Software Requirements Specification

for

CS Advising Website

Version 1.0 approved

Prepared by Team Name: Undecided

Central Washington University

11/20/2017

Table of Contents

1. Introduction

1.1 Purpose

This document is intended as the detailed description of the CS Student Advising Website to be designed for clients, Dr. Vajda and Dr. Davendra. 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 changes and problems.

1.2 Intended Audience and Reading Suggestions

This document is intended for the stakeholder, the client and the developers of the system.

1.3 Product Scope

The objective for this application is to replace the current advising tool used to manually generate graduation plans for Computer Science students at Central Washington University by implementing data contained in the CWU course catalog. This application will be displayed in a website UI where the user will be able to easily create different scenarios for the student based on multiple constraints that will, ultimately, supply the user with the shortest graduation path.

1.4 References

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

2. Overall Description

2.1 Product Perspective

This product is being designed to replace the old advising tools which are currently in use by the CS department. It will be a standalone website that does not tie into any larger or separate set of software, other than the database and web server software that will host the system. The data on which the application acts - Catalog, schedule, and student records - may be hosted either on the systems internal database or accessed from an external database.

2.2 Product Functions

The functions of the product include:

1) A website which provides a UI for the users. This UI will contain two sub-sections:

a) Administration: This will allow the Administrators to add courses, modify existing courses, or modify data contained within the databases.

b) Advising: This will allow the Advisor to generate graduation plans for a selected student, the advisor will be able to input constraints here, such as min/max credits per quarter, min/max number of courses per quarter, enable/disable summer quarter, and enable/disable transfer student checking.

2) Several databases (number and kind to be determined in a future document), which will store necessary information for this tool.

3) An application which will be run from within the website, take the constraints input from the website, and pull any necessary data from the databases to generate the graduation plan, which will then be returned to the website.

2.3 User Classes and Characteristics

This product will be solely used by advisors and appointed administrators within the Computer Science program and Central Washington University. Advisors will have a separate tab from administrators. The advisors tab will be able to operate without technical knowledge. Administrators will have special access to alter the databases, which will be done occasionally.

2.4 Operating Environment

This product will operate, and be stored on a Linux-based server. The resource use of this software is not known at this point, but it is expected to require at least 2 Gb of memory and a modern CPU. The website UI component of the product will be designed for Mozilla Firefox 57.0.x or later, but is expected to run on other browsers, though it will be built for use with Firefox. The application component will be designed exclusively for Linux OS (exact type TBD), and will not be expected to run on other platforms. The database type that will be used is TBD.

2.5 Design and Implementation Constraints

Due to the sensitive nature of the data that is being handled by this product, no testing with real data will be possible. Federal laws prohibit the disclosure of student records to third parties; thus, the developers will not be able to access them during development. Instead, the client must provide the developers with the exact format these records are stored in, such that dummy records can be generated for testing.

The Internet connection is another constraint for the website. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function. The web portal will be constrained by the capacity of the database.

Another constraint is that software must support 8-9 simultaneous users without crashing. As a result, it may be forced to queue incoming requests and therefor increase the time it takes to fetch data. The best time efficiency to generate the graduation plan might not be that fast. Also, since the catalog might change every year, the system must flexible to deal with changes.

2.6 User Documentation

The product will be delivered with an in-depth user manual for both Administrator and Advisor users. This manual will describe how to use each component in the web-interface, as well as the function and purpose of each component. Furthermore, multiple demonstrations of the product will be held for the client during the development stage, where they will be instructed on how to use the software, and where user feedback can be given for improvements. A demonstration of the final product will also be held for the client with the final release.

Additionally, the product will be delivered with documentation for the technical side of this product for the Administrators. This documentation will describe the inner workings of the software, expected in/outputs, structure, and in-depth descriptions of each software module within the product.

2.7 Assumptions and Dependencies

One dependency is the Course Catalog and Student Records, so data will be provided by the school, and the information stored within this programs database should be able to be updated with minimal data conversion. Therefore, the database schema should be as close as possible to the existing Catalog and Student Record databases at Central Washington University.

Another dependency is that the product will not be used by more than 10 people at a time.

3. External Interface Requirements

3.1 User Interfaces

The website will be the only user interface provided by this product, all user interaction will go through this interface. The website will have:

- A Login page, which has two text fields, <Username> and <Password>, as well as one button, <Login>.
  - The two text fields will receive the user's credentials for login.
  - The Login button will call a function to verify the user's credentials, and, if successful, redirect the user to their respective homepage as described below.
    - Unsuccessful login attempts will display an error message to the user, which will prompt them to check their username, and type their password in once more.

- An Administrator Homepage, which the administrator users are redirected to after a successful login attempt. This page will have three buttons, <Logout>, <Administrator Tools>, and <Advising Home>.
  - The Logout button ends the user's session, and terminates the connection with the server, sending the user to the logout page.
  - The Administrator Tools button will redirect the user to the Administrator Tools page.
  - The Advising Home button will redirect the user to the Advising Homepage.

- An Advising Homepage, which Advisor users are automatically redirected to after a successful login attempt. This website will display the main interface for advising, which will allow them to create the shortest graduation path or generate needed documents. Here, there will be a field <Student ID> that will allow the advisor to find the student and display their major progress and another tab <Documents> where the advisor can find all the documents displayed. Advisors will be able to create a shortest graduation path after the student is displayed by clicking a <Generate Graduation Path>. The display will also be a <Logout> button, which serves the same purposes as described in Administrator Home.

- An Administrator Tools page, which only Administrators can access. This page will display all tools available to the administrator, which are to be determined. There will also be a <Logout> button, which serves the same purposes as described in Administrator Home.

- A Logout page, which all users will be redirected to after pressing the <Logout> button on any page. This page will only display a message to the user that they were successfully logged out, and may close the browser window.

All individual pages within the website will follow a common style. They will all display the Department of Computer Science logo, and will all have the standard layout as described by the school.

No keyboard shortcuts will be implemented within the website.
No help functionality will be implemented within the website. Users should refer to the user-manual.
The user interface is specifically design for interacting with the product, not for altering it. The only alterations that can be made via the user interface is the Administrator's ability to alter the databases through the Administrator's tools page.
All pages will be designed to be user friendly, and user testing will be conducted to ensure this website is usable by Advisors, and Administrators

3.2 Hardware Interfaces

The product will only support computers capable of running Linux (exact type TBD) in desktop mode, tablets, smartphones, etc. will not be supported.
The product will use Linux API functions to communicate with hardware components as needed.
The product does not require any special hardware interfaces, or components.

3.3 Software Interfaces

The website of this product will use:

- A browser, which is expected to be Mozilla Firefox 56.0.x or later.
- Databases. All data is expected to be stored in the accompanying databases. The website will access the databases via SQL queries.
- The accompanying application. The website will actively engage with the application for plan generation, and access to the databases. The website will use the above components to verify input, produce output, and retrieve information.
- The user will input credentials to the login page, which will then attempt to retrieve the credentials from the appropriate database for verification.
- The user will click buttons to navigate throughout the website, these inputs will be handled by the website as described in section 3.1 User Interfaces.
- The user will provide input via text boxes, radio buttons, check boxes, and drop-down menus to provide constraints for the application. Upon submission of this input, the application will verify it, and produce the selected output, which will be sent back to the website for display to the user.
- All inputs from the user that are intended to edit databases will be turned into SQL queries, and sent to the selected database for execution. The input will be validated by the website for structure, not content.

The application accompanying this product will

- Interact with the Linux API, the databases, and the website (as described above). Thus, it will require the Linux API libraries to be present on the system.
- Automatically generate SQL queries to the database as needed for the task it is performing.
- Not interact with the operating system beyond what is required to run the application. All events, and tasks will be handled by the software without the operating system.
- Use the network interface to communicate with the databases if they are stored on a separate machine. Thus, in this case, a network connection to the database will be required for the product to function. The Linux API will be used to establish this connection.

All database interactions will be performed using SQL queries. The database will receive these queries, execute them, and return the result to the calling agent via the established connection.

The website and application will share the information received from the databases. Information that the website received from the database will be sent to the application as needed for processing, and information the application retrieves from databases during the processing stage will be sent to the website.

3.4 Communications Interfaces

This product will utilize the HTTPS for network communications between components. Thus, all information sent by the product over the network will be encrypted using the TLS protocol.
Communications between the application and databases will use a SQL connection, as defined within the Linux API.
Communications between the application and the website will depend on where the application will reside. If the application is on a server, it will use HTTPS communications. If the application is on the user's machine, it will use the Linux API to communicate with the website.
All emails generated from within the website will be sent via Outlook, and will be transferred to Outlook using the HTTPS.

4. System Requirements

These are the system requirements that were gathered from client meetings, which will be used to guide the implementation of this product.

4.1 User Friendly Website

The website must be easy to use for the advisors. It should be simple, non-cluttered, and only contain the features requested, without any extra features.

4.1.1 Login Page – Priority: 8
The user should be presented a login page where the user can enter their credentials.

4.1.2 Advising Page – Priority 10
The main page for Advisors, which provides them with the advising tools.

4.1.2.1 Generate Graduation Plan
A button which the Advisor can press which will generate, and display a new graduation plan for the selected student.

4.1.2.2 Drop-down menu for students
A drop-down menu which contains all CS students, from which the advisor can select a student to view the student's graduation plan, and records.

4.1.2.3 Access to student transcript
There should be a way for the Advisor to view the transcript through the website.

4.1.2.4 Ability to alter graduation plan manually
The advisor should be able to alter the graduation plan of a student manually

4.1.2.5 Menu for plan constraints
The advisor should have a menu which allows them to alter the constraints that will be applied to the generation of the student's graduation plan.

4.1.2.6 Navigation bar
The advisor should have a navigation at the top of the website where they can navigate between different pages.

4.1.2.7 Advising Forms
The advisor should be able to automatically generate Advising forms, and have them populated with the student's data.

4.1.2.8 Summer Quarter check box
The advisor should be able to enable/disable the option for the plan to schedule classes for summer quarter.

4.1.3 Administrator Page – Priority 7

There should be an administrative page that allows the Administrators to alter the databases to add, remove, or alter classes, and other contents of the databases.

4.2 Databases

The data used by the product should be stored within databases. These databases should be designed, and populated by the developers.

4.2.1 Courses Database – Priority: 10

A database containing all courses applicable to the CS major, such as, CS-prefixed courses, and acceptable electives from other departments.

4.2.2 Plan Database – Priority: 10

A database containing all graduation plans previously created by the application, and where all new ones will be stored.

4.2.3 Catalog Requirement Database – Priority: 10

A database containing the graduation requirements of each catalog year applicable to the students in the major.

4.2.4 Student Info Database – Priority: 6

A database which contains the student's records, how these records will be updated or added is TBD.

4.3 Application

An application that interacts with both the website, and the databases to generate graduation plans.

4.3.1 Generate plan algorithm – Priority: 10

An algorithm which generates a graduation plan for a given student based on the given constraints in a short period of time (one-minute max). The plan should be based off a skeleton with all core courses in it.

Plan must satisfy these conditions: Load balanced, prerequisites considered, credit requirement is met after completion, all core classes are in plan, 5 electives are in plan, all electives are valid, student is ready to graduate by end of last quarter in plan.

4.3.1.1 Verify user input

All user input must be verified for correctness before processing it.

4.3.1.2 Retrieve student records from database

Retrieve the student's current and previous classes from the database.

4.3.1.3 Retrieve student graduation plan from database

Retrieve the old graduation plan (if applicable) stored for this student so that it can be modified.

4.3.1.4 Generate new plans

Modify old, or create new graduation plans for this student, based on the information retrieved from the databases, and the constraints entered by the advisor.

4.3.1.5 Return plans to website

All generated plans must be returned to the website for display to the user

4.3.2 Save Plan – Priority: 2

Once the user has decided which plan to save, store this plan in the database.

4.3.3 Create and populate forms – Priority: 1

The application should create and populate forms for a student, and allow the advisor to download or print a PDF file of these forms.

4.4 Feasibility Check

4.4.1 User friendly Website

All requirements pertaining to the website are perfectly feasible. The most important step to ensuring this requirement is met is the user testing. If the website is tested extensively, and the user feedback is incorporated into each iteration of the website, there will be no issues making this website.

4.4.2 Databases

The first three databases (4.2.1 – 4.2.3) should not have issues from a feasibility point of view; however, the last database which contains the student information (4.2.4) may present issues. With little knowledge about how the University stores student information, and which information we will

be provided, it will be difficult to design a database to store it. If the Client provides the required details about the information this database will store, this should not present a major problem, but if not, it may not be possible to implement this feature properly which will negatively affect the entire project.

4.4.3 Application

All requirements concerning the Application are mostly feasible. Until more is known about the environment it will run in, we cannot draw conclusions about whether the algorithmic task is feasible; however, as we are expecting to have an optimal environment for this application to run in, there should be no issues. The only problem that may affect the feasibility of this task is the fact that the Client has not provided a very detailed description of what the information passed to the algorithm looks like. Until we learn more about the exact structure of all information being processed by the algorithm, it is difficult to say whether the implementation will be able to do exactly what the Client requested. Furthermore, the information on exactly what the plan must contain to be correct is still not clear. The Client should provide an example of a correct plan, as well as a few examples of an incorrect plan to ensure no errors are made during the implementation.

4.5 Consistency Check

4.5.1 User friendly Website

The current requirements for the website have no inconsistencies and do not contradict each other.

4.5.2 Databases

The only inconsistency with the database requirements that is currently present is the fact that we do not enough about the information the Catalog Requirements (4.2.3) and the Student Info (4.2.4) databases will contain. Though the information does not contradict any other information, this will certainly cause issues during the implementation if the vague requirements for these two databases aren't clarified and updated.

4.5.3 Application

The application contains two inconsistencies from what the Client has told us, namely, the plan should (not) contain the basic skills and general education requirements, and the application should create multiple plans, but also create a skeleton that is filled with electives.

- • We have received conflicting information during two meetings where we have been instructed to put basic/general education classes into the plan, put only dummies into the plan (i.e. GE1, BS1, etc.), and to leave it out of the plan. This must be cleared up for the implementation to work correctly.

- • Since the student gets to pick which electives s/he wants to take, we can only suggest when to take electives, not which ones. Thus, if we create a skeleton with all core classes in it, and then fill it with electives, we will always end up with the same plan, not multiple plans. This needs to be cleared up before the algorithm can be implemented to prevent an incorrect result.

4.6 Validity Check

4.5.1 User friendly Website

The current requirements for the website are all related to the goal. Furthermore, all requested features (Section 5.1, 5.7, and 5.8) can be traced to at least one requirement. The only questionable item mentioned is the auto-population of advising forms, though this is a low priority item, it does not necessarily relate to the goal of creating an advising tool. But it would make the job of the Advisor slightly more efficient, so it should be considered valid.

4.5.2 Databases

All database requirements are valid, and can be traced to multiple requirements. The features (Section 5.2-5) requested for the databases are essential to the product, thus, these are perfectly valid.

4.5.3 Application

The requirements concerning the application, and the features (Section 5.1-5.7) related to it are all valid, and related to the goal of the product.

5. System Features

5.1 Website UI

5.1.1 Description and Priority – Priority: 10

The website will be the UI of this product. The user will interact with the product components through the website, provide all inputs through it, and all outputs will be displayed to the user through this UI.

5.1.2 Stimulus/Response Sequences

The UI will have multiple different pages (as described in Section 3.1 User Interfaces), which allow the user to do multiple things, namely: alter databases, create graduation plan for a student, retrieve a graduation plan for a student, view/populate advising forms, take advising notes, and view student transcript.

5.1.3 Requirements related to this feature:

Requirement 4.1

5.2 Graduation Plan Database

5.2.1 Description and Priority – Priority: 10

This database will store the graduation plans that have been generated for students.

5.3 Courses Database

5.3.1 Description and Priority – Priority: 10

This database will store all valid courses that may be put into the graduation plan.

5.3.2 Stimulus/Response Sequences

The application will retrieve this information when generating plans.

5.3.3 Requirements related to this feature:

Requirement 4.1.2.1
Requirement 4.1.2.4
Requirement 4.2.1
Requirement 4.3.1.4
Requirement 4.3.2

5.4 Catalog Requirements Database

5.4.1 Description and Priority – Priority: 8

This database will store graduation requirements for each catalog year.

5.4.2 Stimulus/Response Sequences

The application will retrieve this information when generating plans.

5.4.3 Requirements related to this feature:

Requirement 4.1.2.1
Requirement 4.1.2.4
Requirement 4.2.3
Requirement 4.3.1.4
Requirement 4.3.3

5.5 Student Information Database

5.5.1 Description and Priority – Priority: 4

This database will store student records, such as classes taken, classes enrolled in, catalog year, etc.

5.5.2 Stimulus/Response Sequences

The application will retrieve this information when generating plans, and filling forms.

5.5.3 Requirements related to this feature:

Requirement 4.1.2
Requirement 4.2.4
Requirement 4.3.1.2
Requirement 4.3.3

5.6 Plan Generation Algorithm

5.4.1 Description and Priority – Priority: 10

This algorithm will generate a number of possible graduation plans the student could use.

5.4.2 Stimulus/Response Sequences

The application will use this algorithm to create new, or alter old, graduation plans. The algorithm will receive a list of classes the student has taken. The algorithm will then progressively add classes to this list, from available classes (based on prerequisite completion). The plan generated will be optimized for fewest quarters, based on the constraints that were entered by the Advisor (e.g. load max, summer quarter, etc.).

5.4.3 Requirements related to this feature:

Requirement 4.1.2.1
Requirement 4.3.1

5.7 Populate Forms

5.4.1 Description and Priority – Priority: 1

This feature will create a PDF form of specified type with student information pre-filled.

5.4.2 Stimulus/Response Sequences

The website will call this function when the user selects a form to generate. The function will generate the form, pre-filling all available information, and then return the PDF file.

5.4.3 Requirements related to this feature:

Requirement 4.1.2.1
Requirement 4.3.1

5.8 Administrative Tools

5.4.1 Description and Priority – Priority: 1

This feature will allow Administrator users to alter Databases via the website, create new users, and edit existing users. This will be the main tool for maintaining the system after release.

5.4.2 Stimulus/Response Sequences

User will input data into a form on the website, using the input, SQL queries will be generated, and sent to the respective database that was selected. The database will either return nothing, or the requested data.

5.4.3 Requirements related to this feature:

Requirement 4.1.3

6. Other Nonfunctional Requirements

6.1 Performance Requirements

The algorithm (Feature 5.6) must be able to finish generating a plan in a reasonable amount of time (no more than one minute). This is to prevent long wait time while the advisor is generating the plan.
The databases, and website must be able to handle ten simultaneous users without any loss in performance; i.e. the user experience should be identical for one and ten simultaneous users.

6.2 Security Requirements

There should be login credentials for all users that prevent unauthorized users from logging in to the website. Administrators should be the only ones capable of creating or giving permissions to users.
Databases storing student information must also be protected by passwords and encryption.

6.3 Software Quality Attributes

The algorithm (Feature 5.6) must generate correct graduation plans. A correct plan is defined as having the following features:

- All prerequisites are met before a student takes a class, unless the Advisor manually changes it.
- International students have at least 5 credits per quarter.
- No student has more than 18 credits in a single quarter, unless the Advisor manually changes it.

•The credit requirement (as defined by the catalog year's graduation requirements) has been met by the last quarter in the plan.

• The elective requirement (as defined by the CS department) has been met by the last quarter in the plan.

• All electives in the plan are approved by the CS department.

• All core classes (as defined by the CS department) are in the plan.

The UI must be easy to use for all users, including non-computer scientists. This should be tested to verify that users have no issues navigating, or understanding the website.
There should not be any features in the UI that were not explicitly requested by the client.

6.4 Business Rules

Administrators should have access to all features on the website, and should have total control over the three databases (Features 5.2-4). Only Administrators should be able to access the Administrator Tools. Advisors should not have access to Administrator tools under any circumstances.

7. Appendix A: Glossary

Acronyms:

CPU	–	Central Processing Unit
CS	–	Computer Science
CWU	–	Central Washington University
e.g.	–	for example
etc.	–	etcetera
Gb	–	Gigabyte
HTTPS	–	Secure Hypertext Transfer Protocol
i.e.	–	in other words
SQL	–	Standard Query Language
TBD	–	to be determined
TLS	–	Transport Layer Security
UI	–	User Interface

Terms:

The Algorithm	–	Feature 5.6, which generates the graduation plans.
The Application	–	The software component of this product that generates graduation plans.
The Client	–	Drs. Vajda and Davendra.
Credentials	–	A unique username and password pair used to login to this product.
The Developers	–	The team building this product.
Modern CPU	–	A CPU released within the last five years.
The product	–	The CS advising website this SRS is for.
The users	–	All CS department advisors, and the system administrator