Software Requirements Specification

for

Advising Website

Version 1.0 approved

Prepared by Delaney Flaherty, Pah Meh, Ryan Rutledge

Western Kentucky University

2/6/2025

Table of Contents

[separate page]

Revision History

Name	**Date**	**Reason For Changes**	**Version**
Delaney, Ryan, Pah	Feb. 6, 2025	Initial creation of document	1.0

Name	**Contribution**
Delaney Flaherty	Purpose, Project scope, User classes and characteristics, Operating environment, System features, Internationalization and localization requirements, Communications interfaces, Document conventions, External interface requirements, Hardware interfaces
Ryan Rutledge	Product perspective, Assumptions and dependencies, usability, Logical Data Model, Data dictionary, Data Acquisition, Integrity, Retention, and Disposal, Software interfaces, Safety
Pah Meh	Design and implementation constraints, reports, performance, security

1. Introduction

<The introduction presents an overview to help the reader understand how the SRS is organized and how to use it.>

Purpose

<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers.>

This document specifies the software requirements for the Western Kentucky University Advising Website. The website is designed to assist students and advisors select courses based on their major and previous courses taken. The intended audience for this document includes developers and users such as students and advisors. Developers and testers will be able to refer to this initial version to make sure all of the features, functional and nonfunctional, are implemented. Users will be able to read this document and learn about our website and the features it has.

Document Conventions

<Describe any standards or typographical conventions used, including the meaning of specific text styles, highlighting, or notations. If you are manually labeling unique requirement identifiers, you might specify the format here for anyone who needs to add one later.>

This document is designed to inform, and it is formatted to ensure clarity and consistency. The bold text represents the section headings and titles. The italicized text is used to describe the intended section information. The bulleted lists are to organize and present structured information clearly. The text without any formatting is information written about our software depending on the section.

Project Scope

<Provide a short description of the software being specified and its purpose. Relate the software to user or corporate goals and to business objectives and strategies. If a separate vision and scope or similar document is available, refer to it rather than duplicating its contents here. An SRS that specifies an incremental release of an evolving product should contain its own scope statement as a subset of the long-term strategic product vision. You might provide a high-level summary of the major features the release contains or the significant functions that it performs.>

The advising website is a web-based platform designed to assist students at Western Kentucky University in planning their academics. The purpose is to make schedule planning for their future semesters easier by uploading their unofficial transcript, allowing it to be analyzed, and generating a personalized courseload recommendation. This website will also provide a basic messaging system to facilitate communication between advisors and advisees, so if there are any questions or concerns, they will be able to communicate about it. This software aligns with the university's goals of enhancing academic advising efficiency. This website will relieve administrative workload and provide students with software that returns timely and informed academic planning based on their course history and by automating the course recommendations. This software is designed for easy use and making sure it integrates seamlessly with existing university systems and information.

References

<List any documents or other resources to which this SRS refers. Include hyperlinks to them if they are in a persistent location. These might include user interface style guides, contracts, standards, system requirements specifications, interface specifications, or the SRS for a related product. Provide enough information so that the reader can access each reference, including its title, author, version number, date, and source, storage location, or URL.>

2. Overall Description

<This section presents a high-level overview of the product and the environment in which it will be used, the anticipated users, and known constraints, assumptions, and dependencies.>

Product Perspective

<Describe the product's context and origin. Is it the next member of a growing product line, the next version of a mature system, a replacement for an existing application, or an entirely new product? If this SRS defines a component of a larger system, state how this software relates to the overall system and identify major interfaces between the two. Consider including visual models such as a context diagram or ecosystem map to show the product's relationship to other systems.>

This product is an entirely new product that aims to automate the process of advising students for what classes they need to take next semester in order to graduate by a certain time. This would work hand in hand with advisers and advisees and would pull information from WKU’s website about when certain classes are available and their prerequisites.

User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product and describe their pertinent characteristics. Some requirements might pertain only to certain user classes. Identify the favored user classes. User classes represent a subset of the stakeholders described in the vision and scope document. User class descriptions are a reusable resource. If available, you can incorporate user class descriptions by simply pointing to them in a master user class catalog instead of duplicating information here.>

This product is designed to serve multiple different groups of users within Western Kentucky University. Students at any academic level can use this whether they are here for their undergrad, master’s or doctorate. This is also for academic advisors. Undergraduate students will use this platform to upload their unofficial transcript and receive a personalized course recommendation based on previous courses taken and the amount of credit hours they are looking to take in the next semester. They can use this for independent research on their academic path to help keep themselves informed, or they can use it to provide talking points and areas of concern for their advisor. This program will also be helpful for graduate students. The use will essentially be the same as undergrad students. They will use it for course recommendations and as a means of communication with their advisor. Advisors will use this product to streamline their advising process. They can use this as a backup to make sure they have their recommended class schedule correct and eliminate human error. Advisors will also have the ability to send out mass alerts to students about registration dates or to schedule and advising meeting. Usability and accessibility are important to ensure that all users can efficiently navigate and use it's features.

Operating Environment

<Describe the environment in which the software will operate, including the hardware platform; operating systems and versions; geographical locations of users, servers, and databases; and organizations that host the related databases, servers, and websites. List any other software components or applications with which the system must peacefully coexist. If extensive technical infrastructure work needs to be performed in conjunction with developing the new system, consider creating a separate infrastructure requirements specification to detail that work.>

The advising website will be developed using Django, a Python-based web framework, and will be deployed using Apache HTTP Server. The website will be designed to run on modern web browsers, such as Google Chrome, Safari, etc. The backend will use a relational database management system to store user data, course requirements, and messages. Since this is made specifically for Western Kentucky University, all users, servers, and databases will be located in the U.S.

Design and Implementation Constraints

<Describe any factors that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing or memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; programming language requirements or restrictions.>

Since this is unpaid work, there are some financial restraints. When searching for AI parsers or any other tools, the group may be limited to what they are able to choose. The options limited to us must be open source or free. We cannot use tools that include paying a subscription or a one-time fee.

Since the team does not have access to WKU's topnet login system, we can't have a login system connected to that. The team will have to create their own login system or use an API available from Google or Microsoft as backup. Most likely, we will create our own.

Since the students will upload an unofficial transcript, there will be security concerns and solutions to reprimand that. Any type of transcript has sensitive information. Our system should store the uploaded file temporarily. After use, the file should be deleted from our system permanently.

Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, reuse expectations, issues around the development or operating environment, or constraints. The project

could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors outside its control.>

Some assumptions that we will be making are when certain classes will and will not be offered based on patterns from previous semesters while the classes for the upcoming semesters have not been posted. We will also not have full access to WKU's database in order to provide more information about classes as well as allow students and advisors to sign in to their WKU accounts.

3. 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, stimulus, response, or combinations of these, whatever makes the most logical sense for your product.>

3.1 Upload and Process Transcript

3.1.1 Description

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority.>

The upload transcript feature allows students or advisors to insert a PDF of their unofficial transcript. The system will take the important information off that document and add it to our database which will be used to help recommend the necessary classes needed. This feature is high priority because if we don't have this feature our system will not work as intended.

3.1.2 Stimulus/Response Sequences

<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.>

- Stimulus: User inserts and uploads transcript

- Response: The system processes and parses the transcript to retrieve the important course and major/minor information

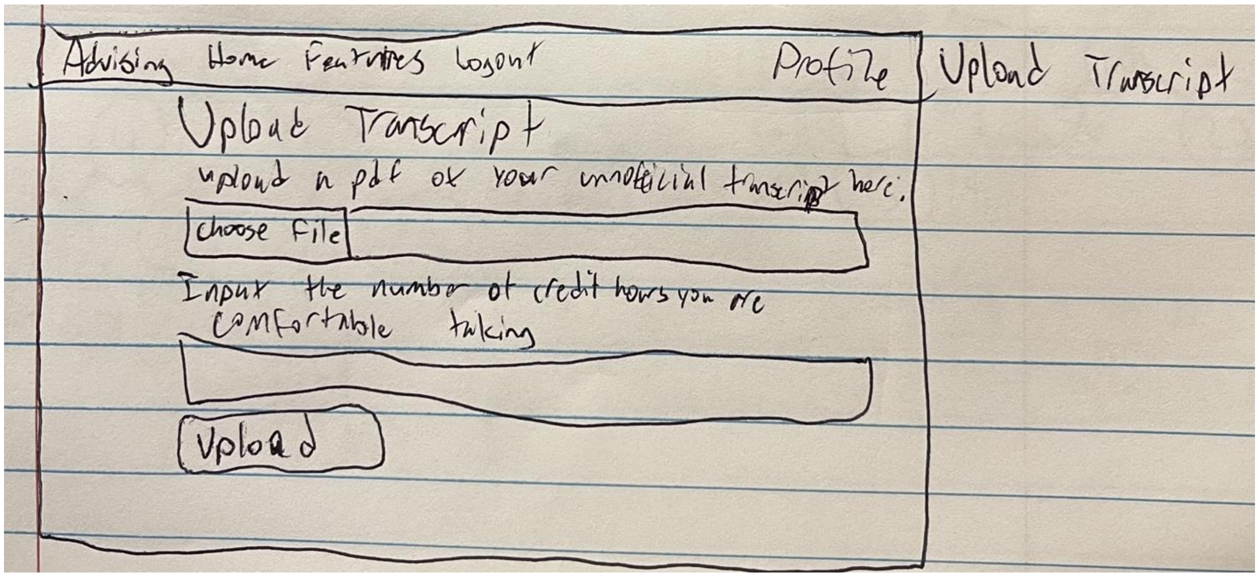
3.1.3 Functional Requirements

<Itemize the specific functional requirements associated with this feature. These are the software capabilities that must be implemented for the user to carry out the feature's services or to perform a use case. Describe how the product should respond to anticipated error conditions. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

- The website must allow users to insert their unofficial transcript and desired credit hours via website form.

- The system must validate that the inserted file is a PDF.
- The system must be able to extract course data from the PDF
- The system must be able to match completed courses with degree requirements to recommend classes.

The system must be able to notify the user of any parsing errors or unsupported file formats.


Advising
Home
Features
Logout
Profile
Upload Transcript
Upload Transcript
upload a pdf of your unofficial transcript here:
choose file
Input the number of credit hours you are comfortable taking
Upload

3.2 Course Recommendation System

3.2.1 Description

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority.>

The course recommendation system analyzes the parsed transcript and suggests courses based on completed courses, degree requirements from WKU’s website, and desired credit hours. This is a high priority feature.

3.2.2 Stimulus/Response Sequences

<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.>

- Stimulus: System processes and compares degree requirements after user upload.
- Response: The system generates a list of suggested courses based off the degree requirements from WKU’s website, completed courses, and desired credit hours.

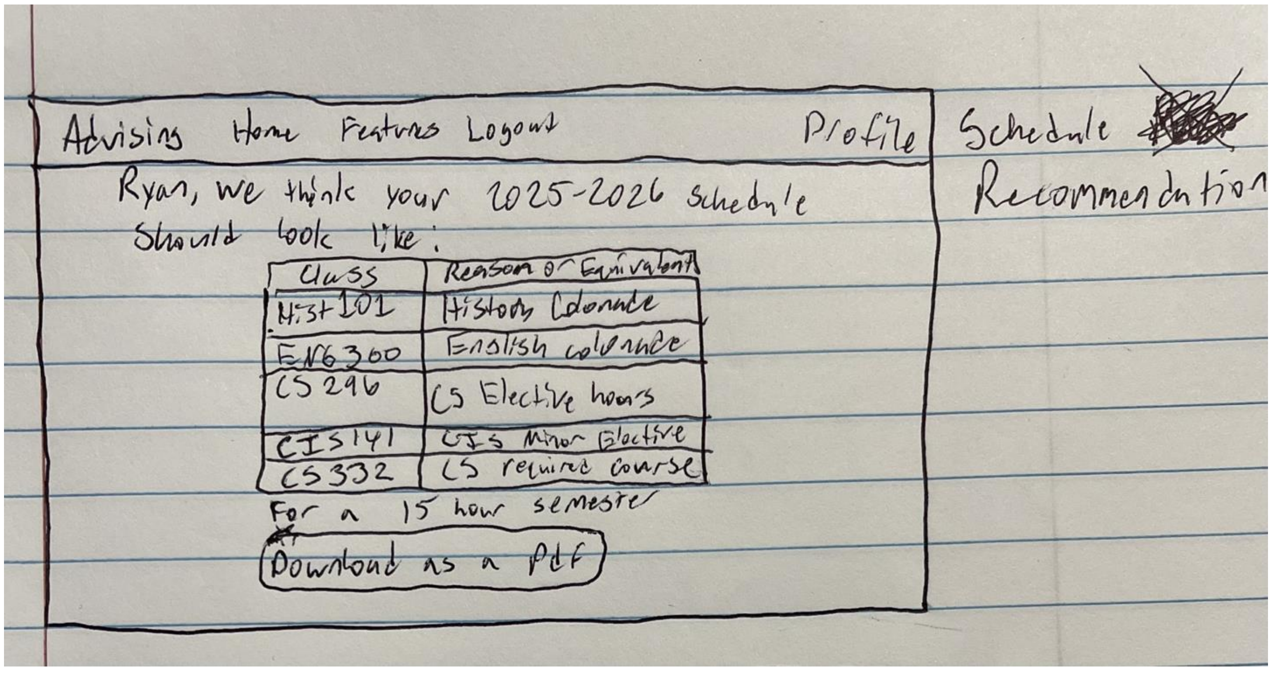
3.2.3 Functional Requirements

<Itemize the specific functional requirements associated with this feature. These are the software capabilities that must be implemented for the user to carry out the feature's services or to perform a use case. Describe how the product should respond to anticipated error conditions. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

- System must analyze the student's completed courses and degree requirements from WKU's website.

- System must check for prerequisites before recommending a class.

The system must display recommended classes for users to be able to read and download as necessary.



Ryan, we think your 2025-2026 schedule should look like:

Class	Reason or Equivalent
Hist 101	History Colonnade
ENG 300	English colonnade
CS 296	CS Elective hours
CIS 141	CIS Minor Elective
CS 332	CS required course

For a 15 hour semester

3.3 Messaging System

3.3.1 Description

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority.>

The messaging system feature enables communication between the student and advisor. We also plan to give the advisors the ability to send out a mass alert for reminders such as registration dates and to schedule advising appointments. This is a high priority as we want there to be open communication between students and advisors.

3.3.2 Stimulus/Response Sequences

<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.>

- Stimulus: Student or advisor sends a message

Response: The system stores and delivers the message to the intended recipient

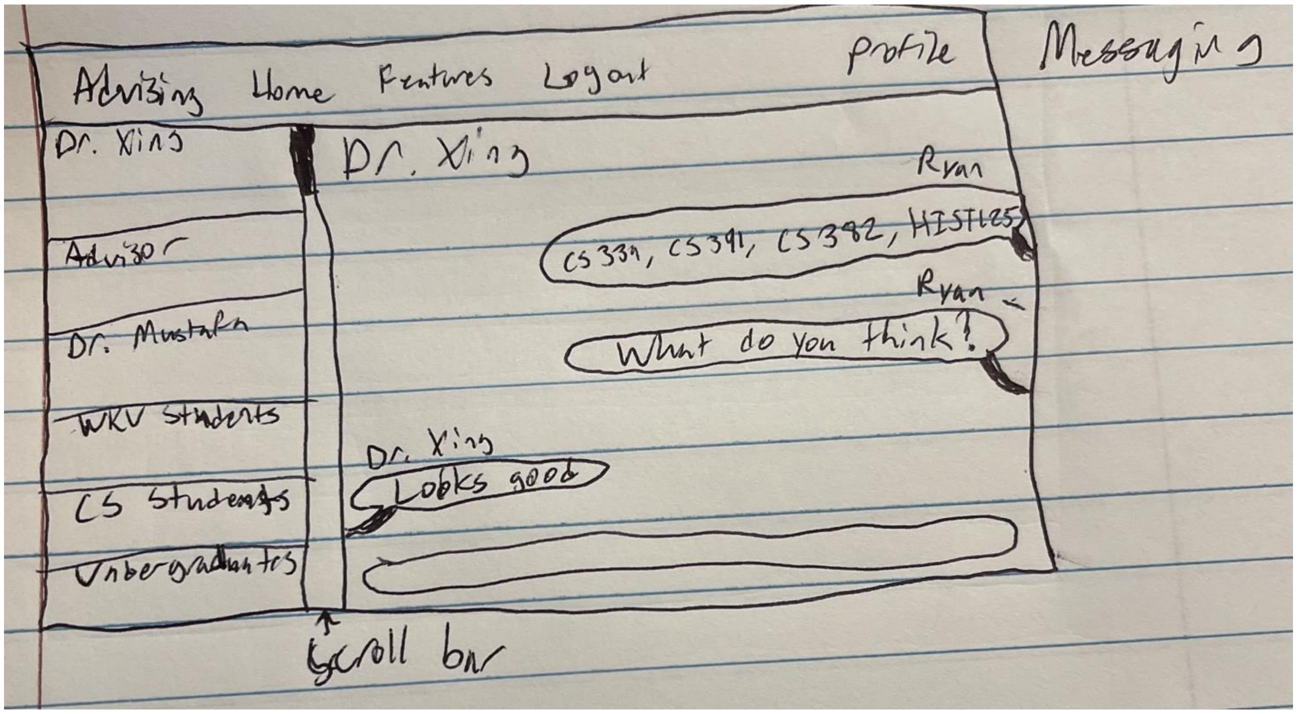
3.3.3 Functional Requirements

<Itemize the specific functional requirements associated with this feature. These are the software capabilities that must be implemented for the user to carry out the feature's services or to perform a use case. Describe how the product should respond to anticipated error conditions. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

- The system must allow students and advisors to send messages to their assigned advisor and students.

- The system must allow advisors and students to respond to received messages.

The system must be able to store messages for future reference.



3.4 User Authentication and Role Management

3.4.1 Description

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority.>

This feature ensures that only authorized users can access the features of this platform. This also includes role-based access control. Advisors will have more features and accessibility than students will. Admins will have the greatest number of features and access to the website and database. This is a high priority as it is a major feature for security.

3.4.2 Stimulus/Response Sequences

<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.>

- Stimulus: User attempts to log in with their credentials.

Response: The system verifies the credentials and grants access.

Stimulus: Advisor logs in

Response: System grants access to more features such as sending out mass messages to students

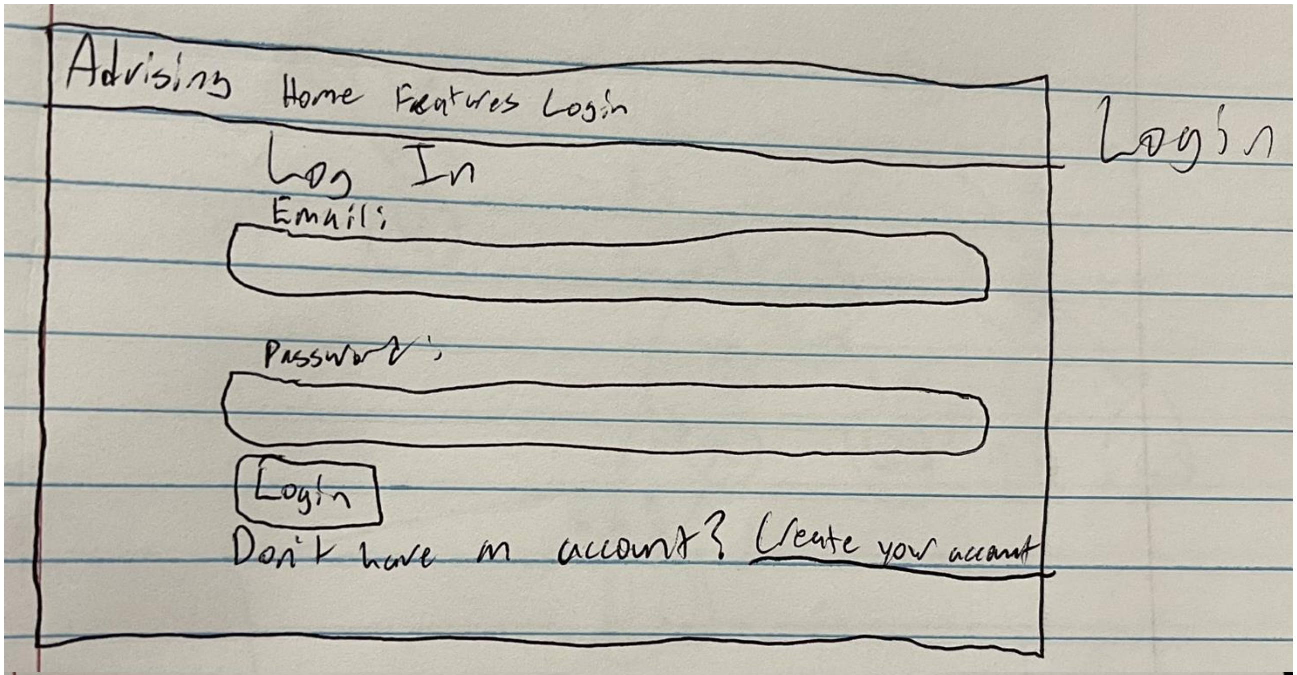
3.4.3 Functional Requirements

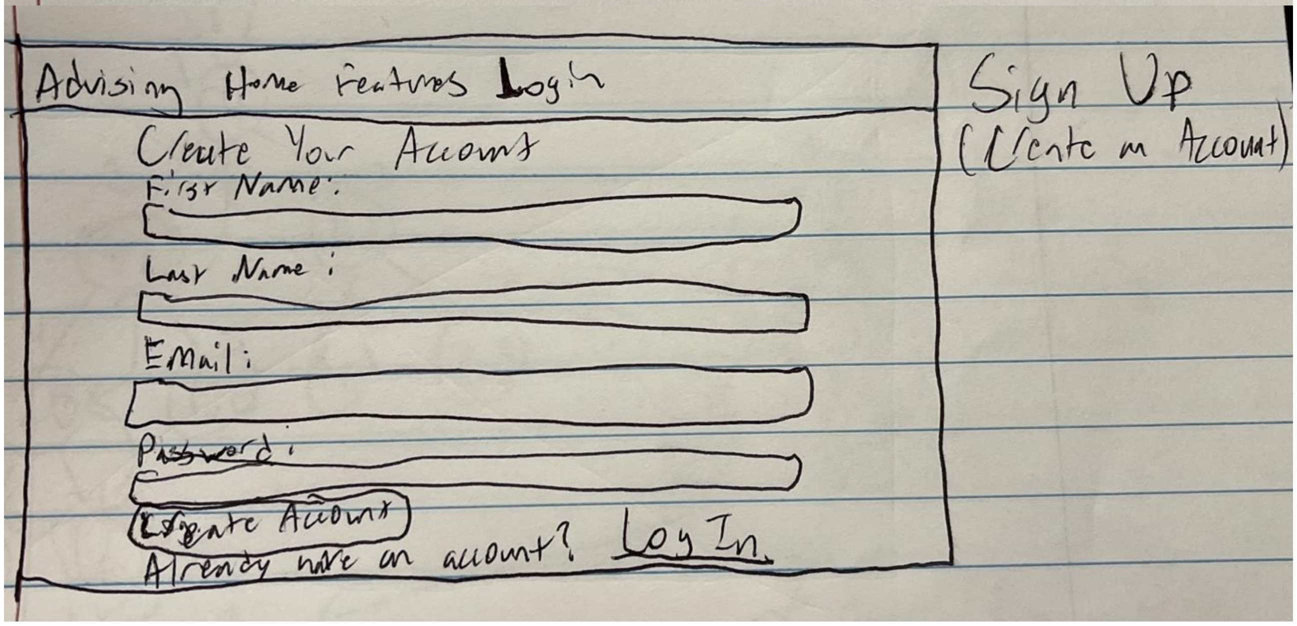
<Itemize the specific functional requirements associated with this feature. These are the software capabilities that must be implemented for the user to carry out the feature's services or to perform a use case. Describe how the product should respond to anticipated error conditions. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

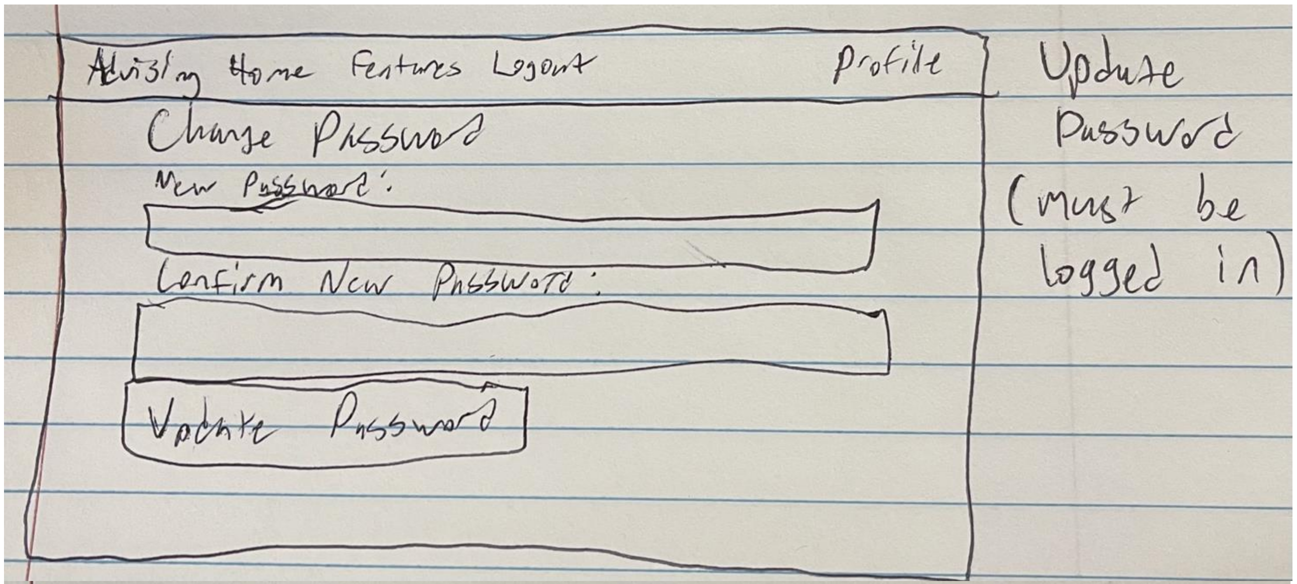
System must require login for authentication

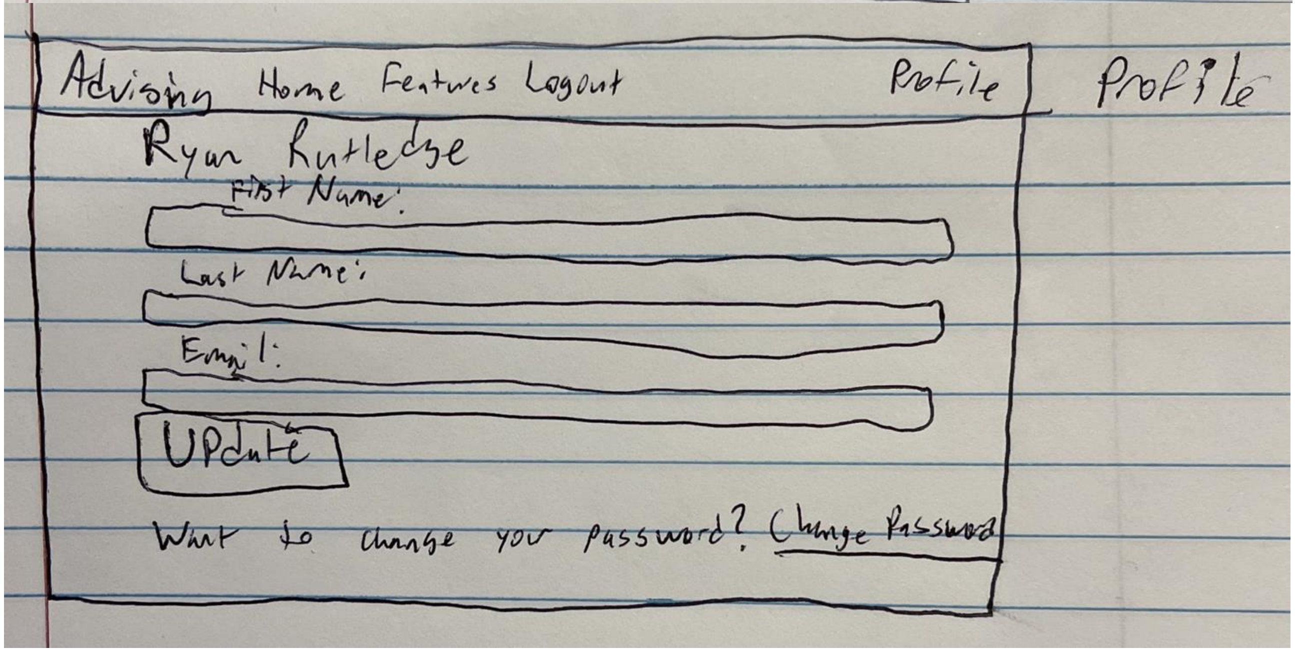
System must alert user of failed log in attempt

System must give proper features based on role

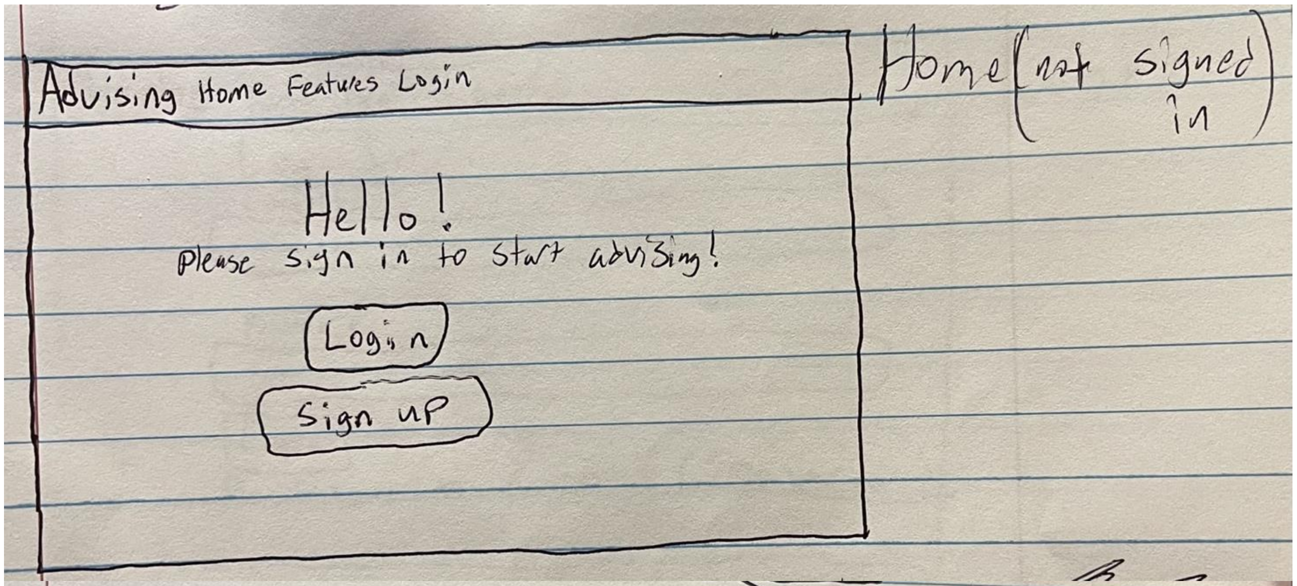

Advising
Home
Features
Login
Login
Log In
Email:
Password:
Login
Don't have an account? Create your account

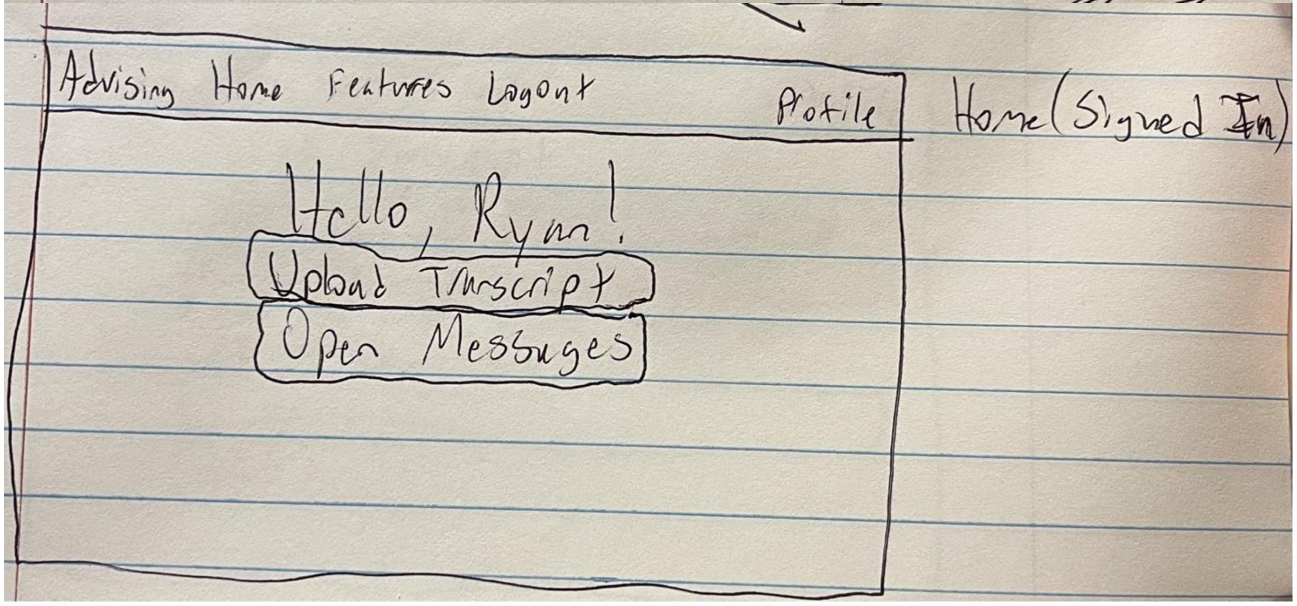

Advising
Home
Features
Login
Sign Up
(Create an Account)
Create Your Account
First Name:
Last Name:
Email:
Password:
Create Account
Already have an account? Log In

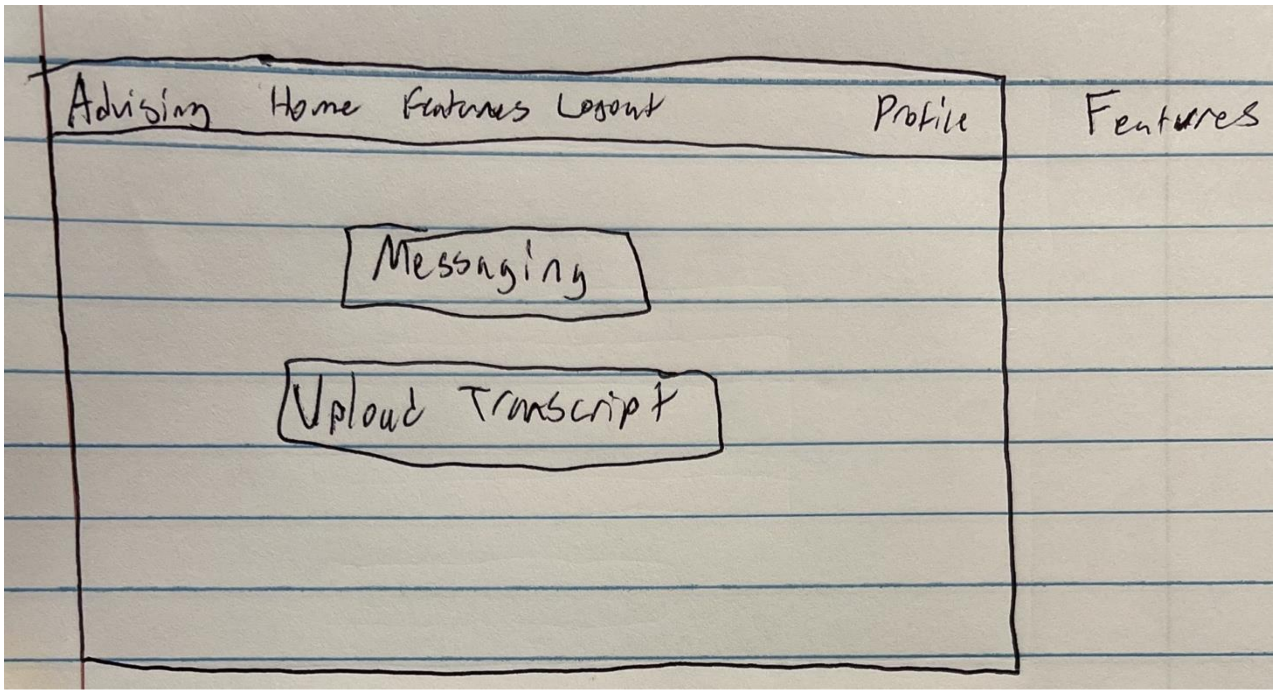

Advising Home Features Logout Profile
Change Password
New Password:
Confirm New Password:
Update Password
Update Password (must be logged in)


Advising Home Features Logout Profile
Ryan Rutledge
First Name:
Last Name:
Email:
Update
Want to change your password? Change Password
Profile

<u>Extra Wireframes for Website</u>


Advising Home Features Login
Hello!
Please sign in to start advising!
Login
Sign up
Home (not signed in)

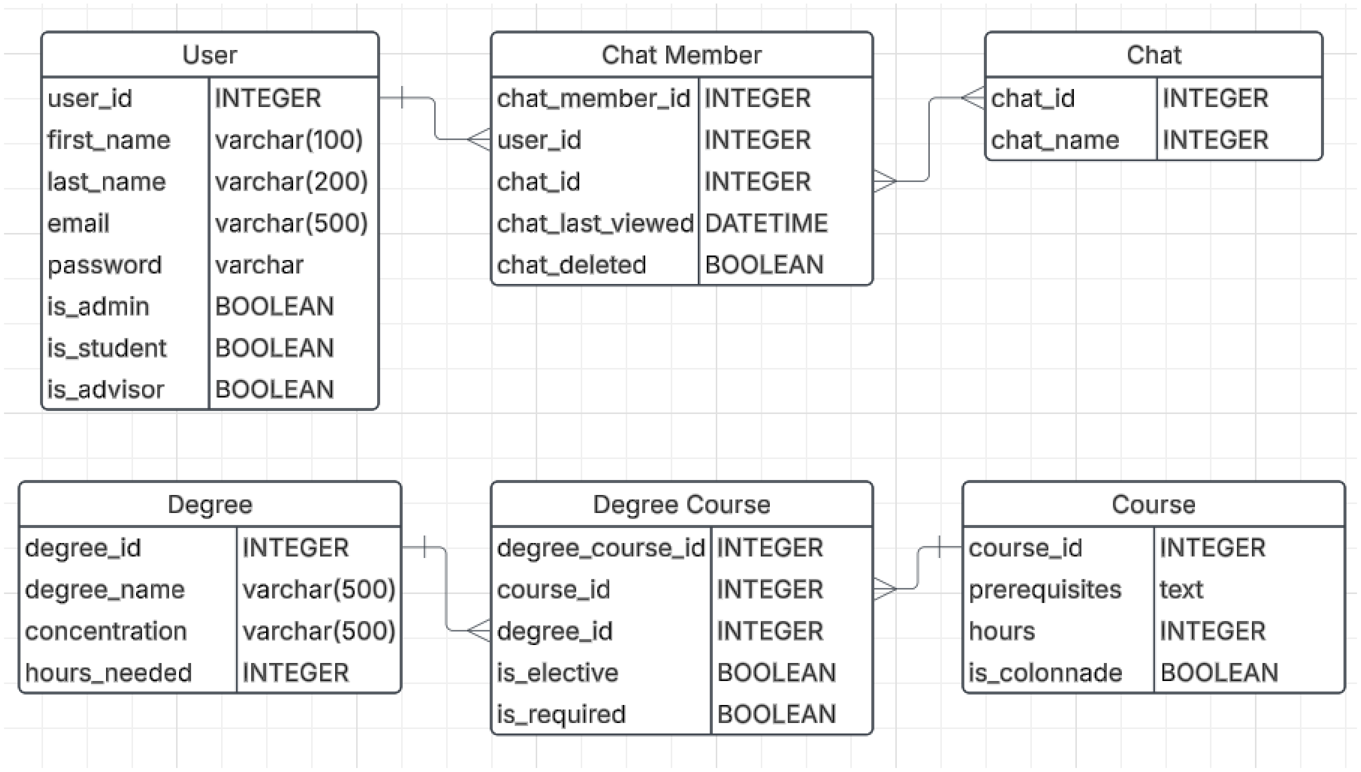

Advising Home Features Logout
Profile
Hello, Ryan!
Upload Transcript
Open Messages
Home (Signed In)



4. Data Requirements

<This section describes various aspects of the data that the system will consume as inputs, process in some fashion, or create as outputs.>

Logical Data Model



Data Dictionary

<The data dictionary defines the composition of data structures and the meaning, data type, length, format, and allowed values for the data elements that make up those structures. In many cases, you are better off storing the data dictionary as a separate artifact, rather than embedding it in the middle of an SRS. That also increases its reusability potential in other projects.>

Tables

User: Used for storing information about users and for RBAC
Chat Member: A User will have an instance of Chat Member in every chat that the User is in, and it is used to store information about when the chat was last viewed and if the chat was deleted.
Chat: Used to store the chat_id and chat_name.
Degree: Used to store information about degrees.
Degree Course: Used to store information about courses that are either electives or required for certain degrees.
Course: Used to store information on courses

Data Types

user id: autogenerated number used as the primary key for a User.
first_name and **last_name**: is for basic information that will be displayed on certain webpages like the profile and the chat pages.
email: used to login to the website as a form of username.
password: a password that will be hashed inside the database and is necessary to login.

is_admin, **is_student**, and **is_advisor**: all part of the RBAC; student is the basic account that is created, and schools would need to activate the advisor and admin role.
chat_id: autogenerated number used as the primary key for a Chat.
chat_name: the name/brief description of a chat.
chat_member_id: autogenerated number used as the primary key for a Chat Member.
chat_last_viewed: Used to store when a chat was last viewed by a User
chat_deleted: Used to tell when a chat has been deleted by a User
degree_id: autogenerated number used as the primary key for a Degree.
degree_name: the name of the degree (ex: Computer Science, Electrical Engineering, Sports Psychology, etc).
concentration: the specific concentration of the degree (ex: Computer Science Scientific Application (CSSA) or Computer Science General Option).
hours_needed: minimum hours needed to complete the degree.
degree_course_id: autogenerated number used as the primary key for a Degree Course
is_elective: Boolean used to determine whether a course is an elective or not
is_required Boolean used to determine whether a course is a required course or not
course_id: autogenerated number used as the primary key for a Course.
prerequisites: prerequisite courses needed before taking the current course.
hours: the number of hours a course is worth.
is_colonnade: used to tell if a course is a required colonnade course or not.

Reports

<If your application will generate any reports, identify them here and describe their characteristics. If a report must conform to a specific predefined layout you can specify that here as a constraint, perhaps with an example. Otherwise, focus on the logical descriptions of the report content, sort sequence, totaling levels, and so forth, deferring the detailed report layout to the design stage.>

Our application can generate a recommended schedule. The recommended schedule will be shown when an unofficial transcript is submitted. Users will have the option to download said recommended schedule.

Data Acquisition, Integrity, Retention, and Disposal

<If relevant, describe how data is acquired and maintained. State any requirements regarding the need to protect the integrity of the system's data. Identify any specific techniques that are necessary, such as backups, checkpointing, mirroring, or data accuracy verification. State policies the system must enforce for either retaining or disposing of data, including temporary data, metadata, residual data (such as deleted records), cached data, local copies, archives, and interim backups.>

Data will be acquired by using a parsing tool that will pull data from WKU's official website to fill out the database for courses, degrees, and colonnade requirements. Data about users will be generated and inputted by the users, and private chats will be created by users and the data from those chats will be stored in the database. Tables will point to a reference other tables with foreign keys to ensure that there are no null reference errors. We will be using techniques to prevent SQL injection to ensure that there are no data breaches and will need to sanitize inputs from users when chatting to prevent any data breaches.

5. External Interface Requirements

<This section provides information to ensure that the system will communicate properly with users and with external hardware or software elements.>

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.>

Our website will provide a navigation bar where users will be able to easily navigate between different screens. There will be a login/register page, a page for submitting your unofficial transcript, a screen for viewing the suggested courses, and a page for the messaging system. There will be a consistent design across the site along with error messages for invalid inputs such as a wrong input file or wrong password.

Software Interfaces

<Describe the connections between this product and other software components (identified by name and version), including other applications, databases, operating systems, tools, libraries, websites, and integrated commercial components. State the purpose, formats, and contents of the messages, data, and control values exchanged between the software components. Specify the mappings of input and output data between the systems and any translations that need to be made for the data to get from one system to the other. Describe the services needed by or from external software components and the nature of the intercomponent communications. Identify data that will be exchanged between or shared across software components. Specify nonfunctional requirements affecting the interface, such as service levels for responses times and frequencies, or security controls and restrictions.>

We will be implementing Bootstrap into the website to help with formatting and customizing the HTML, CSS, and JavaScript, we will have a database to store data for courses, users, and degrees, we will be using a parser to parse websites turned into pdfs to get the data on courses and degrees, and a website to pdf converter to help the parser.

Hardware Interfaces

<Describe the characteristics of each interface between the software and hardware (if any) components of the system. This description might include the supported device types, the data and control interactions between the software and the hardware, and the communication protocols to be used. List the inputs and outputs, their formats, their valid values or ranges, and any timing issues developers need to be aware of. If this information is extensive, consider creating a separate interface specification document.>

This software is developed to run on standard devices such as a laptop or desktop. It is compatible and will need input from the users' keyboard and mouse/touchpad but also from their internal storage where they have their unofficial transcript stored. Students will also be able to download their recommended schedule to their internal storage as well. Our software will also use Apache Web Server.

Communications Interfaces

<State the requirements for any communication functions the product will use, including e-mail, Web browser, network protocols, and electronic forms. Define any pertinent message formatting. Specify communication security or encryption issues, data transfer rates, handshaking, and synchronization mechanisms. State any constraints around these interfaces, such as whether e-mail attachments are acceptable or not.>

This system will facilitate communication between students and advisors through a built-in secure messaging system. We will ensure smooth interactions between course recommendations and academic advising. Messages will be stored in the database and be retrievable by both the sender and recipient. Lastly, the messaging system will support basic text formatting; file attachments will not be accepted.

6. Quality Attributes

Usability

<Specify any requirements regarding characteristics that will make the software appear to be "user-friendly." Usability encompasses ease of use, ease of learning; memorability; error avoidance, handling, and recovery; efficiency of interactions; accessibility; and ergonomics. Sometimes these can conflict with each other, as with ease of use over ease of learning. Indicate any user interface design standards or guidelines to which the application must conform.>

The ability to login will be available as soon as you first open the website to make access to the website's full functionality as easy as possible. From there, there will be a navigation bar that will have all necessary shortcuts to other locations such as the messaging system, the home menu, the logout button, and the auto advising menu that will be visible on all pages to make navigating the website simple and consistent.

Performance

<State specific performance requirements for various system operations. If different functional requirements or features have different performance requirements, it's appropriate to specify those performance goals right with the corresponding functional requirements, rather than collecting them in this section.>

- The system should provide a response time of 10 seconds or less for all general user actions such as uploading a transcript and downloading it
- The system should be able to handle simultaneous requests from users without performance degradation

Security

<Specify any requirements regarding security or privacy issues that restrict access to or use of the product. These could refer to physical, data, or software security. Security requirements often originate in business rules, so identify any security or privacy policies or regulations to which the product must conform. If these are documented in a business rules repository, just refer to them.>

- Uploaded transcripts should be secure and deleted immediately after usage
- User data should be encrypted
- Passwords should be hashed
- Role-based access control for users and admins

Safety

<Specify requirements that are concerned with possible loss, damage, or harm that could result from use of the product. Define any safeguards or actions that must be taken, as well as potentially dangerous actions that must be prevented. Identify any safety certifications, policies, or regulations to which the product must conform.>
Possible damage from our product would be a security breach in our database and data corruption. We will implement security measures and sanitation of inputs to prevent any unauthorized access to information by users, and in the event of the database data corrupting, we can wipe the database and pull data from WKU's website which will work as our backup data if we need it.

[Others as relevant]

<Create a separate section in the SRS for each additional product quality attribute to describe characteristics that will be important to either customers or developers. Possibilities include availability, efficiency, installability, integrity, interoperability, modifiability, portability, reliability, reusability, robustness, scalability, and verifiability. Write these to be specific, quantitative, and verifiable. Clarify the relative priorities for various attributes, such as security over performance.>

7. Internationalization and Localization Requirements

<Internationalization and localization requirements ensure that the product will be suitable for use in nations, cultures, and geographic locations other than those in which it was created. Such requirements might address differences in: currency; formatting of dates, numbers, addresses, and telephone numbers; language, including national spelling conventions within the same language (such as American versus British English), symbols used, and character sets; given name and family name order; time zones; international regulations and laws; cultural and political issues; paper sizes used; weights and measures; electrical voltages and plug shapes; and many others.>

This website is strictly intended for those who attend or work at Western Kentucky University. Internalization and localization are minimal, as the website will be developed entirely in English. There will not be a feature to change the language. The dates and times will follow U.S. formatting (e.g., MM/DD/YYYY), (e.g., 10 AM CST). All academic information will follow the university's requirements and policies. The messaging system will support standard English text input.

8. Other Requirements

<Examples are: legal, regulatory or financial compliance, and standards requirements; requirements for product installation, configuration, startup, and shutdown; and logging, monitoring and audit trail requirements. Instead of just combining these all under "Other," add any new sections to the template that are pertinent to your project. Omit this section if all your requirements are accommodated in other sections. >

Requirements by Dr. Xing

1) User Authentication and Authorization
2) Database Integration
3) Data Complexity
4) Real-Time Features
5) File Upload and Management
6) Security and Data Protection
7) Scalability Considerations
8) Frontend-Backend Separation
9) Advanced User Interface
10) Data Visualization
11) AI/ML Integration (Recommendation System)
12) Enough Sophistication

Appendix A: Glossary

<Define any specialized terms that a reader needs to know to understand the SRS, including acronyms and abbreviations. Spell out each acronym and provide its definition. Consider building a reusable enterprise-level glossary that spans multiple projects and incorporating by reference any terms that pertain to this project.>

Appendix B: Analysis Models

<This optional section includes or points to pertinent analysis models such as data flow diagrams, feature trees, state-transition diagrams, or entity-relationship diagrams. You might prefer to insert certain models into the relevant sections of the specification instead of collecting them at the end.>