test Case TC-014: Testing if a user can properly log in. We expect that when an incorrect username but a correct password is inputted, the user should not be logged in.

Table 8.2.16 Test Case TC-015: Testing if a user can properly log in. We expect that when both an incorrect username and an incorrect password are inputted, the user should not be logged in.

Table 8.2.17 Test Case TC-016: Testing if a user can properly respond to a request. We expect that if the user responds no, there are no changes to the assessment.

Table 8.2.18 Test Case TC-017: Testing if a user can properly register. We expect that when a user inputs an untaken email and an invalid password they should not be entered into the database and an error message is displayed.

Table 8.2.19 Test Case TC-018: Testing if a user can properly register. We expect that when a user inputs a taken email and a valid password they should not be entered into the database and an error message is displayed.

Table 8.2.20 Test Case TC-019: Testing if a user can properly register. We expect that when a user inputs a taken email and an invalid password they should not be entered into the database and an error message is displayed.

Table 8.2.21 Test Case TC-020: Testing if a user can properly generate an assessment. We expect that if descriptive text is given, the user requests to save their assessment, and they say no to sharing with advisors then the assessment should be saved to the database with false visibility.

Table 8.2.22 Test Case TC-021: Testing if a user can properly generate an assessment. We expect that if descriptive text is given and the user says no to saving the assessment then the assessment should not be saved to database.

Table 8.2.23 Test Case TC-022: Testing if a user can properly view the question log. We expect that when they ask to view the question log, the question log should be displayed.

Table 8.2.24 Test Case TC-023: Testing the speed of ask question. We expect that when a user asks a question then the answer should be displayed within 5 seconds.

Table 8.2.25 Test Case TC-024: Testing the management of a user's session. We expect that if the user is inactive for an hour or more, they will be logged out automatically by the system.

Integration Test Cases

Integration tests are being done manually.

System Test Cases

No test cases have been designed for the system at this time

Acceptance Test Cases

No test cases have been designed for user acceptance at this time.

8.3. Test Execution Report

To see a report of each test case being executed, see Appendix TE Tables 8.3.1 - 8.3.24

Unit Testing Report

All unit tests are passing or failing as expected, depending if a specific feature has been implemented at this time.

Integration Testing Report

Integration Tests are being done manually.

System Testing Report

No test cases have been designed for the system at this time

Acceptance Testing Report

No test cases have been designed for user acceptance at this time

9. Challenges & Open Issues

9.1. Challenges Faced in Requirements Engineering

- ➔ Initially, we struggled to determine the scope of the project. Watson has many features, so we had to be very specific about which of these features would make the most sense for our project's initial development.
- ➔ Understanding the domain, or what sort of questions we wanted Watson to answer.
- ➔ Be able to benchmark the accuracy of answering system.
- ➔ Be able to automate the training/learning system.
- ➔ Getting enough questions to be able to train Watson properly.
- ➔ Utilize AlchemyAPI to preprocess user input in order to increase system accuracy
- ➔ Optimize user feedback/system self-learning functionality

9.2. Challenges Faced in System Development

- ➔ Build and automate the backend system for training with retrieve and rank
- ➔ Build workable model to analyze questions and answer accuracy
- ➔ Try to utilize IBM Watson API to maximize the performance
- ➔ Security

9.3. Open Issues & Ideas for Solutions

- ➔ How to automate handle incorrect answer for system to learn the correct one?
- ➔ Use cognitive technology compare big data then use probability analysis

10. System Manuals

10.1. Instructions for System Development

How to setup development environment

The Intelligent Academic Planner project is a web application that will be accessible publicly on Heroku, a Node.JS environment host platform. The source code will be stored by GitHub, for version control purposes. And the server-implementation branch will always be automatically deployed and run on Heroku in development phase. There is no restriction on what IDE will be used by each group member. The ideal team tasks arrangement is 3 team members each work on frontend, backend, testing with domain expert working on training Watson.

Notes on system further extension

- ➔ Automate question and feedback log DB
- ➔ Natural language extension[9]
- ➔ Analytical library
- ➔ Visual recognition on building[7]
- ➔ Attitude analysis on question
- ➔ Campus direction utility
- ➔ Course info helper (location, material, etc.…)
- ➔ LionPath
- ➔ Schedule
- ➔ API
- ➔ News/feeds utility[17]

10.2. Instructions for System Deployment

Platform Requirements

NodeJS: main server platform, version require v6.0 and up
MongoDB: version requires 3.0.x and up
Modern Web Browser: Firefox, Chrome, Edge, Safari, Opera, Iceweasel
BlueMix
Heroku-GitHub

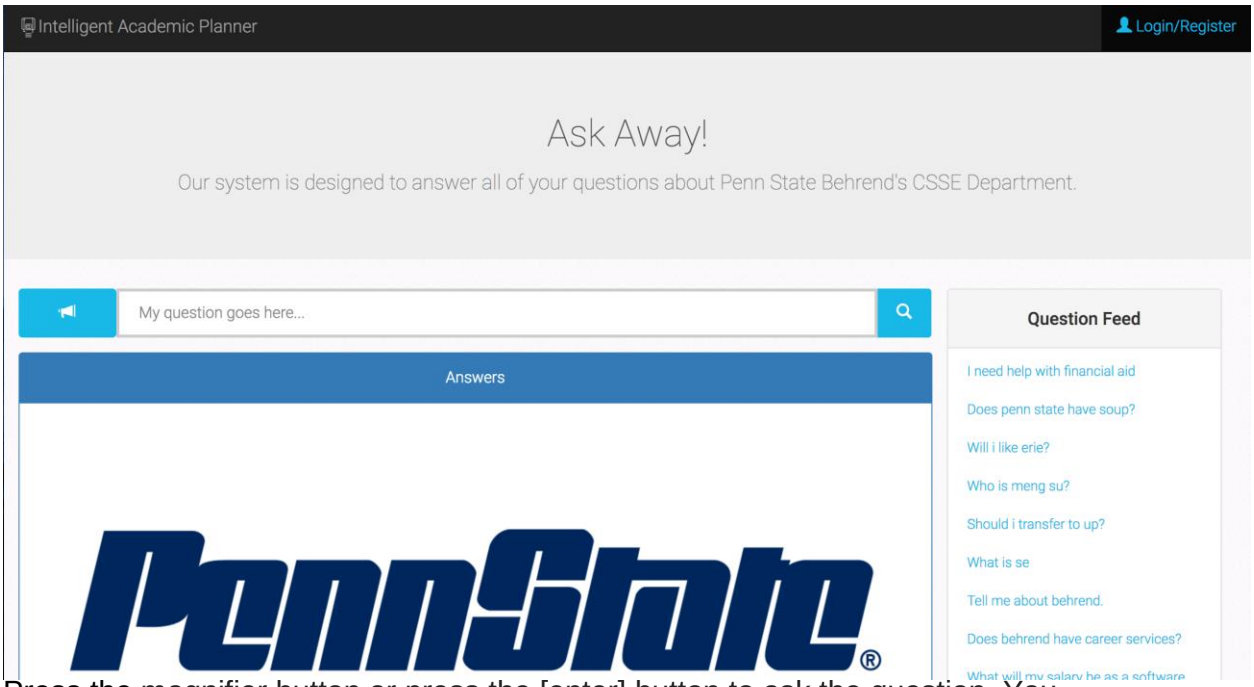
System Installation

NodeJS: local install by installer or any online NodeJS IDE
MongoDB: no installation required
BlueMix: no installation required
Client: user defined browser installation

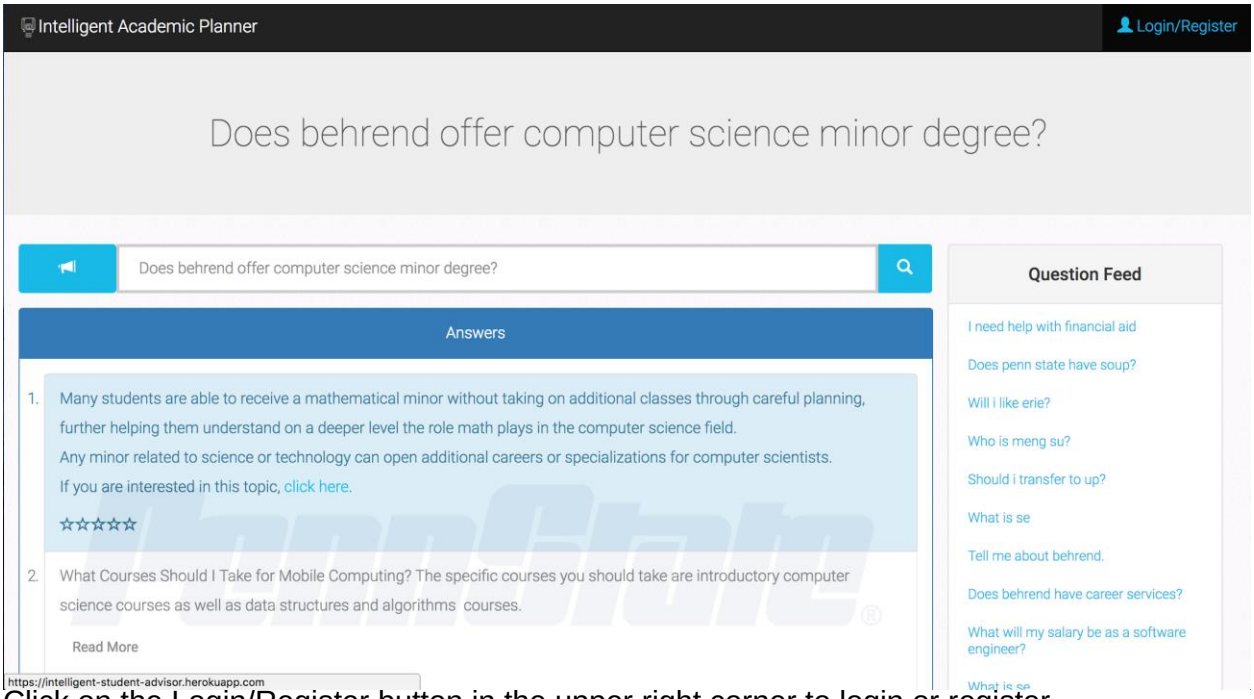
10.3. Instructions for System End Users

Navigate to the intelligent-student-advisor.herokuapp.com.

Ask a question on the index page by typing the question on the search bar in the middle.



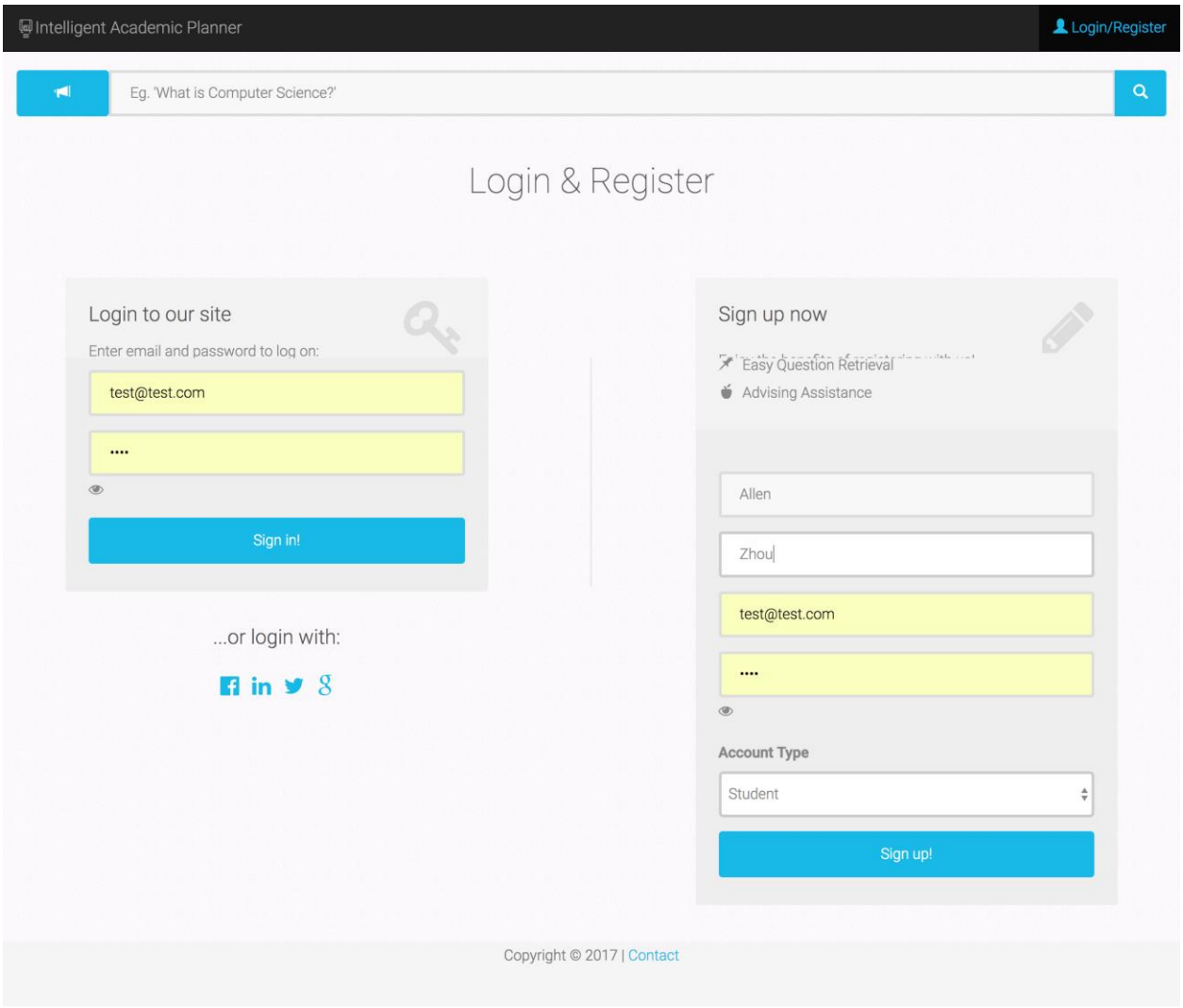
Press the magnifier button or press the [enter] button to ask the question. You should see the answers show up in the answer section as a group of 6. The top answer is on the top with blue background color.



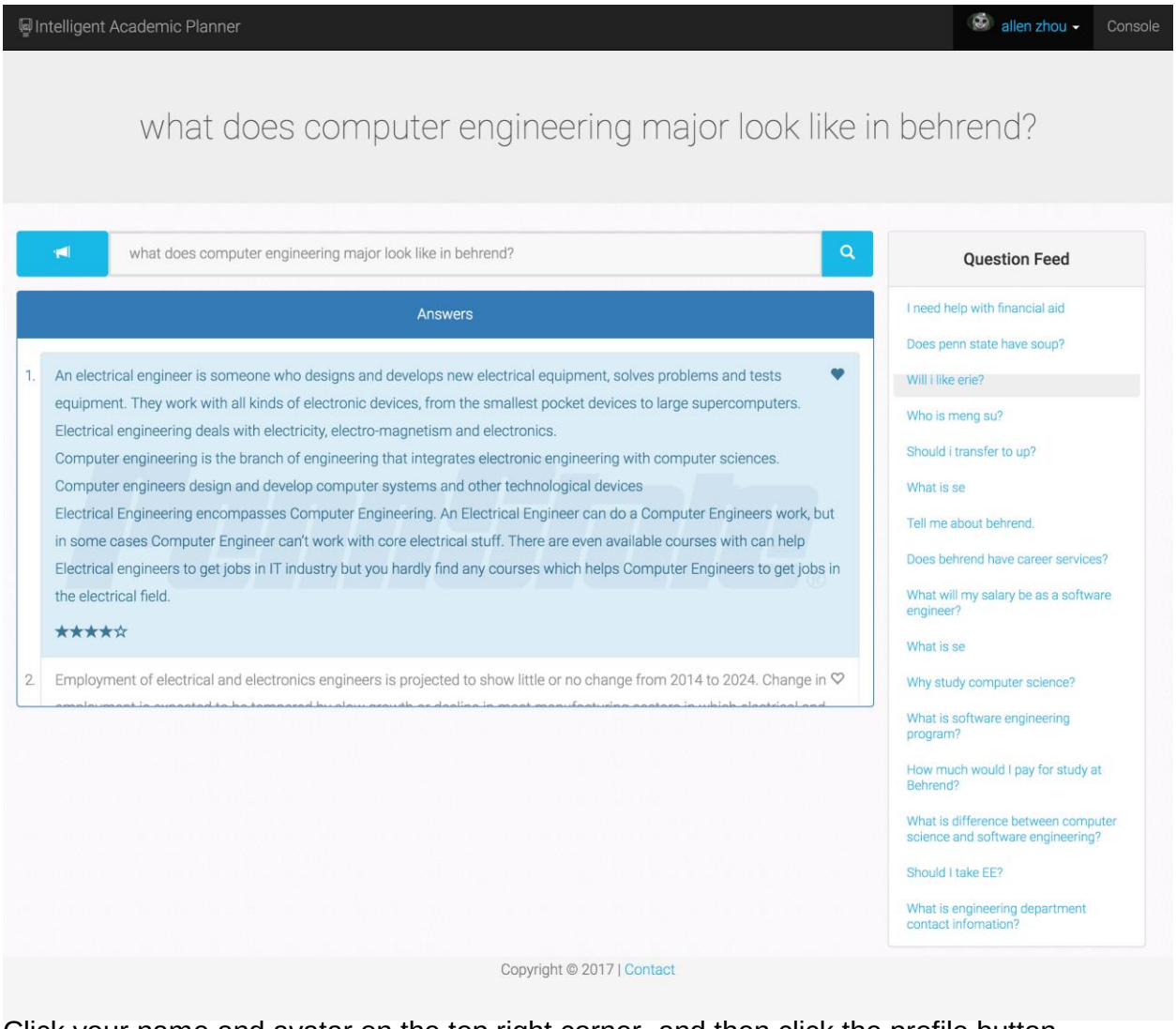
Click on the Login/Register button in the upper right corner to login or register.

Fill in email and password to login.

Or, fill first name, last name, email, password and account role to register.



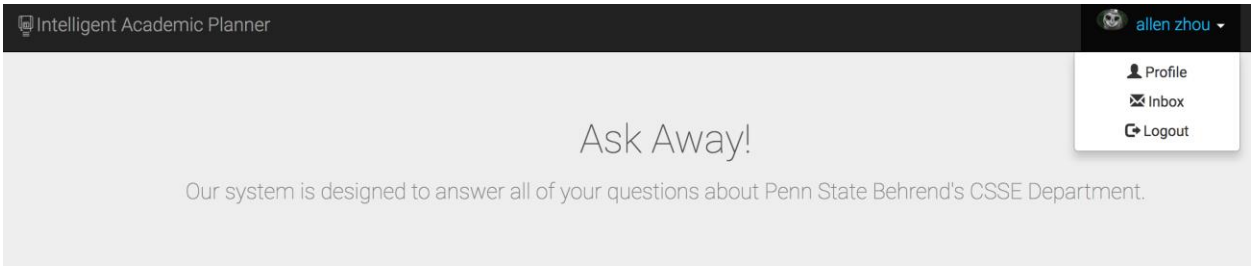
After login, you can favorite and rate answers based on your satisfaction. By favoriting an answer, the system remembers the answer with the question pair for you; you can go to your profile page to view it again in the future. By rating an answer, the system will learn from your rating and improve itself.



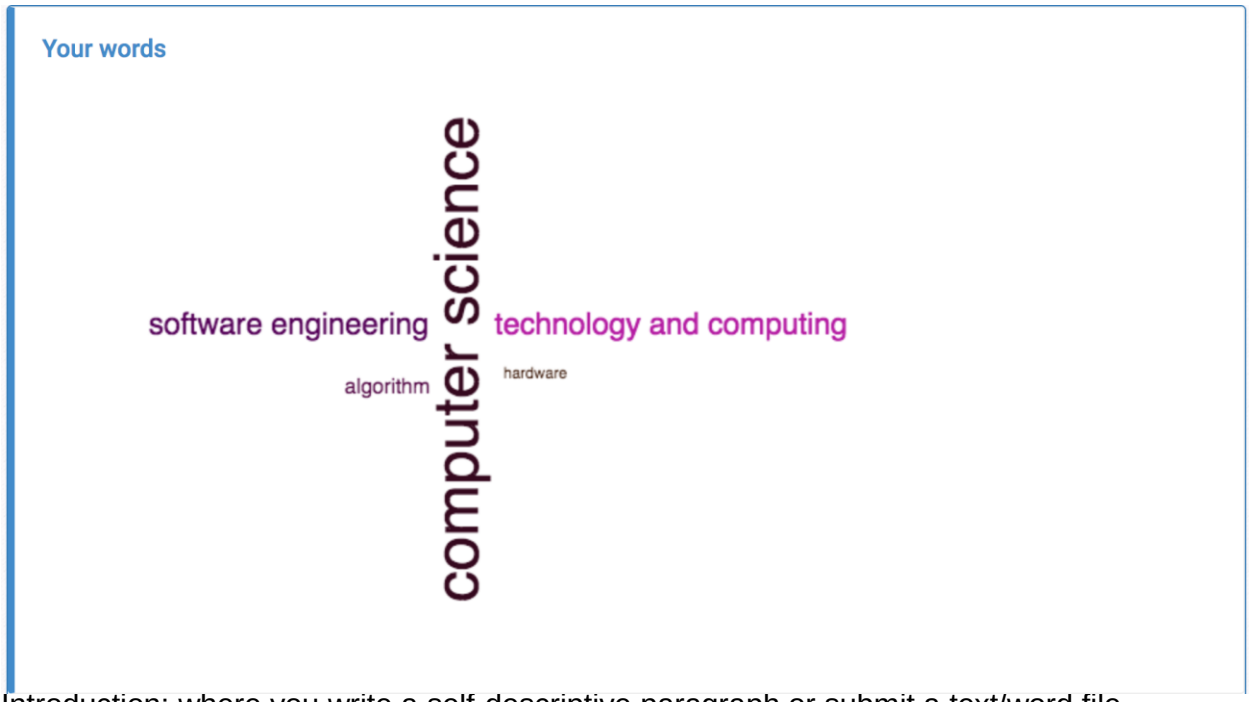
Click your name and avatar on the top right corner, and then click the profile button to view your personal information.

Click the inbox button to view your personal message from advisor(s).

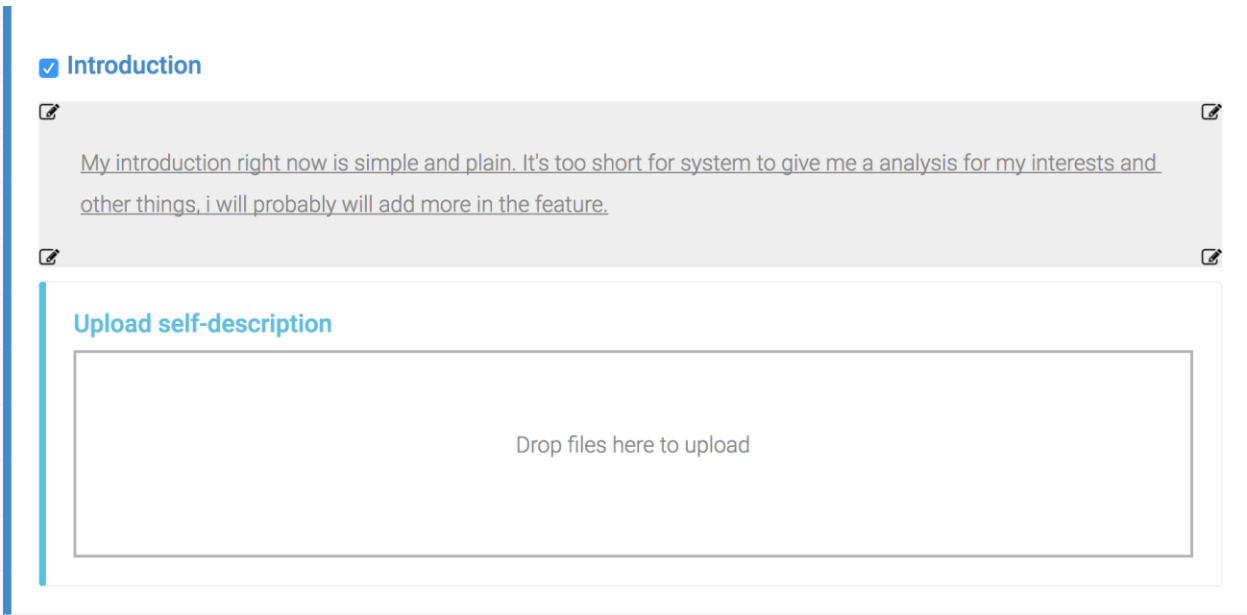
Click the logout button to end the login session and become a visitor again.



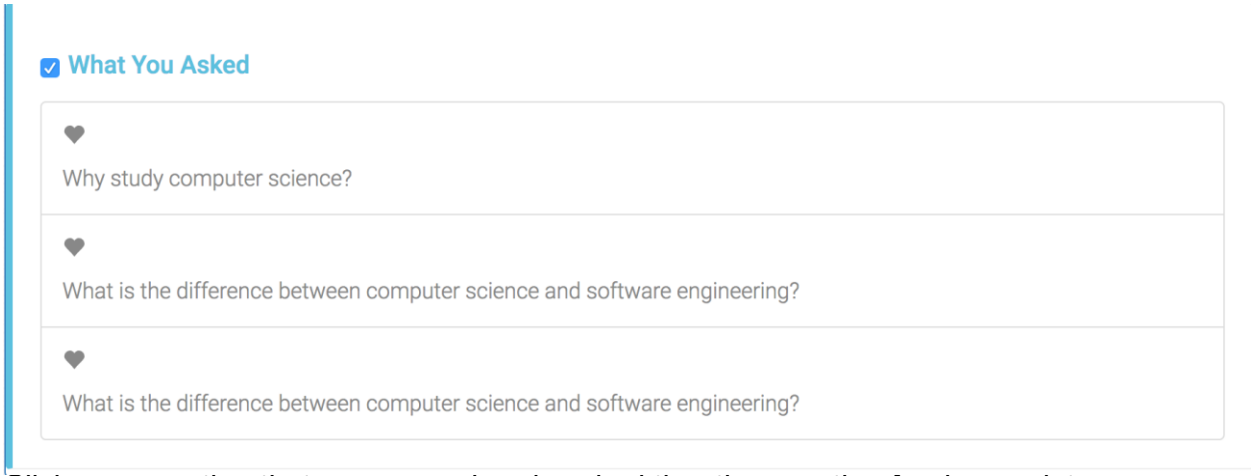
Once you are on the profile page, you can view following section about yourself: InterestCloud, a key term map indicates your interests from your introduction and questions.



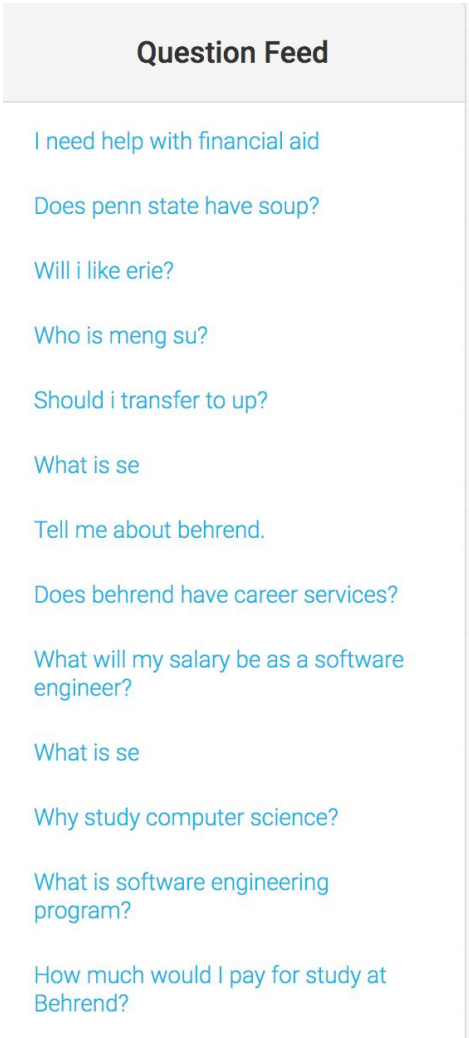
Introduction: where you write a self-descriptive paragraph or submit a text/word file about you, if you want to get an analysis about your introduction, you need to input 100 words minimum.



Question history: where shows your favorite question and answer pairs.

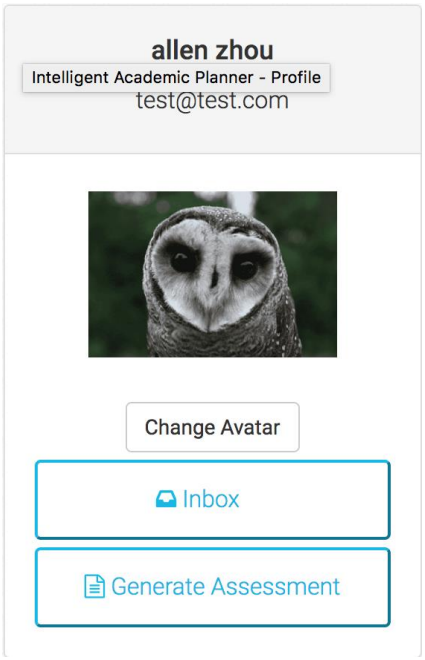


Click any question that someone already asked thru the question feeds panel, to know the answer of that specific question.



Click the change avatar button to change the avatar on the profile page.

Click the generate assessment to get an assessment of available information from you and send to an advisor.



11. Conclusion

This section will be gone over in future reports.

12. References

[1]. I. B. M. What is IBM Watson? http://www.ibm.com/watson/what-is-watson.html (accessed Oct 5, 2016).
[2]. MonkeyLearn http://docs.monkeylearn.com/ (accessed Oct 5, 2016).
[3]. AlchemyAPI http://www.alchemyapi.com/ (accessed Oct 5, 2016).
[4]. Watson Developer Cloud https://www.ibm.com/watson/developercloud/retrieve-rank.html (accessed Oct 5, 2016).
[5]. Watson Developer Cloud https://www.ibm.com/watson/developercloud/conversation.html (accessed Oct 5, 2016).
[6]. Watson Developer Cloud https://www.ibm.com/watson/developercloud/speech-to-text.html (accessed Oct 5, 2016).
[7]. Watson Developer Cloud https://www.ibm.com/watson/developercloud/visual-recognition.html (accessed Oct 5, 2016).
[8]. 3 Types of Survey Research, When to Use Them, and How they Can Benefit Your Organization! https://fluidsurveys.com/university/3-types-survey-research-use-can-benefit-organization/ (accessed Oct 21, 2016).
[9]. Machine Learning Methods in Natural Language Processing http://www.cs.columbia.edu/~mcollins/papers/tutorial_colt.pdf (accessed Oct 21, 2016).
[10]. IBM Knowledge Center https://www.ibm.com/support/knowledgecenter/SSSR99/kc/watsoncurator.html (accessed Oct 21, 2016).
[11]. Essentials of Machine Learning Algorithms (with Python and R Codes) https://www.analyticsvidhya.com/blog/2015/08/common-machine-learning-algorithms/ (accessed Oct 21, 2016).
[12]. NodeJS https://nodejs.org/en/ (accessed Oct 21, 2016).
[13]. Github https://help.github.com/ (accessed Oct 21, 2016).
[14]. MongoDB https://docs.mongodb.com/ (accessed Oct 21, 2016).
[15]. Language Translator-Translate and publish content in multiple languages. https://www.ibm.com/watson/developercloud/language-translator.html (accessed Oct 21, 2016).
[16]. Personality Insights-Uncover a deeper understanding of people's personality characteristics, needs, and values to drive personalization. http://www.ibm.com/watson/developercloud/personality-insights.html (accessed Oct 21, 2016).
[17]. Comparing Closed-Ended and Open-Ended Questions http://fluidsurveys.com/university/comparing-closed-ended-and-open-ended-questions/ (accessed Oct 21, 2016).
[18]. Pennsylvania State University - Erie http://www.ratemyprofessors.com/campusRatings.jsp?sid=1291 (accessed November 12, 2016).
[19]. Pennsylvania State University -- Erie, The Behrend College http://colleges.usnews.rankingsandreviews.com/best-colleges/penn-state-erie-3333/academics (accessed November 12, 2016).
[20]. The Pennsylvania State University http://engineering-schools.startclass.com/l/223/The-Pennsylvania-State-University (accessed November 12, 2016).

[21]. Penn State Erie - The Behrend College https://colleges.niche.com/penn-state-erie----the-behrend-college/ (accessed November 12, 2016).
[22]. Frequently Asked Questions - Honors Program https://psbehrend.psu.edu/Academics/academic-programs/honors/frequently-asked-questions-1 (accessed November 12, 2016).
[23]. What does a Computer Software Engineer do? Could you give me a description of the field? http://tryengineering.org/ask-expert/what-does-computer-software-engineer-do-could-you-give-me-description-field (accessed November 11, 2016).
[24]. Undergraduate Majors and Minors https://psbehrend.psu.edu/Academics/academic-programs/majors-minors (accessed November 11,2016).
[25]. Computer Engineering, B.S. Curriculum https://psbehrend.psu.edu/school-of-engineering/academic-programs/computer-engineering/curriculum (accessed November 11, 2016).
[26]. Computer Engineering http://psbehrend.psu.edu/school-of-engineering/academic-programs/computer-engineering (accessed November 11, 2016).
[27]. Penn State Erie, The Behrend College https://en.wikipedia.org/wiki/Penn_State_Erie,_The_Behrend_College (accessed November 11, 2016).
[28]. Computer Science & Engineering http://www.cs.washington.edu/prospective_students/undergrad/faq (accessed November 10, 2016).
[29]. Science and Engineering Indicators 2012 https://www.nsf.gov/statistics/seind12/c2/c2s2.htm (accessed November 10, 2016).
[30]. Annual Security Reports http://police.psu.edu/annual-security-reports (accessed November 10, 2016).
[31]. Adult Learner Support Services https://psbehrend.psu.edu/Academics/academic-services/adult/support-services (accessed November 10, 2016).
[32]. Electrical & Computer Engineering http://www.ee.uh.edu/undergraduate/computer-engineering-faq (accessed November 10, 2016).
[33]. Culture and engineering in the USA and Japan http://link.springer.com/article/10.1007/s00146-003-0280-z (accessed November 9, 2016).
[34]. Traveling opportunities as a software engineer? http://www.flyertalk.com/forum/travelbuzz/920489-traveling-opportunities-software-engineer.html (accessed November 9, 2016).
[35]. A job in computers that lets me travel a lot? http://ask.metafilter.com/88578/A-job-in-computers-that-lets-me-travel-a-lot (accessed November 9, 2016).
[36]. Traveling computer jobs http://www.indeed.com/q-Traveling-Computer-jobs.html (accessed November 9, 2016).
[37]. Average weather for Erie, Pennsylvania, USA https://weatherspark.com/averages/30194/Erie-Pennsylvania-United-States (accessed November 8, 2016).
[38]. Campus Dining http://behrendcampusliving.psu.edu/campus-dining (accessed November 8, 2016).
[39]. Tuition Calculator http://tuition.psu.edu/costestimate.aspx (accessed November 8, 2016).

[40]. Find 2016 Engineering Internships http://www.internships.com/engineering (accessed November 8, 2016).
[41]. What is the percentage of junior year computer science students in the top 30 US universities that are likely to be accepted to internships in 2016? https://www.quora.com/What-is-the-percentage-of-junior-year-computer-science-students-in-the-top-30-US-universities-that-are-likely-to-be-accepted-to-internships-in-2016 (accessed November 8, 2016).
[42]. Software Engineering / Computer Science difference https://users.csc.calpoly.edu/~djanzen/secsdiff.html (accessed November 7, 2016).
[43]. Pennsylvania State University Erie Behrend College http://www.studentsreview.com/PA/PSUE_comments.html (accessed November 10, 2016).
[44]. Academic Calendars http://registrar.psu.edu/academic_calendar/calendar_index.cfm (accessed November 7, 2016).
[45]. Top 10 Benefits of Becoming an Engineer http://old.ece.ufl.edu/admission/undergraduate/benefits.htm (accessed November 10, 2016).
[46]. How Much Does a Computer Engineer get Paid Per Hour? http://work.chron.com/much-computer-engineer-paid-per-hour-21954.html (accessed November 8, 2016).
[47]. Tau Beta Pi https://www.tbp.org/home.cfm (accessed November 8, 2016).
[48]. Cost of Living in Erie https://www.numbeo.com/cost-of-living/in/Erie (accessed November 10, 2016).
[49]. Computer Hardware Engineer Career https://www.mymajors.com/career/computer-engineers/education/ (accessed November 12, 2016).
[50]. What do Engineers have in common? http://engineeringrevision.com/367/what-do-engineers-have-in-common/ (accessed November 12, 2016).
[51]. Hobbies that engineers have? https://www.reddit.com/r/engineering/comments/2ed8sj/hobbies_that_engineers_have/ (accessed November 12, 2016).
[52]. Computer Engineering, Software Engineering, or Computer Science? https://engiegirlsatuwaterloo.wordpress.com/2013/08/29/computer-engineering-software-engineering-or-computer-science/ (accessed November 12, 2016).

Table 4.1. Use Case Index Table

Project Name: Intelligent Academic Planner				
Use Case ID	**Use Case Name**	**Level**	**Author**	**Version**
UC-001	Respond to Request	Primary task	Daria Cook	0.2
UC-004	Register	Primary task	Daria Cook	0.7
UC-005	View assessment	Subfunction	Xiaoyu Zhou	0.8
UC-006	View question log	Primary task	Xiaoyu Zhou	0.3
UC-007	Train Watson	Primary task	Xiaoyu Zhou	0.6
UC-009	Real-time translation	Subfunction	Xiaoyu Zhou	0.6
UC-010	Answer quality feedback	Primary task	Xiaoyu Zhou	0.7
UC-011	Ask question	Primary task	Xiaoyu Zhou	0.6
UC-012	Analyze Question	Primary task	Xiaoyu Zhou	0.6
UC-013	Submit Profile	Primary task	Xiaoyu Zhou	0.3
UC-015	Generate Assessment	Primary task	Xiaoyu Zhou	0.3
UC-016	Login	Primary task	Xiaoyu Zhou	0.5
UC-018	Logout	Primary task	Xiaoyu Zhou	0.3
UC-019	Request Assessment	Primary task	Xiaoyu Zhou	0.4
Acknowledgment: Generated from the CapStone process management system ©2015				

Table 4.2. Use Case UC-001

Project Name:	
Use Case ID:	UC-001
Use Case Name:	Respond to Request
User Goal:	User approves of advisor viewing assessment
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-E
Relevant System Reqs:	SF-E-05
Primary Actor:	Student
Precondition:	User has a request awaiting approval
Minimal Guarantee:	User's assessment are non-view-able
Success Guarantee:	User's assessment becomes view-able for advisor that requested
Trigger:	User requests to respond to request
Success Scenario:	**Step** **Actions** 1 The user requests to respond to request 2 The system asks for user's response 3 The user responds
Extensions:	Branching Scenarios
3A	**Condition:** If request is declined
	Step **Actions** 1 The system sends notification to advisor 2 Exit out of functionality
3B	**Condition:** If request is accepted
	Step **Actions** 1 The system sends notification to advisor 2 The system allows advisor to view assessment of student
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.3. Use Case UC-004

Project Name:	
Use Case ID:	UC-004
Use Case Name:	Register
User Goal:	To be recognized by the system.
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-F
Relevant System Reqs:	SF-F-01
Primary Actor:	Visitor, System Devleoper
Precondition:	User is viewing program
Minimal Guarantee:	User is not recognized by system
Success Guarantee:	User is recognized by system
Trigger:	User requests to register
Success Scenario:	**Step** **Actions** 1 The user requests to register 2 The system asks for registration information 3 The user inputs registration information 4 The system validates information 5 The system accepts user registration
Extensions:	Branching Scenarios
4A	**Condition:** Information is invalid
	Step **Actions** 1 The system notifies user of invalid information 2 Return to step 2
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.4. Use Case UC-005

Project Name:	
Use Case ID:	UC-005
Use Case Name:	View assessment
User Goal:	User is able to view assessment
Scope:	IAP System
Level:	Subfunction
Relevant User Reqs:	UF-E
Relevant System Reqs:	SF-E-03
Primary Actor:	student, advisor
Precondition:	The user is logged in
Minimal Guarantee:	System does not display student's assessment
Success Guarantee:	system display student's assessment
Trigger:	User requests to view assessments
Success Scenario:	Step / Actions 1 The user requests to view assessments 2 The system displays assessments that can be viewed
Extensions:	Branching Scenarios
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.5. Use Case UC-006

Project Name:	
Use Case ID:	UC-006
Use Case Name:	View question log
User Goal:	User is able to view the question log
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-G
Relevant System Reqs:	SF-G-01
Primary Actor:	Adviser, System Developer
Precondition:	User is logged in
Minimal Guarantee:	System does not display question log
Success Guarantee:	System displays question log
Trigger:	User requests to view question log
Success Scenario:	Step / Actions 1 The user requests to view question log 2 The system asks for filter information 3 The user enters filter information 4 The system displays all questions based on filter information
Extensions:	Branching Scenarios
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.6. Use Case UC-007

Project Name:	
Use Case ID:	UC-007
Use Case Name:	Train Watson
User Goal:	User be able to train watson with new questions that user asked but hasn't been answered
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-H
Relevant System Reqs:	SF-A-01
Primary Actor:	System Developer
Precondition:	User is logged in
Minimal Guarantee:	Watson is not further trained
Success Guarantee:	Watson is further trained
Trigger:	User requests to train Watson
Success Scenario:	**Step** **Actions** 1 The user request system export new questions 2 The system export new question list as a text file 3 The user request system export incorrect answered questions 4 The system export downvotes answered questions 5 The user request view answer quality 6 The system display the statistical analysis of the answer for each of the questions 7 The user request view the feedback of answer from user 8 The system display the most up voted help feedback for the answer 9 The user request the analysis of a specific question 10 The system display analysis of a question by keyterm
Extensions:	Branching Scenarios
1A	**Condition:** User request system export all new questions from last time export
	Step **Actions** 1 The system export question list that is new from last time
1B	**Condition:** User request system export all new questions by major
	Step **Actions**
1C	**Condition:** User request system export all new questions other condition
	Step **Actions** 1 The system export top 10 unanswered FAQ 2 The system export top 10% unanswered FAQ
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.7. Use Case UC-009

Project Name:	
Use Case ID:	UC-009
Use Case Name:	Real-time translation
User Goal:	User is able to see input/output in english
Scope:	IAP System
Level:	Subfunction
Relevant User Reqs:	UF-C
Relevant System Reqs:	SF-C-02
Primary Actor:	Visitor, Student, Advisor
Precondition:	User is asking a question
Minimal Guarantee:	Input/output is not translated
Success Guarantee:	System displays translated text
Trigger:	User requests translation
Success Scenario:	Step / Actions 1 The user input non-english input 2 The system determines what language is being used 3 The system translates text to english 4 The system displays english answer
Extensions:	Branching Scenarios
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.8. Use Case UC-010

Project Name:	
Use Case ID:	UC-010
Use Case Name:	Answer quality feedback
User Goal:	User is able to provide feedback on question response.
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-H
Relevant System Reqs:	SF-H-01
Primary Actor:	Visitor, Student, Advisor
Precondition:	User is asking a question
Minimal Guarantee:	Feedback is not stored
Success Guarantee:	Feedback is stored
Trigger:	User requests to submit feedback
Success Scenario:	Step / Actions 1 The user requests to submit feedback 2 The system allows user to enter feedback 3 The user submits feedback 4 The system stores feedback
Extensions:	Branching Scenarios
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.9. Use Case UC-011

Project Name:	
Use Case ID:	UC-011
Use Case Name:	Ask question
User Goal:	User is able to ask questions
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-C
Relevant System Reqs:	SF-C-01,SF-C-02,SF-C-03,SF-C-04,SF-C-05
Primary Actor:	Visitor, Student, Advisor
Precondition:	User is viewing program
Minimal Guarantee:	Question is not logged
Success Guarantee:	Question is logged
Trigger:	The user requests to ask a question
Success Scenario:	**Step** **Actions** 1 The user requests to ask a question 2 The system requests user's question 3 The user enters question and requests answer 4 The system analyzes question <<Analyze Question>> 5 The system displays answer 6 The system logs question to question log 7 The system asks user if they would like to submit feedback on answer quality
Extensions:	Branching Scenarios
3A	**Condition:** If user requests translation
	Step **Actions** 1 The system translates text <<Realtime Translation>>
7A	**Condition:** If user requests to provide feedback
	Step **Actions** 1 The system requests feedback information <<Answer Quality Feedback>>
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.10. Use Case UC-012

Project Name:	
Use Case ID:	UC-012
Use Case Name:	Analyze Question
User Goal:	System determines answer for user
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-C,UF-D
Relevant System Reqs:	SF-C-01,SF-C-03,SF-C-04,SF-C-05,SF-D-01
Primary Actor:	N/A
Precondition:	User asks a question
Minimal Guarantee:	Question is not analyzed
Success Guarantee:	Question is analyzed
Trigger:	User submits a question
Success Scenario:	**Step** **Actions** 1 The user submits a question 2 The system performs textual analysis on the question 3 The system determines if another question could be asked to clarify answer 4 The system displays answer
Extensions:	Branching Scenarios
1A	**Condition:** If another question could be asked
	Step **Actions** 1 The system asks user the question 2 The user responds to question 3 Return to step 2 in main scenario using response given
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.11. Use Case UC-013

Project Name:	
Use Case ID:	UC-013
Use Case Name:	Submit Profile
User Goal:	User can create profile
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-E
Relevant System Reqs:	SF-E-01,SF-E-02,SF-E-03,SF-E-04,SF-E-05
Primary Actor:	student
Precondition:	User is logged in
Minimal Guarantee:	Profile is not stored
Success Guarantee:	Profile is stored
Trigger:	User requests to update profile
Success Scenario:	**Step** **Actions** 1 The user requests to update profile 2 The system verify the profile 3 The system upload the profile to server
Extensions:	Branching Scenarios
2A	**Condition:** If system detect invalid format of the profile
	Step **Actions** 1 The system notify user of problem 2 Return to step 1
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.12. Use Case UC-015

Project Name:	
Use Case ID:	UC-015
Use Case Name:	Generate Assessment
User Goal:	User receives assessment of profile information
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-E
Relevant System Reqs:	SF-E-04,SF-E-05
Primary Actor:	Student
Precondition:	User has submitted information to profile
Minimal Guarantee:	No assessment generated
Success Guarantee:	System generate assessment and display to user
Trigger:	User requests to generate assessment
Success Scenario:	**Step** **Actions** 1 The user requests to generate assessment 2 The system analyzes profile and generates assessment 3 The system displays assessment
Extensions:	Branching Scenarios
3A	**Condition:** keep assessment on user's record
	Step **Actions** 1 The user request system save the assessment 2 The system upload assessment to server and relate to user account
3B	**Condition:** The user request assessment to be viewable by advisers
	Step **Actions** 1 The user request assessment to be viewable by advisers 2 The system mark user's assessment as public and save to user's account
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.13. Use Case UC-016

Project Name:	
Use Case ID:	UC-016
Use Case Name:	Login
User Goal:	User is able to log in
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-A
Relevant System Reqs:	SF-A-01
Primary Actor:	Student, Advisor, System Devel
Precondition:	User is registered
Minimal Guarantee:	User is not logged in
Success Guarantee:	User is logged in
Trigger:	User requests to log in
Success Scenario:	**Step** **Actions** 1 The user request to login to the system 2 The system verifies user's login credential 3 The system logs in user
Extensions:	Branching Scenarios
2A	**Condition:** login credential doesn't match account info
	Step **Actions** 1 The system notifies user of problem 2 Return to step 1
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.14. Use Case UC-018

Project Name:	
Use Case ID:	UC-018
Use Case Name:	Logout
User Goal:	User is able to log out
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-B
Relevant System Reqs:	SF-B-01
Primary Actor:	Student, Advisor, System Devel
Precondition:	User is logged in
Minimal Guarantee:	User is not logged out
Success Guarantee:	User is logged out
Trigger:	User requests to log out
Success Scenario:	**Step** **Actions** 1 The user requests to logout 2 The system verifies all information is saved 3 The system logs users out
Extensions:	Branching Scenarios
2A	**Condition:** If some information is unsaved
	Step **Actions** 1 The system checks if user still wants to log out 2 The user responds 3 BRANCH - If user responds no exit functionality. Else - Continue on.
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.15. Use Case UC-019

Project Name:	
Use Case ID:	UC-019
Use Case Name:	Request Assessment
User Goal:	System notify student that an adviser wants to see his assessment
Scope:	IAP System
Level:	Primary task
Relevant User Reqs:	UF-E
Relevant System Reqs:	SF-E-05
Primary Actor:	Advisor
Precondition:	User is logged in
Minimal Guarantee:	No request is sent
Success Guarantee:	Request is sent
Trigger:	User requests to send request to student
Success Scenario:	**Step** **Actions** 1 The user requests to send request to student 2 The system asks for student information 3 The user inputs student information 4 The system validates student information 5 The system sends request to student
Extensions:	Branching Scenarios
4A	**Condition:** If invalid student information
	Step **Actions** 1 The system notifies user of problem 2 Return to step 2
Acknowledgment: Generated from the CapStone process management system ©2015	

Table 4.16. User Functional Requirements: UF-A

Project Name:	Intelligent Academic Planner				
Requirement #:	UF-A			Type: Functional / Non-Functional	
Creation:	Sep 20 2016 01:25 PM			User: ☒ / ☐	
Modification:	Oct 05 2016 01:50 AM			System: ☐ / ☐	
Description:	A user can log in.				
Priority:	Highest	✔ High	Medium	Low	Lowest
This Req. is Refined Into:	SF-A-01				
Justify why UF-A can be completely covered by SF-A-01	If a user can login within 5 seconds, they can log in properly.				
Traceability:	Use cases cf.	UC-016			
	Test cases cf.	TC-001, TC-013, TC-014, TC-015			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.17. User Functional Requirements: UF-B

Project Name:	Intelligent Academic Planner				
Requirement #:	UF-B			Type: Functional / Non-Functional	
Creation:	Sep 20 2016 04:17 PM			User: ☒ / ☐	
Modification:	Oct 05 2016 01:50 AM			System: ☐ / ☐	
Description:	A user can log out.				
Priority:	Highest	✔ High	Medium	Low	Lowest
This Req. is Refined Into:	SF-B-01				
Justify why UF-B can be completely covered by SF-B-01	If a user can log out within 5 seconds, they can log out properly				
Traceability:	Use cases cf.	UC-018			
	Test cases cf.	TC-002			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.18. User Functional Requirements: UF-C

Project Name:	Intelligent Academic Planner				
Requirement #:	UF-C			Type: Functional / Non-Functional	
Creation:	Sep 20 2016 04:18 PM			User: ☒ / ☐	
Modification:	Sep 20 2016 04:58 PM			System: ☐ / ☐	
Description:	A user can ask the system questions.				
Priority:	✔ Highest	High	Medium	Low	Lowest
This Req. is Refined Into:	SF-C-01, SF-C-02, SF-C-03, SF-C-04, SF-C-05				
Justify why UF-C can be completely covered by SF-C-01, SF-C-02, SF-C-03, SF-C-04, SF-C-05	If the system is able to perform the functionality in SF-C-01 through SF-C-06 then a user will undoubtedly have been able to ask the system questions. The system would not be able to perform these functions without a user first asking it a question.				
Traceability:	Use cases cf.	UC-009, UC-011, UC-012			
	Test cases cf.	TC-004, TC-006, TC-007			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.19. User Functional Requirements: UF-D

Project Name:	Intelligent Academic Planner				
Requirement #:	UF-D			Type: Functional / Non-Functional	
Creation:	Sep 23 2016 12:52 PM			User: ☒ / ☐	
Modification:	Oct 05 2016 01:51 AM			System: ☐ / ☐	
Description:	A user should receive multiple responses to a question.				
Priority:	Highest	High	Medium	✔ Low	Lowest
This Req. is Refined Into:	SF-D-01				
Justify why UF-D can be completely covered by SF-D-01	By requiring a minimum of 1 response to be given, it is a given that multiple responses are given to a question.				
Traceability:	Use cases cf.	UC-012			
	Test cases cf.	TC-007			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.20. User Functional Requirements: UF-E

Project Name:	Intelligent Academic Planner				
Requirement #:	UF-E			Type: Functional / Non-Functional	
Creation:	Oct 04 2016 11:51 PM			User: ☒ / ☐	
Modification:	Oct 05 2016 12:06 AM			System: ☐ / ☐	
Description:	A user can create a profile				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Refined Into:	SF-E-01, SF-E-02, SF-E-03, SF-E-04, SF-E-05				
Justify why UF-E can be completely covered by SF-E-01, SF-E-02, SF-E-03, SF-E-04, SF-E-05	All of the system requirements associated with this cover what can be put into their profile.				
Traceability:	Use cases cf.	UC-001, UC-005, UC-013, UC-015, UC-019			
	Test cases cf.	TC-008, TC-009, TC-010, TC-011, TC-016, TC-020, TC-021			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.21. User Functional Requirements: UF-F

Project Name:	Intelligent Academic Planner				
Requirement #:	UF-F			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 12:15 AM			User: ☒ / ☐	
Modification:	Oct 05 2016 01:51 AM			System: ☐ / ☐	
Description:	A user can register				
Priority:	Highest	✔ High	Medium	Low	Lowest
This Req. is Refined Into:	SF-F-01				
Justify why UF-F can be completely covered by SF-F-01	If a user can register within 5 seconds, they can register properly.				
Traceability:	Use cases cf.	UC-004			
	Test cases cf.	TC-003, TC-017, TC-018, TC-019			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.22. User Functional Requirements: UF-G

Project Name:	Intelligent Academic Planner				
Requirement #:	UF-G			Type: User ☒ System ☐	Functional: ☐ ☐ Non-Functional: ☐ ☐
Creation:	Oct 05 2016 12:36 AM				
Modification:	Oct 05 2016 01:52 AM				
Description:	A user can view a log of asked questions.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Refined Into:	SF-G-01				
Justify why UF-G can be completely covered by SF-G-01	If advisors and system developers can view the question log, then users can view the question log.				
Traceability:	Use cases cf.	UC-006			
	Test cases cf.	TC-022			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.23. User Functional Requirements: UF-H

Project Name:	Intelligent Academic Planner				
Requirement #:	UF-H			Type: User ☒ System ☐	Functional: ☐ ☐ Non-Functional: ☐ ☐
Creation:	Oct 05 2016 02:12 AM				
Modification:	Oct 05 2016 02:16 AM				
Description:	A user can provide information to improve accuracy of the system.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Refined Into:	SF-H-01				
Justify why UF-H can be completely covered by SF-H-01	If answer quality feedback is submitted, accuracy of responses can be increased.				
Traceability:	Use cases cf.	UC-007, UC-010			
	Test cases cf.	TC-005, TC-012			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.24. User NonFunctional Requirements: UP-03

Project Name:	Intelligent Academic Planner				
Requirement #:	UP-03			Type / Functional / Non-Functional	
Creation:	Oct 05 2016 01:36 AM			User ☐ ☒	
Modification:	Oct 05 2016 01:54 AM			System ☐ ☐	
Description:	A user should receive a quick response after asking a question			Product (sub-type below)	
				Performance Requirements	
Priority:	Highest	High	Medium	✔ Low	Lowest
This Req. is Refined Into:	SP-03-01				
Justify why UP-03 can be completely covered by SP-03-01	By specifying performance requirements, it is ensured that the question will be answered in a quick manner.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	TC-023			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.25. User NonFunctional Requirements: UP-01

Project Name:	Intelligent Academic Planner				
Requirement #:	UP-01			Type / Functional / Non-Functional	
Creation:	Oct 05 2016 12:10 AM			User ☐ ☒	
Modification:	Oct 05 2016 01:53 AM			System ☐ ☐	
Description:	A user's profile should be secure.			Product (sub-type below)	
				Dependability/Reliability/Security	
Priority:	Highest	✔ High	Medium	Low	Lowest
This Req. is Refined Into:	SP-01-01				
Justify why UP-01 can be completely covered by SP-01-01	Ensures only specific people can view a user's profile, making it secure.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	TC-009			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.26. User NonFunctional Requirements: UO-01

Project Name:	Intelligent Academic Planner				
Requirement #:	UO-01			Type: Functional / Non-Functional	
Creation:	Sep 23 2016 12:42 PM			User: ☐ / ☒	
Modification:	Oct 05 2016 01:53 AM			System: ☐ / ☐	
Description:	A user's session should be managed.			Organizational (sub-type below)	
				Development Requirements	
Priority:	Highest	✔ High	Medium	Low	Lowest
This Req. is Refined Into:	SO-01-01				
Justify why UO-01 can be completely covered by SO-01-01	Ensures that a user can only be logged in for 1 hour, managing their session.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	TC-024			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.27. System Functional Requirements: SF-A-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-A-01			Type: Functional / Non-Functional	
Creation:	Sep 23 2016 12:22 PM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:41 AM			System: ☒ / ☐	
Description:	The system should log-in a user within 5 seconds.				
Priority:	Highest	High	Medium	✔ Low	Lowest
This Req. is Engineered From:	UF-A				
Justify why meeting SF-A-01 can contribute to the fulfilment of UF-A	Provides performance requirement for logging in.				
Traceability:	Use cases cf.	UC-007, UC-016			
	Test cases cf.	TC-001, TC-013, TC-014, TC-015			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.28. System Functional Requirements: SF-B-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-B-01			Type: Functional / Non-Functional	
Creation:	Sep 23 2016 12:32 PM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:41 AM			System: ☒ / ☐	
Description:	The system should log-out a user within 5 seconds.				
Priority:	Highest	High	Medium	✔ Low	Lowest
This Req. is Engineered From:	UF-B				
Justify why meeting SF-B-01 can contribute to the fulfilment of UF-B	Explains performance requirement.				
Traceability:	Use cases cf.	UC-018			
	Test cases cf.	TC-002			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.29. System Functional Requirements: SF-C-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-C-01			Type: Functional / Non-Functional	
Creation:	Sep 20 2016 04:22 PM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:42 AM			System: ☒ / ☐	
Description:	The system should conduct textual analyses.				
Priority:	Highest	✔ High	Medium	Low	Lowest
This Req. is Engineered From:	UF-C				
Justify why meeting SF-C-01 can contribute to the fulfilment of UF-C	In order to provide an answer to questions asked, system must be able to perform this function.				
Traceability:	Use cases cf.	UC-011, UC-012			
	Test cases cf.	TC-004, TC-007			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.30. System Functional Requirements: SF-C-02

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-C-02			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 01:59 AM			User: ☐ / ☐	
Modification:	Oct 05 2016 02:00 AM			System: ☒ / ☐	
Description:	The system should be able to handle input from multiple well-known languages.				
Priority:	Highest	High	Medium	Low	✔ Lowest
This Req. is Engineered From:	UF-C				
Justify why meeting SF-C-02 can contribute to the fulfilment of UF-C	Allows users to ask a question in a variety of ways.				
Traceability:	Use cases cf.	UC-009, UC-011			
	Test cases cf.	TC-004, TC-006			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.31. System Functional Requirements: SF-C-03

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-C-03			Type: Functional / Non-Functional	
Creation:	Sep 20 2016 04:31 PM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:44 AM			System: ☒ / ☐	
Description:	The system should recommend majors suitable for the user based on the personality assessment.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-C				
Justify why meeting SF-C-03 can contribute to the fulfilment of UF-C	This allows questions to be answered more accurately.				
Traceability:	Use cases cf.	UC-011, UC-012			
	Test cases cf.	TC-004, TC-007			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.32. System Functional Requirements: SF-C-04

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-C-04			Type: Functional / Non-Functional	
Creation:	Sep 20 2016 04:34 PM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:44 AM			System: ☒ / ☐	
Description:	The system should gather data unique to each user.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-C				
Justify why meeting SF-C-04 can contribute to the fulfilment of UF-C	This allows questions to be answered more accurately.				
Traceability:	Use cases cf.	UC-011, UC-012			
	Test cases cf.	TC-004, TC-007			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.33. System Functional Requirements: SF-C-05

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-C-05			Type: Functional / Non-Functional	
Creation:	Sep 20 2016 04:35 PM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:44 AM			System: ☒ / ☐	
Description:	The system should recommend courses based on the recommended majors.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-C				
Justify why meeting SF-C-05 can contribute to the fulfilment of UF-C	This allows questions to be answered more accurately.				
Traceability:	Use cases cf.	UC-011, UC-012			
	Test cases cf.	TC-004, TC-007			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.34. System Functional Requirements: SF-D-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-D-01			Type: Functional / Non-Functional	
Creation:	Sep 23 2016 12:54 PM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:46 AM			System: ☒ / ☐	
Description:	The system should show a minimum of 1 related search/question.				
Priority:	Highest	High	Medium	✔ Low	Lowest
This Req. is Engineered From:	UF-D				
Justify why meeting SF-D-01 can contribute to the fulfilment of UF-D	This allows responses to be structured and more accurate.				
Traceability:	Use cases cf.	UC-012			
	Test cases cf.	TC-007			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.35. System Functional Requirements: SF-E-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-E-01			Type: Functional / Non-Functional	
Creation:	Oct 04 2016 11:56 PM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:46 AM			System: ☒ / ☐	
Description:	The system should allow between 100 and 600 words to describe a user's academic and professional interests.				
Priority:	✔ Highest	High	Medium	Low	Lowest
This Req. is Engineered From:	UF-E				
Justify why meeting SF-E-01 can contribute to the fulfilment of UF-E	Allows user to enter information about themselves to their profile				
Traceability:	Use cases cf.	UC-013			
	Test cases cf.	TC-008			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.36. System Functional Requirements: SF-E-02

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-E-02			Type: Functional / Non-Functional	
Creation:	Oct 04 2016 11:56 PM			User ☐ ☐	
Modification:	Oct 05 2016 01:47 AM			System ☒ ☐	
Description:	The system should allow a user to submit 100 words of self-description about their personality.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-E				
Justify why meeting SF-E-02 can contribute to the fulfilment of UF-E	Allows user to enter personality information on their profile				
Traceability:	Use cases cf.	UC-013			
	Test cases cf.	TC-008			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.37. System Functional Requirements: SF-E-03

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-E-03			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 12:00 AM			User ☐ ☐	
Modification:	Oct 05 2016 01:47 AM			System ☒ ☐	
Description:	The system should allow a user to view their personality assessments.				
Priority:	Highest	✔ High	Medium	Low	Lowest
This Req. is Engineered From:	UF-E				
Justify why meeting SF-E-03 can contribute to the fulfilment of UF-E	Allows a user to learn about themselves based on profile information.				
Traceability:	Use cases cf.	UC-005, UC-013			
	Test cases cf.	TC-011			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.38. System Functional Requirements: SF-E-04

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-E-04			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 01:43 AM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:48 AM			System: ☒ / ☐	
Description:	The system should create a personality assessment unique to each user based on the data gathered.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-E				
Justify why meeting SF-E-04 can contribute to the fulfilment of UF-E	Allows user to view information about themselves on their profile that they did not input.				
Traceability:	Use cases cf.	UC-013, UC-015			
	Test cases cf.	TC-009, TC-020, TC-021			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.39. System Functional Requirements: SF-E-05

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-E-05			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 01:45 AM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:45 AM			System: ☒ / ☐	
Description:	The system should summarize this data to be used by an advisor directing the student.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-E				
Justify why meeting SF-E-05 can contribute to the fulfilment of UF-E	Allows user to get assistance from advisors based on their profile.				
Traceability:	Use cases cf.	UC-001, UC-013, UC-015, UC-019			
	Test cases cf.	TC-009, TC-010, TC-016, TC-020, TC-021			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.40. System Functional Requirements: SF-F-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-F-01			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 12:16 AM			User: ☐ / ☐	
Modification:	Oct 05 2016 01:49 AM			System: ☒ / ☐	
Description:	The system should register the user within 5 seconds.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-F				
Justify why meeting SF-F-01 can contribute to the fulfilment of UF-F	Places a performance requirement on registration.				
Traceability:	Use cases cf.	UC-004			
	Test cases cf.	TC-003, TC-017, TC-018, TC-019			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.41. System Functional Requirements: SF-G-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-G-01			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 12:37 AM			User: ☐ / ☐	
Modification:	Oct 05 2016 12:37 AM			System: ☒ / ☐	
Description:	The system should only allow Advisors and System Developers to view the question log.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-G				
Justify why meeting SF-G-01 can contribute to the fulfilment of UF-G	Adds security to the question log.				
Traceability:	Use cases cf.	UC-006			
	Test cases cf.	TC-022			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.42. System Functional Requirements: SF-H-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SF-H-01			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 02:15 AM			User ☐ ☐	
Modification:	Oct 05 2016 02:15 AM			System ☒ ☐	
Description:	A user can provide answer quality feedback after asking a question.				
Priority:	Highest	High	✔ Medium	Low	Lowest
This Req. is Engineered From:	UF-H				
Justify why meeting SF-H-01 can contribute to the fulfilment of UF-H	Allows feedback to be submitted.				
Traceability:	Use cases cf.	UC-010			
	Test cases cf.	TC-005			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.43. System NonFunctional Requirements: SP-03-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SP-03-01			Type: Functional / Non-Functional	
Creation:	Oct 20 2016 01:53 AM			User ☐ ☐	
Modification:	Oct 20 2016 01:54 AM			System ☐ ☒	
Description:	The system should provide an answer to a question within 5 seconds.			Product (sub-type below)	
				Performance Requirements	
Priority:	Highest	High	Medium	✔ Low	Lowest
This Req. is Engineered From:	UP-03				
Justify why meeting SP-03-01 can contribute to the fulfilment of UP-03	Ensures an answer is quickly given by adding a performance requirement.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	TC-023			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.44. System NonFunctional Requirements: SP-01-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SP-01-01			Type: Functional / Non-Functional	
Creation:	Oct 05 2016 12:12 AM			User: ☐ / ☐	
Modification:	Oct 05 2016 12:14 AM			System: ☐ / ☒	
Description:	The system should only display information that the user knows is being displayed.			Product (sub-type below)	
				Dependability/Reliability/Security	
Priority:	Highest	✔ High	Medium	Low	Lowest
This Req. is Engineered From:	UP-01				
Justify why meeting SP-01-01 can contribute to the fulfilment of UP-01	Makes the user's profile more secure by allowing the user to specify which information is visible.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	TC-009			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.45. System NonFunctional Requirements: SO-01-01

Project Name:	Intelligent Academic Planner				
Requirement #:	SO-01-01			Type: Functional / Non-Functional	
Creation:	Sep 23 2016 12:42 PM			User: ☐ / ☐	
Modification:	Oct 20 2016 01:59 AM			System: ☐ / ☒	
Description:	The system should log the user out after 1 hour of inactivity.			Organizational (sub-type below)	
				Development Requirements	
Priority:	Highest	High	Medium	✔ Low	Lowest
This Req. is Engineered From:	UO-01				
Justify why meeting SO-01-01 can contribute to the fulfilment of UO-01	Ensures a user is not logged on for too long, managing their session.				
Traceability:	Use cases cf.	N/A			
	Test cases cf.	TC-024			
Acknowledgment	*Generated from the CapStone Process Management System ©2015*				

Table 4.46. Mapping from user requirements to system requirements

Project Name: Intelligent Academic Planner			
User Requirements		System Requirements	
Req ID	Description	Req ID	Description
UF-A	A user can log in.	SF-A-01	The system should log-in a user within 5 seconds.
UF-B	A user can log out.	SF-B-01	The system should log-out a user within 5 seconds.
UF-C	A user can ask the system questions.	SF-C-01	The system should conduct textual analyses.
		SF-C-02	The system should be able to handle input from multiple well-known languages.
		SF-C-03	The system should recommend majors suitable for the user based on the personality assessment.
		SF-C-04	The system should gather data unique to each user.
		SF-C-05	The system should recommend courses based on the recommended majors.
UF-D	A user should receive multiple responses to a question.	SF-D-01	The system should show a minimum of 1 related search/question.
UF-E	A user can create a profile	SF-E-01	The system should allow between 100 and 600 words to describe a user's academic and professional interests.
		SF-E-02	The system should allow a user to submit 100 words of self-description about their personality.
		SF-E-03	The system should allow a user to view their personality assessments.
		SF-E-04	The system should create a personality assessment unique to each user based on the data gathered.
		SF-E-05	The system should summarize this data to be used by an advisor directing the student.
UF-F	A user can register	SF-F-01	The system should register the user within 5 seconds.

UF-G	A user can view a log of asked questions.	SF-G-01	The system should only allow Advisors and System Developers to view the question log.
UF-H	A user can provide information to improve accuracy of the system.	SF-H-01	A user can provide answer quality feedback after asking a question.
UO-01	A user's session should be managed.	SO-01-01	The system should log the user out after 1 hour of inactivity.
UP-01	A user's profile should be secure.	SP-01-01	The system should only display information that the user knows is being displayed.
UP-03	A user should receive a quick response after asking a question	SP-03-01	The system should provide an answer to a question within 5 seconds.

Table 8.2.1. Test Suite TS-001: Unit Tests

TS-001: Unit Tests

Test Case ID	Test Stage	Test Case Description
TC-001	Unit	Login with valid user and password
TC-002	Unit	logout
TC-003	Unit	Register with untaken email and valid password
TC-004	Unit	Ask question
TC-005	Unit	answer quality feedback
TC-006	Unit	Real time translation
TC-007	Unit	Analyze question
TC-008	Unit	submit profile
TC-009	Unit	generate assessment and save with visbility
TC-010	Unit	Respond yes to request
TC-011	Unit	view assessment
TC-012	Unit	train watson
TC-013	Unit	Login with incorrect password
TC-014	Unit	Login with incorrect username
TC-015	Unit	Login with both incorrect username and password
TC-016	Unit	Respond no to Request
TC-017	Unit	Register with invalid password
TC-018	Unit	Register with taken email
TC-019	Unit	Register with both taken email and invalid password
TC-020	Unit	Generate assessment and save with false visiblity
TC-021	Unit	Generate Assessment and do not save
TC-022	Unit	View Question Log
TC-023	Unit	Ask Question performance
TC-024	Unit	Log out after 1 hour of inactivity

Table 8.2.2. Test Case TC-001

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-001 (Unit Test)	
What To Test	Login with valid user and password	
Test Data Input	valid username, valid password	
Expected Result	User is logged in	
Traceability	Relevant User Req.(s)	UF-A
	Relevant System Req.(s)	SF-A-01
	Relevant Use Case(s)	UC-016
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.3. Test Case TC-002

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-002 (Unit Test)	
What To Test	logout	
Test Data Input		
Expected Result	User is logged out.	
Traceability	Relevant User Req.(s)	UF-B
	Relevant System Req.(s)	SF-B-01
	Relevant Use Case(s)	UC-018
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.4. Test Case TC-003

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-003 (Unit Test)	
What To Test	Register with untaken email and valid password	
Test Data Input	untaken email, valid password	
Expected Result	User is entered into database.	
Traceability	Relevant User Req.(s)	UF-F
	Relevant System Req.(s)	SF-F-01
	Relevant Use Case(s)	UC-004
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.5. Test Case TC-004

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-004 (Unit Test)	
What To Test	Ask question	
Test Data Input	question string	
Expected Result	Question is logged to database and proper response is displayed	
Traceability	Relevant User Req.(s)	UF-C
	Relevant System Req.(s)	SF-C-01,SF-C-02,SF-C-03,SF-C-04,SF-C-05
	Relevant Use Case(s)	UC-011
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.6. Test Case TC-005

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-005 (Unit Test)	
What To Test	answer quality feedback	
Test Data Input	feedback	
Expected Result	feedback is stored in database, thank you message is displayed	
Traceability	Relevant User Req.(s)	UF-H
	Relevant System Req.(s)	SF-H-01
	Relevant Use Case(s)	UC-010
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.7. Test Case TC-006

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-006 (Unit Test)	
What To Test	Real time translation	
Test Data Input	non-english text	
Expected Result	english text	
Traceability	Relevant User Req.(s)	UF-C
	Relevant System Req.(s)	SF-C-02
	Relevant Use Case(s)	UC-009
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.8. Test Case TC-007

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-007 (Unit Test)	
What To Test	Analyze question	
Test Data Input	question with ambiguous response	
Expected Result	another question to make answer less ambiguous	
Traceability	Relevant User Req.(s)	UF-C,UF-D
	Relevant System Req.(s)	SF-C-01,SF-C-03,SF-C-04,SF-C-05,SF-D-01
	Relevant Use Case(s)	UC-012
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.9. Test Case TC-008

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-008 (Unit Test)	
What To Test	submit profile	
Test Data Input	new profile information	
Expected Result	profile information is stored to database	
Traceability	Relevant User Req.(s)	UF-E
	Relevant System Req.(s)	SF-E-01,SF-E-02
	Relevant Use Case(s)	UC-013
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.10. Test Case TC-009

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-009 (Unit Test)	
What To Test	generate assessment and save with visbility	
Test Data Input	descriptive text, yes to saving assessment, yes to visible to advisors	
Expected Result	assessment is stored to database with true visibility	
Traceability	Relevant User Req.(s)	UF-E,UP-01
	Relevant System Req.(s)	SF-E-04,SF-E-05,SP-01-01
	Relevant Use Case(s)	UC-015
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.11. Test Case TC-010

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-010 (Unit Test)	
What To Test	Respond yes to request	
Test Data Input	yes to request	
Expected Result	assessment is visible to person that requested it	
Traceability	Relevant User Req.(s)	UF-E
	Relevant System Req.(s)	SF-E-05
	Relevant Use Case(s)	UC-001
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.12. Test Case TC-011

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-011 (Unit Test)	
What To Test	view assessment	
Test Data Input		
Expected Result	assessment is displayed	
Traceability	Relevant User Req.(s)	UF-E
	Relevant System Req.(s)	SF-E-03
	Relevant Use Case(s)	UC-005
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.13. Test Case TC-012

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-012 (Unit Test)	
What To Test	train watson	
Test Data Input	answer question pair	
Expected Result	question is properly answered when asked	
Traceability	Relevant User Req.(s)	UF-H
	Relevant System Req.(s)	
	Relevant Use Case(s)	UC-007
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.14. Test Case TC-013

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-013 (Unit Test)	
What To Test	Login with incorrect password	
Test Data Input	correct username, incorrect password	
Expected Result	User is not logged in. Error message is displayed	
Traceability	Relevant User Req.(s)	UF-A
	Relevant System Req.(s)	SF-A-01
	Relevant Use Case(s)	UC-016
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.15. Test Case TC-014

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-014 (Unit Test)	
What To Test	Login with incorrect username	
Test Data Input	incorrect username, correct password	
Expected Result	User is not logged in. Error message is displayed.	
Traceability	Relevant User Req.(s)	UF-A
	Relevant System Req.(s)	SF-A-01
	Relevant Use Case(s)	UC-016
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.16. Test Case TC-015

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-015 (Unit Test)	
What To Test	Login with both incorrect username and password	
Test Data Input	incorrect username, incorrect password	
Expected Result	User is not logged in. Error message is displayed.	
Traceability	Relevant User Req.(s)	UF-A
	Relevant System Req.(s)	SF-A-01
	Relevant Use Case(s)	UC-016
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.17. Test Case TC-016

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-016 (Unit Test)	
What To Test	Respond no to Request	
Test Data Input	no to request	
Expected Result	no changes made to visibility	
Traceability	Relevant User Req.(s)	UF-E
	Relevant System Req.(s)	SF-E-05
	Relevant Use Case(s)	UC-001
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.18. Test Case TC-017

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-017 (Unit Test)	
What To Test	Register with invalid password	
Test Data Input	untaken email, invalid password	
Expected Result	User is not stored in database. Error message indicating invalid password displayed.	
Traceability	Relevant User Req.(s)	UF-F
	Relevant System Req.(s)	SF-F-01
	Relevant Use Case(s)	UC-004
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.19. Test Case TC-018

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-018 (Unit Test)	
What To Test	Register with taken email	
Test Data Input	taken email, valid password	
Expected Result	User is not entered into database. Error is displayed indicating email is taken.	
Traceability	Relevant User Req.(s)	UF-F
	Relevant System Req.(s)	SF-F-01
	Relevant Use Case(s)	UC-004
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.20. Test Case TC-019

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-019 (Unit Test)	
What To Test	Register with both taken email and invalid password	
Test Data Input	taken email, invalid password	
Expected Result	User is not entered into database. Error is displayed indicating taken email.	
Traceability	Relevant User Req.(s)	UF-F
	Relevant System Req.(s)	SF-F-01
	Relevant Use Case(s)	UC-004
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.21. Test Case TC-020

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-020 (Unit Test)	
What To Test	Generate assessment and save with false visiblity	
Test Data Input	descriptive text, yes to saving assessment, no to visible to advisors	
Expected Result	Assessment is stored to database with false visibility	
Traceability	Relevant User Req.(s)	UF-E
	Relevant System Req.(s)	SF-E-04,SF-E-05
	Relevant Use Case(s)	UC-015
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.22. Test Case TC-021

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-021 (Unit Test)	
What To Test	Generate Assessment and do not save	
Test Data Input	descriptive text, no to saving assessment	
Expected Result	Assessment is not stored to database.	
Traceability	Relevant User Req.(s)	UF-E
	Relevant System Req.(s)	SF-E-04,SF-E-05
	Relevant Use Case(s)	UC-015
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.23. Test Case TC-022

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-022 (Unit Test)	
What To Test	View Question Log	
Test Data Input		
Expected Result	Question Log is displayed	
Traceability	Relevant User Req.(s)	UF-G
	Relevant System Req.(s)	SF-G-01
	Relevant Use Case(s)	UC-006
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.24. Test Case TC-023

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-023 (Unit Test)	
What To Test	Ask Question performance	
Test Data Input	Question with known answer	
Expected Result	Answer is received within 5 seconds	
Traceability	Relevant User Req.(s)	UP-03
	Relevant System Req.(s)	SP-03-01
	Relevant Use Case(s)	UC-011
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.2.25. Test Case TC-024

Project Name:	Intelligent Academic Planner	
Test Suite	TS-001: Unit Tests	
Test Case ID	TC-024 (Unit Test)	
What To Test	Log out after 1 hour of inactivity	
Test Data Input		
Expected Result	User is logged out after 1 hour of inactivity	
Traceability	Relevant User Req.(s)	UO-01
	Relevant System Req.(s)	SO-01-01
	Relevant Use Case(s)	
Acknowledgment: Generated from the CapStone process management system ©2015		

Table 8.3.1. Execution Report of Test Case TC-001

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-001	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	click left side bar
	2	click user icon
	3	click perfered login method
	4	input valid credential

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/15/2016	User is not logged in	Fail	Not implemented	10/30/2016 by Allen
2	Daria	11/15/2016	User is logged in	Pass		
3	Daria	2/1/2017	User is logged in	Pass		

Execution Summary:	Success

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.2. Execution Report of Test Case TC-002

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-002	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	click left sidebar
	2	click user icon
	3	click logout button
	4	back to main page

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/15/2016	User is not logged out	Fail	Not yet implemented	10/30/2016 by Allen
2	Daria	11/15/2016	User is logged out	Pass		
3	Daria	2/1/2017	User is logged out	Pass		

Execution Summary:	Success

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.3. Execution Report of Test Case TC-003

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-003	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	click left side bar
	2	click user icon
	3	click sign in icon
	4	click sign up
	5	input username, email, password
	6	go to profile page

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/15/2016	User is not added to database	Fail	Not yet implemented	10/30/2016 by Allen
2	Daria	11/15/2016	User is added to database	Pass		
3	Daria	2/1/2017	User is added to database	Pass		

Execution Summary:	Success

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.4. Execution Report of Test Case TC-004

Project Name:	Intelligent Academic Planner						
Test Case ID:	TC-004						
Testing Tools Used:	Mocha						
Testing Type:	Agile (automated) testing						
Execution Steps:	1	input question					
	2	view answer					
Test Execution Records:							
#	Tester	Test Date	Actual Result	Status	Defect	Correction	
1	Daria	11/15/2016	Question is not answered	Fail	Not yet implemented		
Execution Summary:	Not yet implemented						
Acknowledgment: Generated from the CapStone process management system ©2015							

Table 8.3.5. Execution Report of Test Case TC-005

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-005	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	User asks question
	2	System displays response and requests feedback
	3	User enters feedback
	4	See that feedback is stored in database

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/15/2016	Feedback is not added to database and thank you message not displayed	Fail	Not yet implemented	

Execution Summary:	Not yet implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.6. Execution Report of Test Case TC-006

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-006	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	User enters untranslated text
	2	User requests translation
	3	See that text becomes translated

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/15/2016	No text outputted	Fail	Not yet implemented	

Execution Summary:	Not yet implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.7. Execution Report of Test Case TC-007

Project Name:	Intelligent Academic Planner					
Test Case ID:	TC-007					
Testing Tools Used:	Mocha					
Testing Type:	Agile (automated) testing					
Execution Steps:	1	User enters ambiguous question				
	2	See that system responds with another question relevant to the question asked				
Test Execution Records:						
#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/15/2016	No output	Fail	Not yet implemented	
Execution Summary:	Not yet implemented					
Acknowledgment: Generated from the CapStone process management system ©2015						

Table 8.3.8. Execution Report of Test Case TC-008

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-008	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	User enters profile information
	2	User submits profile information
	3	See that profile information is stored in database

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/15/2016	Profile information not stored to database	Fail	Not yet implemented	

Execution Summary:	Not yet implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.9. Execution Report of Test Case TC-009

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-009	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	User requests assessment
	2	User requests to save assessment
	3	User requests that assessment is made visible to advisers
	4	See that assessment is stored in database with true visibility

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/15/2016	No assessment shown	Fail	Not yet implemented	

Execution Summary:	Not yet implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.10. Execution Report of Test Case TC-010

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-010	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	User receives a request to view assessment
	2	User agrees to allow access to assessment
	3	See that visibility of assessment is change to true in database

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/15/2016	Visibility not changed	Fail	Not yet implemented	

Execution Summary:	Not yet implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.11. Execution Report of Test Case TC-011

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-011	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	User requests to view assessment
	2	See that assessment is displayed

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/15/2016	Assessment not shown	Fail	Not yet implemented	

Execution Summary:	Not yet implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.12. Execution Report of Test Case TC-012

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-012	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	go to retrieve and rank
	2	go to corpus
	3	input queststion and answer
	4	select answer

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/15/2016	No change in answer quality	Fail	Not yet implemented	10/30/2016 by Allen
2	Daria	11/15/2016	Question successfully answered	Pass		

Execution Summary:	Success

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.13. Execution Report of Test Case TC-013

Project Name:	Intelligent Academic Planner
Test Case ID:	TC-013
Testing Tools Used:	Mocha
Testing Type:	Parameter value coverage
Execution Steps:	1 User attempts to login with invalid password 2 See that user is not logged in and you stay on current page.

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/18/2016	User is not logged in but no error is displayed	Fail	Not implemented	10/30/2016 by Allen
2	Daria	2/1/2017	User is not logged in and you stay on current page	Pass		

Execution Summary:	Passing as expected

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.14. Execution Report of Test Case TC-014

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-014	
Testing Tools Used:	Mocha	
Testing Type:	Parameter value coverage	
Execution Steps:	1	User attempts to login with incorrect username
	2	See that user is not logged in and you stay on current page

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/18/2016	User is not logged in but error is not displayed	Fail	Not implemented	10/30/2016 by Allen
2	Daria	2/1/2017	User is not logged in and you stay on current page	Pass		

Execution Summary:	Passing as expected

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.15. Execution Report of Test Case TC-015

Project Name:	Intelligent Academic Planner
Test Case ID:	TC-015
Testing Tools Used:	Mocha
Testing Type:	Parameter value coverage
Execution Steps:	1 User attempts to login with both incorrect username and password
	2 See that error message is displayed and you stay on current page

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/18/2016	User is not logged in but no error is displayed	Fail	Not implemented	10/30/2016 by Allen
2	Daria	2/1/2017	User is not logged in and stays on current page	Pass		

Execution Summary:	Passing as expected

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.16. Execution Report of Test Case TC-016

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-016	
Testing Tools Used:	Mocha	
Testing Type:	Parameter value coverage	
Execution Steps:	1	User receives request to view assessment
	2	User denies request
	3	See that request's visibility stays false in database

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/18/2016	Can't Test	Fail	Not implemented	

Execution Summary:	Not implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.17. Execution Report of Test Case TC-017

Project Name:	Intelligent Academic Planner
Test Case ID:	TC-017
Testing Tools Used:	Mocha
Testing Type:	Parameter value coverage
Execution Steps:	1 User attempts to register with invalid password 2 See that a user is not added to database and you stay on current page

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/18/2016	User is not added to database but no error is displayed	Fail	Not implemented	10/30/2016 by Allen
2	Daria	2/1/2017	User is not added to database and you stay on current page	Pass		

Execution Summary:	Passing as expected

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.18. Execution Report of Test Case TC-018

Project Name:	Intelligent Academic Planner
Test Case ID:	TC-018
Testing Tools Used:	Mocha
Testing Type:	Parameter value coverage
Execution Steps:	1 User attempts to register with a taken email 2 See that the user is not added to database and you stay on current page

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/18/2016	User is not added to database but no error is displayed	Fail	Not implemented	10/30/2016 by Allen
2	Daria	2/1/2017	User is not added to database and you stay on current page	Pass		

Execution Summary:	Passing as expected

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.19. Execution Report of Test Case TC-019

Project Name:	Intelligent Academic Planner
Test Case ID:	TC-019
Testing Tools Used:	Mocha
Testing Type:	Parameter value coverage
Execution Steps:	1 User attempts to login with both taken email and invalid password
	2 See that the user is not added to database and you stay on current page

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	10/18/2016	User is not added to database but no error is displayed	Fail	Not implemented	10/30/2016 by Allen
2	Daria	2/1/2017	User is not added to database and you stay on current page	Pass		

Execution Summary:	Passing as expected

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.20. Execution Report of Test Case TC-020

Project Name:	Intelligent Academic Planner
Test Case ID:	TC-020
Testing Tools Used:	Mocha
Testing Type:	Parameter value coverage
Execution Steps:	1 User requests to generate assessment
	2 User requests to save assessment
	3 User requests that advisers cannot view their assessment
	4 See that the assessment is stored in the database with false visiblity

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/18/2016	Can't Test	Fail	Not Implemented	

Execution Summary:	Not Implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.21. Execution Report of Test Case TC-021

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-021	
Testing Tools Used:	Mocha	
Testing Type:	Parameter value coverage	
Execution Steps:	1	User requests to generate assessment
	2	User requests to not save assessment
	3	See that assessment is not stored in database

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/18/2016	Can't Test	Fail	Not Implemented	

Execution Summary:	Not Implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.22. Execution Report of Test Case TC-022

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-022	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	Developer requests to view question log
	2	See that question log is displayed

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/17/2016	Can't Test	Fail	Not Implemented	

Execution Summary:	Not Implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.23. Execution Report of Test Case TC-023

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-023	
Testing Tools Used:	Mocha	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	User asks question
	2	See that system responds within 5 seconds

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/18/2016	Can't Test	Fail	Not Implemented	

Execution Summary:	Not Implemented

Acknowledgment: Generated from the CapStone process management system ©2015

Table 8.3.24. Execution Report of Test Case TC-024

Project Name:	Intelligent Academic Planner	
Test Case ID:	TC-024	
Testing Tools Used:	Manual	
Testing Type:	Agile (automated) testing	
Execution Steps:	1	User is inactive for 1 hour
	2	See that system logs user out

Test Execution Records:

#	Tester	Test Date	Actual Result	Status	Defect	Correction
1	Daria	11/18/2016	User stays logged in	Fail	Not implemented	
2	Daria	2/1/2017	User stays logged in	Fail	Not Implemented	

Execution Summary:	Not Implemented

Acknowledgment: Generated from the CapStone process management system ©2015