**Reviewing Group No.: 1**

**Reviewed Group No.: 8**

**System Requirement Specification**

Group 1 – Korey Diaz, Owen Ehlen, Alex Kollmann, Jarid Rosanske, and Ryan Easterday

**1.**

**Purpose**

Each semester, students are faced with the burden of finding out which classes they must take in order to fulfill their graduation requirements for their intended major. Occasionally, a student will incorrectly plan out what courses they must take which results in them not being able to graduate on time. Our software would be able to guide a student into correctly planning the courses they need to take over their four years here at the University of Wisconsin-Platteville.

**2.**

**Scope**

Specifically, this software is directed towards students and staff at the University of Wisconsin-Platteville. The Four-Year Student Plan shall provide a four-year plan consisting of classes that a student must take based on their specified major. The software’s primary goal is uncomplicating the process of schedule building for both students and staff. This Software Requirements Specification (SRS) was created in order to assess and identify the requirements of the Four-Year Student Plan.

**3.**

**Definitions**

UWP – University of Wisconsin-Platteville

SRS – Software Requirements Specification

GUI – Graphic User Interface

**4.**

**Overview of Document**

The remaining sections of this SRS consist of a general description, specific functional and non-functional requirements, and references. Section 5 withholds a background for the requirements in the section titled specific requirements. Section 6 contains detailed software requirements needed for the application. Section 7 provides a list of documents that were used to create the SRS.

**5.**

**General Description**

The Four-Year Student Plan shall allow for user input of their major. The program shall then print to the user a list of required courses. The user shall be able to interact with each course by dragging them into a specific semester and year. This shall allow for easy movement of classes to test different schedules and find out what is compatible. The product shall be a four-year schedule, decided by the user, which shall allow the student to oversee the classes they will take.

**6.**

**Specific Requirements**

**6.1** **Functional Requirements**

**6.1.1 Four-Year Student Plan**

6.1.1.1 The program shall print a list containing every offered major at UWP.

6.1.1.2 The majors printed on the list shall be organized in alphabetical order.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | Maybe say something like “the majors stored in the list” not “printed on the list” |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.1.1.3 The user shall be able to choose their major from the list.

6.1.1.4 The user shall be able to have their schedule created automatically using the integrated algorithm by selecting the “Automatic” option.

6.1.1.5 The user shall be able to manually create their schedule by selecting the “Manual” option.

6.1.1.6 The program shall have a confirm button for users to click once they have chosen their major.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | Reword or move to non-functional requirements |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

**6.1.2 Schedule Editing**

6.1.2.1 The program shall print a list of required courses based on the major titled “Required Courses.”

6.1.2.2 The program shall have a button titled “Add.”

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | GUI elements should be non-functional requirements |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.1.2.3 The “Add” button shall add a course to either the “Fall Term” or “Spring Term.”

6.1.2.4 The program shall have a button titled “Remove.”

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | This requirement should be a non-functional requirement unless function of the remove button is added |
| Complete | N | Requirement should have function of the button |
| Concise | Y | It is at its smallest unit |
| Consistent | Y | Only document we know so it has to be consistent |
| Specific | Y | The requirement is concise so it is specific as well |
| Measurable | Y | Easy to see if program has a button called remove |
| Attainable | Y | Simple to make in any GUI program |
| Realizable | Y | Possible to make |
| Traceable | Y | Easy to see if the remove button is made |

6.1.2.5 The “Remove” button shall remove a selected course from either the “Fall Term” or “Spring Term.”

6.1.2.6 The program shall print a list that decides which semester the classes shall be added to titled “Select Term.”

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y | It is a functional requirement |
| Complete | X | How does program print the list and when |
| Concise | Y | It is at its smallest unit |
| Consistent | Y | Only document we know so it has to be consistent |
| Specific | X | Should tell when the list is printed phrasing is confusing |
| Measurable | Y | If a list is printed |
| Attainable | Y | Printing list in GUI can be attained |
| Realizable | Y | Possible to make |
| Traceable | Y | Easy to see if printing list is being done |

6.1.2.7 The “Select Term” list shall contain two options: “Semester 1” and “Semester 2.”

6.1.2.8 The program shall print a list that allows users to select what year they are in titled “Select Year.”

6.1.2.9 The “Select Year” list shall contain four options: “Freshman,” “Sophomore,” “Junior”, and “Senior.”

6.1.2.10 The program shall print a drop box titled “Fall Term.”

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | This is a non-functional requirement |
| Complete | N | What does fall term box do |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | N | What does fall term box do |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.1.2.11 The program shall print a drop box titled “Spring Term.”

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | This is a non-functional requirement |
| Complete | N | What does spring term box do |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | N | What does spring term box do |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.1.2.12 The courses in the terms shall be able to be moved by the user.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | N | Doesn’t explain moved, moved to another term? Moved to a new location? Moved to a new time? |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

**6.1.3 Schedule Viewer**

6.1.3.1 If the user chooses an automatically generated schedule, the program shall print a window titled “Classes for Four Years.”

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | Print or display? |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.1.3.2 The program shall print a schedule of classes required for each semester.

6.1.3.3 The program shall print a schedule for each year.

6.1.3.4 The program shall have a set of buttons titled “Edit,” “Export,” and “Exit.”

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | I feel as if this requirement is GUI based, it should be non-functional. |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.1.3.5 The “Edit” button shall allow users to modify their courses by taking them to the “Schedule Editor.”

6.1.3.6 The “Export” button shall turn the generated schedule into a .pdf file.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | How does the user know the pdf is created? Is the file saved to their system? |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.1.3.7 The “Exit” button shall terminate the program.

**6.2** **Non-Functional Requirements** (Under the hood requirements, users do not interact with them.)

**6.2.1 Databases**

6.2.1.1 The majors database shall contain all available majors on the UW-Platteville campus.

6.2.1.2 The courses database shall contain all classes available on the UW-Platteville campus.

6.2.1.3 The program shall pull all of classes from a database that match the inputted major.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | N | Reword “shall put all of classes from a database”.  “All of the” ? |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.1.4 The program shall pull all the majors from the second database to select which major the user desires.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | N | Where in the program? |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.1.5 The majors database shall contain a general name for each major.

6.2.1.6 The course database shall contain an identifier for each course in the database.

6.2.1.7 The course database shall contain a general name for each course in the course database.

6.2.1.8 The course database shall contain a list of prerequisites for each course in the course database.

6.2.1.9 The course database shall contain a list of post requisites for each course in the course database.

6.2.1.10 The course database shall contain a list of corequisites for each course in the course database.

6.2.1.11 The course database shall contain a Boolean to specify if a course is a prerequisite for each course in the course database.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | Not possible to test if the course database contains a boolean |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.1.12 The course database shall contain a Boolean to specify if a course is a post requisite for each course in the course database.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | Not possible to test if the course database contains a boolean to specify if a course is a postrequisite |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.1.13 The course database shall contain a Boolean to specify if a course is a corequisite for each course in the course database.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | Not possible to test if the course database contains a boolean to specify if a course is a corequisite |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.1.14 The majors database shall contain a list of required courses for each major in the major database.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | Not possible to test if the course database contains a list of required courses |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.1.15 The majors database shall contain an identifier for each major in the major database.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | Not possible to test if the database has an identifier for each major in major database |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

**6.2.2 Four-Year Plan**

6.2.2.1 The classes shall be placed in a schedule that spans across a typical undergrad 4-year period.

6.2.2.2 Each semester shall default to a 15-credit hour schedule.

6.2.2.3 Each semester shall be at least a 12-credit hour schedule to maintain a full-time student status.

6.2.2.4 Each semester shall only be allowed 18-credit hours maximum.

6.2.2.5 Each semester’s credit hours can be changed in a range of 12-18 depending on the classes required.

6.2.2.6 The program shall check if the course is a prerequisite.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | Not testable, no way to test if the program checked. |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.2.7 The program shall check if the course has a corequisite.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | Not testable, no way to test if the program checked. |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.2.8 The program shall check if the course has a prerequisite.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | Y |  |
| Measurable | N | Not testable, no way to test if the program checked. |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

6.2.2.9 If a course has a prerequisite the program will force that course to be taken in a previous semester.

6.2.2.10 If the course has a corequisite the program shall allow the course to be taken during the same semester.

|  |  |  |
| --- | --- | --- |
| **Smart Attribute** | **Status** | **Comments** |
| Correct | Y |  |
| Complete | Y |  |
| Concise | Y |  |
| Consistent | Y |  |
| Specific | N | Same semester as? |
| Measurable | Y |  |
| Attainable | Y |  |
| Realizable | Y |  |
| Traceable | Y |  |

**7.**

**References**

**7.1** SRS Example – Atef(Located on Canvas)

**7.2** SRS Example – e-Store (Located on Canvas)

**7.3** A1-Group1-SystemRequest (Document from Assignment 1)

**Other Questions, Issues, or Missing Requirements**

Just a few small fixed and rewording of some requirements. Also, might need to add a few more functional and non-functional requirements.