Software Requirements Specification

for

College Community App (Student Connect)

Version 1.0 approved

Prepared by Anumeha Agrawal, Rosa Anil George, Selvan Sunitha

NITK Surathkal

17 Jan 2019

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Version 2	31/01/2019	Adding content about system features	2
Version 3	06/02/2019	Adding content about Non-functional requirements	3
Version 4	21/02/2019	Adding use case diagram, user classes, references and communication interface	4

1. Introduction

1.1 Purpose

The purpose of this document is to build a college community web application system. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli.

1.2 Document Conventions

To make the document more effective and readable Times New Roman font style was used and font size 16 for the title, 14 for headings and 11 for the content. The headings and title are in bold and have been indexed.When writing the SRS document for College Event Web App the following terminologies are used:

- Web server (WS)- The container of content comprising of two layers overlay, which is a collection of Web service host (e.g. Apache, Tomcat), Service Level Agreement (SLA): allocator, and policy agent, and core, which refers to the underlying hardware infrastructure.
- DB - Database

1.3 Intended Audience and Reading Suggestions

This document has been prepare for the developers and the users of the web app. It has been implemented as a course project.
Developers are advised to go through the Introduction (Section 1), Overall Description (Section 2). Section 4 describes all the features implemented as part of the application. Users are requested to go through Section 2.6 in addition to the above Sections for additional User resources and documentations.

1.4 Product Scope

Student Connect is a college counselling platform connecting university applicants to university students. We also recommend colleges based on several factors like applicant's interests, economic status, scores and ethnicity. Colleges are rated based on university student's reviews. Genuine and authentic because after all, a college is defined by its students. Student Connect directly connects applicants to existing students(counsellors) who counsel them with the whole procedure and give them a student's perspective of college

1.5 References

1.5.1 https://materializecss.com/
1.5.2 https://azure.microsoft.com/en-in/services/cognitive-services/
1.5.3 https://www.nltk.org/

2. Overall Description

2.1 Product Perspective

A large number of students go away from home to study abroad and it is certainly not an easy job. Parents in India usually save 52% of their salary to fund their children in future. College recommendation and counselling is very expensive and not everyone can afford it. Our aim is to bridge the gap between students and counsellors. Shortlisting colleges is a hassle and therefore we recommend the best colleges suited for the applicant. All these services are provided at a minimal cost, benefiting counsellors earn some quick money and build students' careers.

2.2 Product Functions

1. Home view displays information on the product. It's the same home view for unauthenticated as well as authenticated users. The navbar in this view though, behaves much differently for authenticated users
2. Profile view contains all details of the user. It only appears to authenticated users.
3. Recommendation view consists of two pages, one being the form and the other to display result to the query in this form. The form takes in various inputs to build its query and display a result. The inputs are ethnicity, sat score, gpa, interests and income
4. College view lists all the various colleges along with their ratings and various characteristics like the ethnicity, median tuition per annum, specialisations etc.

2.3 User Classes and Characteristics

2.3.1 Student class :- Interact with the interface for the following tasks

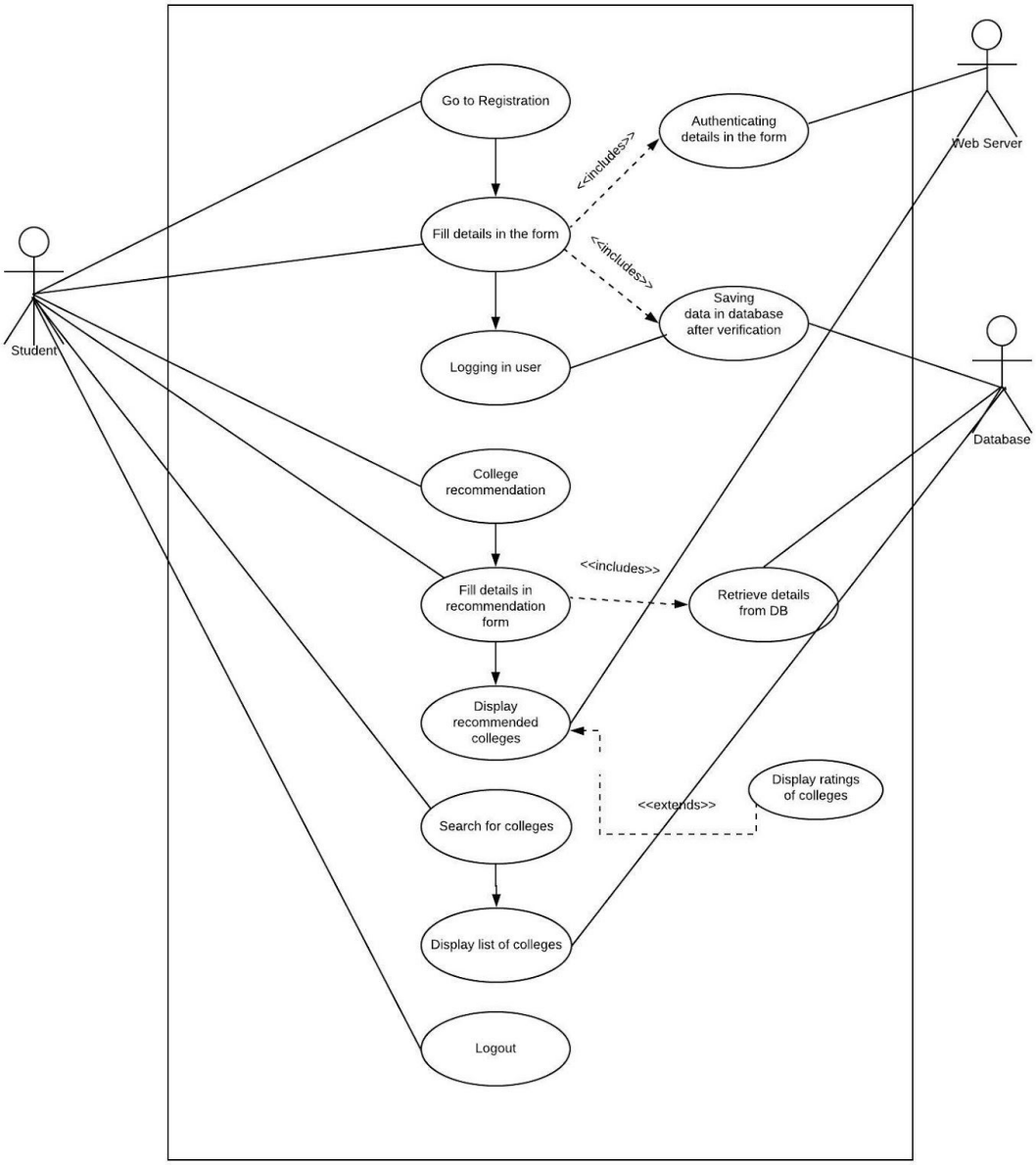
- Fill in authentication form and create login credentials
- Fill form to view a list of recommended colleges on the basis of various parameters
- View the list of counsellors and contact them through email
- Chat with other similar users using the group chat feature

2.3.2 Database user class :- Interact with the interface for the following tasks

- Save the details of student in the database
- Help in retrieval of details from database
- Fetch the list of recommended colleges

2.3.3 Web server user class :- Interact with the interface for the following tasks

- Authenticate and verify the details in the login form
- Display the list of recommended colleges after performing NLP and data analytics tasks



2.4 Operating Environment

1. This product will operate on any operating system and version
2. This will require setting up django in the development environment
3. Software requirements:- Python 3+, Django
4. Testing will be component based using Django's inbuilt testing module.
5. Microsoft Cognitive Services
6. Azure account with subscription key
7. SQLite DB
8. Javascript

2.5 Design and Implementation Constraints

1. Chat system without opening multiple windows for testing
2. Azure credentials which need to be renewed regularly unless the API is paid for a long duration.
3. Any user who has an account on the platform can join the chat room, there is no private chat.

User Documentation

There will be a user manual available on the web application that will have a clear description of all the functionalities as well as a step-by-step explanation on how the various features can be used and benefited from.

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

2.6 Assumptions and Dependencies

1. The web application will be run locally and attempts will be made to host the web application on heroku.
2. Several python packages will be required and will be specified in the package.json files. (Exact packages to be determined at the time of development)

3. External Interface Requirements

3.1 User Interfaces

1. User interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

3.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

3.3 Software Interfaces

It is connected to Azure Cloud Services and we use sqlite for the DB.

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

3.4 Communications Interfaces

It requires connectivity to the Internet. The application can be accessed using a web browser. It also includes an email option. We will be using HTTP as the communication standard.

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

4. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 Authentication feature

4.1.1 Description and Priority

Implementing authentication system with Django's inbuilt login authentication module. This view will accept username and password.

Difficulty - 6

The registration view will be customised to accept username, email and password with certain constraints on the password.

Difficulty - 7

4.1.2 Stimulus/Response Sequences

1. Users will need to enter data in text boxes and click on the register button to create an account.
2. They will then need to go to the login page and re-enter their credentials.
3. This will take the user to their dashboard.

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

In order to create the authentication module:-

REQ 1:- A valid form needs to be present. For eg:- the password entered needs to match with the rules, eg - password needs to be of min 6 characters, 1 number needs to be present, a special character needs to be present etc

REQ 2:- The submit button should trigger the validation module.

REQ 3:- Only in case of successful validation, the page should redirect to dashboard.

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:
REQ-2:

4.2 Display list of counsellors

4.2.1 Description and Priority

Display a dropdown menu with the list of colleges. For each college a list of counsellors will be shown. Each counsellor's details will be displayed with ratings and contact details.

Difficulty : 6

4.2.2 Stimulus/Response Sequences

1. The counsellor tab will be present in the nav bar. An authenticated user can view the list of counsellors for a college.
2. The drop down will have a list of colleges and one college needs to be selected.
3. On clicking the submit button a list of counsellors is displayed.

4.2.3 Functional Requirements

In order to create counsellors module:

REQ 1 :- Working trigger for search method

REQ 2:- List view to show a list of counsellors.

5. Other Nonfunctional Requirements

In this section, last group of the requirements which is nonfunctional requirements will be explained in detail. Nonfunctional requirements include performance requirements, security requirements and portability requirements.

5.1 Performance Requirements

The system must be interactive and the delays involved must be less. So in every action-response of the system, there are no immediate delays. In case of opening windows forms, of popping error messages and saving the settings or sessions there is delay much below 2 seconds, In case of opening DBs, sorting questions and evaluation there are no delays and the operation is performed in less than 2 seconds for opening, sorting, computing, posting > 95% of the files. Also when connecting to the WS the delay is based editing on the distance of the 2 systems and the configuration between them so there is high probability that there will be or not a successful connection in less than 20 seconds for sake of good communication.

Performance requirement by the user side is, the web application must be developed as a lightweight app so that it can work on almost any platform even with slower internet connections.

5.2 Safety Requirements

The safety requirements include managing and keeping the user's data secure as in case of any failure, the user data must not be lost nor should be vulnerable to attack by any malicious source. Information transmission should be securely transmitted to server without any changes in information.

5.3 Security Requirements

Since this software will be hosted on cloud server, all the user data will be kept on the cloud server. Product should be able to protect privacy of user data. Workspace of the user should only be accessed through user own credentials and any other user should not be able to access any other user's private data.

The user will be restricted in terms of user rights and should be restricted so as to not harm the system by the programs they run or by the commands they run on the terminal.

Since all the data will be transferred on the web, the system should also use an encryption and decryption mechanism so that only the intended user can decode and work on the data.

5.4 Software Quality Attributes

The quality of the system is maintained in a way that is user friendly to all users. The software quality attributes assumed include:

5.4.1 Availability

If the internet service gets disrupted while sending information to the server, the information can be send again for verification.

5.4.2 Usability

As the system is easy to handle and navigates in the most expected way with no delays. In that case the system program reacts accordingly and transverses quickly between its states.

5.5 Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>