*STUDENT ADVISEMENT SYSTEM (SAS)*

PROJECT #3

*Patrick, Alan, Geoff*

*9/24/2016*

REVISION HISTORY

**-**Version 1.0 Released 2016-02-25 Patrick, Geoff, AlanStudent Advisement System Project #3

*-*Revision 1.1 Released 2016-03-24 Patrick, Geoff, AlanStudent Advisement System Project #3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version Number** | **Details of Changes** | **Prepared By** | **Approved By** | **Date of Approval** |
| 1.1 | Added Revision History | Patrick | Team 3 | 3-20-16 |
| Project Scope Subsections  2.0-11.0 | Patrick | Team 3 | 3-20-16 |
| Project Feasibility Tangible & Intangible benefits | Alan | Team 3 | 3-20-16 |
| Redefined WBS Structure | Geoff | Team 3 | 3-20-16 |
| Restructured schedule as tabular form (Task ID, Description, Duration, Date) | Alan / Patrick | Team 3 | 3-20-16 |
| Assign resources to tasks | Patrick | Team 3 | 3-20-16 |
| Identified risks & prioritized risks plan | Alan | Team 3 | 3-20-16 |
| Included meeting agenda participants | Geoff / Alan / Patrick | Team 3 | 3-20-16 |
| Included meeting minutes of users & minutes | Geoff / Alan / Patrick | Team 3 | 3-20-16 |
| System Requirements Specifications | Geoff | Team 3 | 3-20-16 |
| Analysis Documents Use Case Diagram | Patrick | Team 3 | 3-20-16 |
| Team Workload Distribution | Geoff / Alan / Patrick | Team 3 | 3-20-16 |

*-*Version 2.0 Released 2016-04-14 Patrick, Geoff, AlanStudent Advisement System Project #3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version Number** | **Details of Changes** | **Prepared By** | **Approved By** | **Date of Approval** |
| 2.0 | Modified revision history into tabular form | Patrick | Team 3 | 4-12-16 |
| Project Organization Overview Goal | Patrick / Geoff / Alan | Team 3 | 4-12-16 |
| Redefined Project Requirements | Patrick / Geoff / Alan | Team 3 | 4-12-16 |
| Redefined Non-Functional Project Specifications | Patrick / Geoff / Alan | Team 3 | 4-12-16 |
| Redefined Critical assumptions | Patrick / Geoff / Alan | Team 3 | 4-12-16 |
| Redefined Project Constraints | Patrick / Geoff / Alan | Team 3 | 4-12-16 |
| Redefined Project Success Criteria | Patrick / Geoff / Alan | Team 3 | 4-12-16 |
| Included Team Skills in Technical Feasibility | Patrick / Geoff / Alan | Team 3 | 4-12-16 |
| Prioritized Risks | Alan | Team 3 | 4-12-16 |
| Re-Structured Risk Management Plans | Alan | Team 3 | 4-12-16 |
| Re-Formatted Workload Distribution | Patrick | Team 3 | 4-12-16 |
| Modified Use-Case Diagram | Patrick | Team 3 | 4-12-16 |
| Re-Defined SRS Advisement Process | Geoff | Team 3 | 4-12-16 |
| Added 3 Fully Dressed Use Cases | Patrick / Geoff | Team 3 | 4-12-16 |
| Added Domain Model | Patrick / Geoff | Team 3 | 4-12-16 |

TABLE OF CONTENTS

1. PROJECT SCOPE………………………………………………………………………….…..5

1.1 PROJECT OVERVIEW…………………………………………………………...……….5

1.2 OVERVIEW OF THE ORGANIZATION……………………………………....………...5

1.3 CURRENT SITUATION AND PROBLEM/OPPORTUNITY STATEMENT…………………………………………………………….……………………..5

1.4 STATEMENT OF PURPOSE……………………….………………………………….….6

1.5 PROJECT OBJECTIVES………………………………………………………………….6

1.6 PROJECT REQUIREMENTS…………………………………………………………..…6

1.7 PROJECT SPECIFICATIONS…………………………………………………………...6-7

1.8 CRITICAL ASSUMPTIONS…………………………………………………………….…7

1.9 PROJECT CONSTRAINTS…………………………………………………………..…....7

1.10 PROJECT ACCEPTANCE CRITERIA…………………………………………….....…7

1.11 PROJECT SUCCESS CRITERIA……………………………………………………......8

1. 12 PROJECT DELIVERABLES………………………………………………………….…8

2. FEASIBILITY STUDY…………………………………………………………….…………....…9-11

3. WORK BREAKDOWN STRUCTURE (WBS)…………………………………………………..9-10

4. COST ESTIMATE……………………………………………………………………………………11

5. PROJECT SCHEDULE………………………………………………………….……………….12-13

6. RESOURCE ALLOCATION……………………………………………………………………..….13

7. RISK MANAGEMENT PLAN (RMP)……………………………………………………….......14-19

8. MEETING AGENDAS & USER RECORDINGS…………………………………………….....19-26

9. WORKLOAD DISTRIBUTION………………………………………………….............................26

10. USE CASE DIAGRAM……………………………………………………………………………..27

11. FULLY DRESSED USE CASES (3)…………………………………………...……………….28-30

12. DOMAIN MODEL………………………………………………………………………………….31

13. SYSTEM REQUIREMENTS SPECIFICATIONS (SRS)…………………………………….32-40

**14. DEPENDINCIES CHART………………………………………………………………………….41**

**15. ESTIMATES METHOD…………………………………………………………………………....42**

SECTION 1 PROJECT SCOPE

1.0 Project Overview

The goal for the advisement system is to create a system that will prepare student advisees for the upcoming semesters by helping decide what courses are required and updating degree track sheets in order to maintain steady progression towards graduation. This project will consist of creating a system that will allow advisers to electronically document and maintain the progression of students check sheets instead of normally using an old-fashioned pencil and paper. The estimated completion date for this project is set for December 2016. Upon completion, this system will be fully functional and implemented within Kutztown University of Pennsylvania for academic advisers.

1.1 Overview of the Organization

In order for the Kutztown University CSIT department to keep graduate students and undergraduate students on track, the faculty advises the students at the beginning of each upcoming semester. Kutztown University CSIT department is in need of an efficient system that will aid the student advisement process. The system will convert the process to an electronic based operation to allow faculty advisors to optimally advise advisees at the beginning of each semester.

1.2 Current Situation and Problem/Opportunity Statement

Kutztown University CSIT department is in need of an advisement system to help assist faculty during the advisement period in order to keep students on track for graduation. Currently the Kutztown advisement process involves the student bringing in a physical copy of the track sheet and requires the adviser to manually write on the physical track sheet copy. The current process of updating a student track sheet is not electronic, which makes the process tedious and un-uniform for the advisers. In order to fill this gap between the “old fashioned” advisement process and the conversion to an electronic process will require a user friendly system that will allow the adviser to assist the students in remaining on track for graduating.

1.3 Statement of Purpose

The purpose of developing this advisement system is to provide advisers with a user friendly system that will help aid the advisement process and to convert it to a fully electronic procedure. This system will create recommendations for course selections for future semesters, display courses that are offered for the semester, will produce warnings for graduation delays, generate clearances for each student and will enable advisors to edit student’s track sheets. This system must also be user friendly for both the adviser and the student. This will also enable a smooth, compact and efficient advisement process for advisors and students.

1.4 Project Objectives

The objective of this system is to aid faculty advisors be expediting the advisement process and preventing advisees from enrolling in a course sequence that would delay the advisee’s planned graduation date.

1.5 Project Requirements

Requirements and resources that will be needed include software for coding our project, hardware for the SAS implementation and time allocation.

1.6 Project Specifications

Project specifications consist of high level functional and non-functions specifications.

* Functional specifications
  + Provide a user friendly GUI in check sheet form for advisor to view, load and edit advisee’s academic records.
  + Recommend course sequences for upcoming semester.
  + Provide alerts if advisee’s at risk of deviating from advisee’s course sequence.
  + Keep track of advisee’s academic progress and stores student academic records.
  + Generate information to help advisor’s fill out the advisee form.
  + Allow advisor to store notes for each student.
* Non-Functional Specification
  + System will expedite advisement process.
  + System will prevent the advisee from choosing incorrect course sequences and deviating the student’s graduation.
  + System will be reliable.
  + System will be portable.
  + System will be efficient.

1.7 Critical Assumptions

- Assuming check sheet information, course catalog, university requirements and semester offerings are not modified during the semester after being posted to KU website.

- Assuming that the Kutztown website is online for accessing records through URL’s.

- Assuming advisee’s academic records are in the pars-able format.

1.8 Project Constraints

The most important constraints are the triple constraints – time, quality and resources. Constraints that pertain to the SAS include:

|  |  |
| --- | --- |
| ***Constraints*** | ***Brief Descriptions*** |
| ***Deadline*** | System must be completed by the end of December 2016. |
| ***Project Budget*** | There is no budget for the completion of the system. |
| ***Conflicting Schedule*** | Team members may have conflicting schedules. |

1.9 Project Acceptance Criteria

- Ability to create and update advisee academic records.

- Ability to generate future course sequences.

- Ability to recommend course scheduling for upcoming semester.

- Ability for advisor to store notes about advisee.

- Provide alerts to faculty advisor when advisee is in danger of deviating from the course sequence in such a way that would delay advisee’s planned graduation date.

- Ability to retrieve and update course information, university requirements, department check sheets, and semester offerings.

1.10 Project Success Criteria

- System shall be secure.

- System shall meet user expectations and requirements.

- System shall be feasible.

1.11 Project Deliverables

- Student Advisement System software.

- Faculty Advisor user manual.

- Project plan.

- System Requirements Specifications (SRS)

SECTION 2 PROJECT FEASIBILITY

Operational Feasibility

There is a need for an advisement system for both students and advisers. The Kutztown faculty needs this system to aid the advising process. This means prolonged use and thus, a very high operational feasibility.

Technical Feasibility

From a technical standpoint the system does not require any new or revolutionary technology. Though complex in a sense of design it will merely be an application that exists to provide information about the advisee to the advisor. The production of this system requires specific skill sets each team member must possess. Familiarity with HTML and JavaScript is required. As such similar technologies exist it is certainly feasible with the technology we have.

Economic Feasibility

* + Tangible benefits:
    - Saves advisor time
    - Reduces probability of human error
    - Saves advisee extra semesters of enrollment
  + Intangible benefits:
    - Decreased amount of unhappy users
    - Increased faculty advisor production (Most optimal guidance for advisee)

This project is very economically feasible. The cost is not an issue as this project will not have a monetary cost.

Schedule Feasibility

Schedule feasibility is the only point of concern. Members of the team have conflicting schedules. The project itself appears feasible given what features and systems are required.

SECTION 3 PROJECT WORK BREAKDOWN STRUCTURE (WBS)

1. Design Faculty Advisor and IT Maintenance UI’s.

2. Create Faculty Advisor and IT Maintenance UI Prototypes.

a. No data required.

b. Only used to gauge user experience.

3A. Scrape information we need about the items listed below from KU website. It is all publically accessible from the Kutztown website, so that is fair game for us.

a. Check sheets.

b. University requirements.

c. Department requirements.

d. Track requirements.

e. GPA requirements.

f. Grad school entry requirements.

g. Prerequisite major/course information.

h. Course catalog.

i. Semester courses offered.

4. Compile all test data.

a. Include "fake" student records, and advisement forms.

5. Determine entities and relationships.

6. Design E-R diagrams.

7. Analyze data to design advising algorithms.

a. Based on prerequisite courses, taken courses, course sequence, course availability, GPA, major/track, etc.

8. Design algorithm to provide alerts if the student is deviating from recommended course sequence.

9. Design algorithm to make scheduling recommendations.

10. Design algorithm to provide information for advisement forms.

11. Create and populate integrated database.

a. All ER diagrams will be implemented in one integrated database.

b. Testing required at this phase.

12A. Code Faculty Advisor UI

a. Testing required at this phase.

12B. Get user feedback on usability and feel of interface

13. Implement advising algorithms.

a. Testing required at this phase.

14A. Code IT Maintenance UI.

a. Testing required at this phase.

14B. Get user feedback on usability and feel of interface.

15. Get user feedback on all parts of the system.

SECTION 4 PROJECT COST ESTIMATE

Although many professional projects must be funded and budgeted, our team is sponsored by Kutztown University CSIT Department and has access to resources that would regularly cost a professional team. Resources we have access to include Kutztown database records, database storage and software to produce code for implementing our project. If we need a specific piece of hardware or software, a limited budget may be prepared by the project sponsor. Kutztown faculty is also willing to assist in any problematic occurrences.

SECTION 5 PROJECT SCHEDULE

|  |  |  |
| --- | --- | --- |
| ***Task ID*** | ***Description*** | ***Date/Duration*** |
| ***01*** | Meet with Holly Fox about MyKU database and our system. | 2/23/2016 |
| ***02*** | Find or create a collection of check sheets for aiding in designing algorithms for system. | 2/25/2016 |
| ***03*** | Plan check sheet format and design. | 2/29/2016 |
| ***04*** | Gather information about GUI and its presentation. | 3/8/2016 |
| ***05*** | Gather data on advisor and their needs by this time to start planning GUI. | 3/15/2016 |
| ***06*** | Phase 1 Revision. | 3/24/2016 |
| ***07*** | Early prototyping and interface for GUI. | April / August 2016 |
| ***09*** | Early prototyping of check sheets. | April / August 2016 |
| ***10*** | Work on database and back end connections to the database for system. | August |
| ***11*** | Refine user interface and account implementation. (If needed) | September |
| ***12*** | Develop and plan code for hooking up the GUI and the database. | October |
| ***13*** | Start testing and large scale debugging. | November |
| ***14*** | Project final testing | Late November 2016 |
| ***15*** | Project completion | December 2016 |

SECTION 6 RESOURCE ALLOCATION

The most important resource we need to keep track of and manage is time. Each member of the team will devote a minimum of 5 hours per week to a maximum of 20 hours a week to plan, discuss and develop our system. Due to a large variation in our team member’s schedules, this time resource may be allocated at each team members discretion, provided they continue to complete the agreed upon project activities.

|  |  |  |
| --- | --- | --- |
| ***Task ID*** | ***Resource(s)*** | ***Task Description*** |
| ***01*** | Time | Time needed to plan and complete project. |
| ***02*** | Software | Software needed for developing code. |
| ***03*** | Hardware | Computers for documentation, developing code and gathering resources. |
| ***04*** | Storage | Storage for student check sheets and other information. |
| ***05*** | Student Information | System requires student information as input |

SECTION 7 RISK MANAGEMENT PLAN

RISK IMPACT

5 – Risk that can very likely harm the system and cause failure to function. Very high impact on project cost and performance.

4 – Risk that will have high potential to slow progression of the project and greatly impact project cost or performance.

3 – Risk that will have slight potential to slow progression of the project and may impact project cost or performance.

2 – Risk that will have little potential to slow progression of the project and may impact project cost or performance.

1 – Risk that will have a very negligible impact on the project.

RISK IDENTIFCATION TABLE

|  |  |  |  |
| --- | --- | --- | --- |
| ***Possible Risks*** | ***Probability*** | ***Impact*** | ***Exposure*** |
| ***Certain parts of the project do not interface with the required systems.*** | 50% | 5 | 2.5 |
| ***Unable to access data we need*** | 20% | 5 | 1.0 |
| ***Data is lost as well as all backups*** | 20% | 5 | 1.0 |
| ***Delayed completion time*** | 20% | 5 | 1.0 |
| ***One or more people are not available to finish the project.*** | 50% | 4 | 2.0 |
| ***Requirement changes last minute.*** | 50% | 4 | 2.0 |
| ***Portions of the project are delayed.*** | 50% | 3 | 1.5 |
| ***There is too much required for us to implement on time.*** | 50% | 4 | 2.0 |
| ***Bugs with software*** | 50% | 4 | 2.0 |
| ***The group is unable to meet and work on the project.*** | 50% | 4 | 2.0 |
| ***The group is disorganized and not on the same page*** | 50% | 4 | 2.0 |
| ***Database is inaccessible by software*** | 20% | 4 | 0.8 |
| ***User is unable to input and store data*** | 20% | 4 | 0.8 |
| ***Not maintainable*** | 20% | 4 | 0.8 |
| ***Permanent Loss of a team member*** | 20% | 4 | 0.8 |
| ***Change of Requirements*** | 20% | 4 | 0.8 |
| ***Will not meet expectations of user*** | 20% | 4 | 0.8 |
| ***Our data is lost*** | 20% | 4 | 0.8 |
| ***Budget is reduced*** | 5% | 2 | 0.1 |

PRIORITIZED RISK IDENTIFICATION TABLE (SORTED BY EXPOSURE)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Number*** | ***Possible Risks*** | ***Probability*** | ***Impact*** | ***Exposure*** |
| ***1*** | Certain parts of the project do not interface with the required systems. | 50% | 5 | 2.5 |
| ***2*** | One or more people are not available to finish the project. | 50% | 4 | 2 |
| ***3*** | Requirement changes last minute. | 50% | 4 | 2 |
| ***4*** | There is too much required for us to implement on time. | 50% | 4 | 2 |
| ***5*** | Bugs with software | 50% | 4 | 2 |
| ***6*** | The group is unable to meet and work on the project. | 50% | 4 | 2 |

RISK MANAGEMENT PLANNING

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Information sheet** | | | | |
| **Risk ID**:  1 | **Date**:  4/12/16 | **Probability**:  50% | **Impact**:  5 | **Exposure**: 3.5 |
| **Description**:  The web based application for this system is unable to extract the required files from the Kutztown University webpage. | | | | |
| **Refinement/context**:  Sub condition 1: Outside websites containing information we need may be unavailable to the application due to formatting or location changes of the stored data. | | | | |
| **Mitigation/monitoring**:  Set up prototype to see if the data is available.  Check the data and make sure we have what we need. | | | | |
| **Management/contingency plan/trigger**:  1. Reduce our scope to compensate for loss of available data.  Trigger: Mitigation steps unproductive as of 3/26/2016. | | | | |
| **Current status**:  3/14/2016: Mitigation steps initiated | | | | |
| **Originator**: Alan Duffy-Guy | | **Assigned**: Geoffrey Pitman | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Information sheet** | | | | |
| **Risk ID**:  2 | **Date**:  4/12/16 | **Probability**: 50% | **Impact**:  4 | **Exposure**:  2.0 |
| **Description**: The project team loses a member of the team for the duration of the project | | | | |
| **Refinement/context**:  Sub condition 1: 1 member permanently leaves the team.  Sub condition 2: More than one member permanently leaves the team. | | | | |
| **Mitigation/monitoring**:  Stay on good terms with all members.  Keep track of their moods and anything negatively impacting them. | | | | |
| **Management/contingency plan/trigger**:  1. Split up work amongst current members  2. Spend extra time on work  Trigger: Mitigation steps unproductive as of 3/26/2016 | | | | |
| **Current status**:  3/14/2016: Mitigation steps initiated | | | | |
| **Originator**: Alan Duffy-Guy | | **Assigned**: Alan Duffy-Guy | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Information sheet** | | | | |
| **Risk ID**:  3 | **Date**:  4/12/16 | **Probability**: 50% | **Impact**:  4 | **Exposure**:  2 |
| **Description**:  One or more parts of the project get delayed. | | | | |
| **Refinement/context**:  Sub condition 1: One part of the project is delayed by more than 3 weeks.  Sub condition 2: More than one part of the project is delayed by more than 3 weeks. | | | | |
| **Mitigation/monitoring**:  1. Careful project planning  2. Schedule extra time | | | | |
| **Management/contingency plan/trigger**:  1. Ask members to spend more time working on project.  Trigger: Mitigation steps unproductive as of 3/26/2016. | | | | |
| **Current status**:  3/14/2016: Mitigation steps initiated. | | | | |
| **Originator**: Alan Duffy-Guy | | **Assigned**: Geoffrey Pitman | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Information sheet** | | | | |
| **Risk ID**:  4 | **Date**:  4/12/16 | **Probability**:  50% | **Impact**:  4 | **Exposure**: 2 |
| **Description**:  There have been more requirements planned than can be feasibly finished. | | | | |
| **Refinement/context**:  Sub condition 1: Falling behind on tasks and missing more than one milestone . | | | | |
| **Mitigation/monitoring**:  Carefully monitor the rate of competition. | | | | |
| **Management/contingency plan/trigger**:  Negotiate with sponsor to reduce project scope.  Trigger: Mitigation steps unproductive as of 3/26/2016. | | | | |
| **Current status**:  3/14/2016: Mitigation steps initiated. | | | | |
| **Originator**: Alan Duffy-Guy | | **Assigned**: Patrick Gagliano | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Information sheet** | | | | |
| **Risk ID**:  5 | **Date**:  4/12/16 | **Probability**:  50% | **Impact**: 4 | **Exposure**: 2 |
| **Description**:  Nearing the release of the project, there are too many bugs to get rid of in time. | | | | |
| **Refinement/context**:  Sub condition 1: Within 1 month of the release of the project there are more bugs than can be dealt with in a single month’s time. | | | | |
| **Mitigation/monitoring**:  Testing the product thoroughly at least once per milestone.  Allocating extra time for debugging. | | | | |
| **Management/contingency plan/trigger**:  1. Postpone other unnecessary parts of the project to work on debugging.  2. Add momentary members to team to help with debugging.  Trigger: Mitigation steps unproductive as of 11/01/2016 | | | | |
| **Current status**:  3/14/2016: Mitigation steps initiated | | | | |
| **Originator**: Alan Duffy-Guy | | **Assigned**: Patrick Gagliano | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Information sheet** | | | | |
| **Risk ID**:  6 | **Date**:  4/12/16 | **Probability**:  50% | **Impact**: 4 | **Exposure**: 2 |
| **Description**:  The Group is unable to physically meet up and work on the project. | | | | |
| **Refinement/context**:  Sub condition 1: Not having group meetings in over a week. | | | | |
| **Mitigation/monitoring**:  Acquire the schedule of all team members to find when they have free time.  Make free time in schedule if no free time exists. | | | | |
| **Management/contingency plan/trigger**:  1. Schedule online meetings if there are problems with scheduling.  2. Only meet with a few members and catch the rest of the members up later.  Trigger: Mitigation steps unproductive for as long as one week since last weekly condition check. | | | | |
| **Current status**:  No current initiation. | | | | |
| **Originator**: Alan Duffy-Guy | | **Assigned**: Alan Duffy-Guy | | |

SECTION 8 MEETING AGENDA

Our team plans to meet once, even possibly twice a week to discuss matters of the project. We also plan to meet with the project sponsor if there are any troubles we encounter. Our team will also set dates to meet and discuss matters with other important figures that may assist us in creating our system. We have recorded each of our meetings in a text file and will continue to update and maintain our records. Below is a preview of our records so far.

TEAM MEETING MINUTES

Name(s): Geoff, Alan, Patrick

Date: 2/4/2016

Time: 2:30-3pm

Minutes: 30 minutes (In Person)

1. Discussed projects we might be interested in.

2. Drew a list of pros and cons for each project.

3. Decided to use a weighted top 3 choice approach to be fair.

Name(s): Geoff, Alan, Patrick

Date: 2/9/2016

Time: 2:30-3pm

Minutes: 30 minutes (In Person)

1. Decided on doing the advisement project.

2. Discussed roles we might be interested in.

3. Will make the final decision next time we meet.

Name(s): Geoff, Alan, Patrick

Date: 2/10/2016

Time: 7-7:30pm

Minutes: 30 minutes (Online)

1. Decided on the roles we wanted (see project details)

2. Exchanged additional contact info.

Name(s): Geoff, Alan, Patrick

Date: 2/11/2016

Time: 7-7:30pm

Minutes: 30 minutes (Online)

1. Discussed setting up a meeting with project sponsor (Dr. Hussain)

- Our schedules do not match up during his office hours so 2 people will go.

- Will use a recording device to supplement.

2. Created project repository

3. Added deliverable due dates to repository calendar.

Name(s): Geoff, Alan, Patrick

Date: 2/14/2016

Time: 8-8:30pm

Minutes: 30 minutes (Online)

1. Came up with all the questions we thought we needed to ask in our interview with project sponsor.

Name(s): Geoff, Alan, Patrick

Date: 2/17/2016

Time: 9-9:15pm

Minutes: 15 minutes (Online)

1. Decided on work distribution for deliverables.

Name(s): Geoff, Alan, Patrick

Date: 2/21/2016

Time: 7-7:20pm

Minutes: 20 minutes (Online)

1. Submitted deliverables to project drive.
2. Gave each other constructive suggestions.

Name(s): Geoff, Alan, Patrick

Date: 2/23/2016

Time: 3pm-3:30pm

Minutes: 30 minutes (In Person)

1. Discussed how to re-approach our project after meeting with Holly Fox

* We will need to track down our own data.
* Also need to make necessary corrections to deliverables to reflect the new information.

Name(s): Geoff, Alan, Patrick

Date: 2/25/2016

Time: 3pm-3:30pm

Minutes: 30 minutes (In Person)

1. Decided to create our own test files because we could not access student’s records from MyKU database.
2. Discussed possible file formats for student track sheets.

Name(s): Geoff, Alan, Patrick

Date: 3/1/2016

Time: 3pm-3:30pm

Minutes: 30 minutes (In Person)

1. Sat down after class and decided what is required for when we come back from spring break.

Name(s): Geoff, Alan, Patrick

Date: 3/15/2016

Time: 3pm-3:30pm

Minutes: 30 minutes (In Person)

1. Professor returned grades for phase 1
2. Decided what to revise from phase 1.
3. Discussed problems and solutions for our revision.
4. Discussed available times for revision.

* Deliverables due by March 24th

Name(s): Geoff, Alan, Patrick

Date: 3/17/2016

Time: 3pm-3:30pm

Minutes: 30 minutes (In Person)

1. Split up roles for phase 1 revisions
2. Decided to meet later in the evening to begin revisions.

Name(s): Geoff, Alan, Patrick

Date: 3/17/2016

Time: 6-9pm

Minutes: 3 Hours (In Person)

1. Met in the library to work on revisions.
2. Project Scope, Feasibility study, Risk Management Plan and Project cost estimate revised.
3. Discussed requirements for SRS and analysis documents.
4. Will meet online over this weekend.

Name(s): Geoff, Alan, Patrick

Date: 3/20/2016

Time: 8-9pm

Minutes: 1 Hour (Online)

1. Prioritized and decided what to work on for Tuesday 3/22/2016
2. Edited and uploaded revised documents over online Microsoft meeting.

Name(s): Geoff, Alan, Patrick

Date: 3/22/2016

Time: 3:30-9pm

Minutes: 5.5 Hours (In Person)

1. Worked on Work Breakdown Structure, Risk Management Plan, System Requirements Specifications, Analysis Use Case Diagram.

Name(s): Geoff, Patrick

Date: 4/10/2016

Time: 4-8pm

Minutes: 4 Hours (In Person)

1. Worked on Domain Model and Fully Dressed Use Cases.
2. Planned for meeting after getting project feedback.

Name(s): Geoff, Alan, Patrick

Date: 4/11/2016

Time: 3:00-7pm

Minutes: 4 Hours (In Person)

1. Got project feedback.
2. Worked on phase 2 deliverables
   1. Phase 1 Revisions
   2. Use Case Diagram
   3. Fully Dressed Use Cases
   4. Domain Model
   5. Meeting Minutes

USER MEETING MINUTES

Meeting with Dr. Hussain on Monday March 24, 2016 from 1:05pm-1:58 pm

Attendees: Alan Duffy-Guy, Geoffrey Pitman, and Dr. Hussain

Meeting Minutes: 53 Minutes

1. There are 2 checkpoints in the advisement system that the user would like the advisement checking system to check for the status of the advisee
   1. Help check the Junior clearance from
   2. Help Check the Graduation clearance form
2. The Data received does not need to deal with any privacy concerns as long as it is securely stored
   1. No one else but authorized personnel should be able to access that information
   2. The advisor has the right to look at that information as it is the Advisor’s job to see an assess it.
3. My KU is still out of scope
   1. Student records are not publically available
   2. Everything else is available publicly.
   3. However not everything that is publicly available is easy to access or put into the system
   4. Transfer students’ data, however is not totally available because the competencies are not completely there (such as if they have completed a Visual Learning Course)
4. The Advisor may have to, either by themselves or with the advisee, fill in the competences.
5. The scope needs to be reduced to only dealing with Computer Science Department and not all of KU
   1. This is because it will be harder to acquire the other department’s requirements and prerequisites are for each department, other than looking them all up manually.
   2. The other departments do not have prerequisites in a parse able format.
6. Prerequisites should be saved with check sheets optimally as an object.
7. System should give these popup notifications as warnings and not limit the user’s ability to make these check sheets
8. Algorithm should match the required course to its competency
9. Perhaps “Category Options” for specific elective requirements (like art or music for example) Use this to show options to the advisor.
10. Acceptance Criteria
    1. System makes recommendations for the upcoming semester
    2. Makes warning about sequence deviations that cause the advisee to graduate later than accepted
    3. generates check sheets
    4. work for undergraduate students and graduate students also
    5. System must be secure
11. Success Criteria that should also be met
    1. Check junior clearance form
    2. Check graduate clearance form
    3. Both are just verifying from a check list
    4. Examples will be provided
12. As far as differences between undergrad and grad students there are not many known by the project sponsor
    1. The 400 degree is counted in the undergrad and grad
    2. As far as the process is concerned it should be the same
13. The fields in the checks sheet could be modifiable.
    1. The file could be uploaded into the checking system.
14. Request for a prototype of the interface to be shown.
    1. A demo should be shown to the project sponsor in person
    2. Would like to see the prototype as soon as possible
    3. Let project sponsor know ahead of time
    4. Show prototype before the system is started
15. Extra suggestions from The project sponsor for features
    1. Substitutions should be accounted for in the system
    2. Updates will be saved somewhere also
    3. Notes should be able to be written and saved with the checklist
16. Meeting Adjourned

SECTION 9 WORKLOAD DISTRIBUTION

|  |  |  |
| --- | --- | --- |
| Name | Activity | Distribution |
| Patrick Gagliano | 1. Project Plan Revision  2. Revision History  3. Project Scope Subsections  4. Project Cost Estimate  5. Resource Allocation  6. Team Meeting Minutes  7. User Meeting Minutes  8. Use Case Diagram  9. Fully Dressed Use Cases  10. Domain Model | 33.3% |
| Alan Duffy-Guy | 1. Project Plan Revision  2. Project Scope Subsections  3. Project Feasibility  4. Project Schedule  5. Resource Allocation  6. Risk Management Plan  7. Team Meeting Minutes  8. User Meeting Minutes  9. Domain Model | 33.3% |
| Geoffrey Pitman | 1. Project Plan Revision  2. Project Scope Subsections  3. Work Breakdown Structure  4. Resource Allocation  5. Meeting Agenda  6. Team Meeting Minutes  7. User Meeting Minutes  8. SRS  9. Fully Dressed Use Cases  10. Domain Model | 33.3% |

SECTION 10 ADVISEMENT SYSTEM USE CASE DIAGRAM

C:\Users\Helpdesk\Desktop\UCA1.0 (1).png

C:\Users\gpitm379\Desktop\turnin\UCM1.0.png

SECTION 11 FULLY DRESSED USE CASES

**Use Case UC1**: Maintain Advisee Records

**Level**: User goal

**Primary Actor**: Faculty Advisor

**Stakeholders and Interests**:

* Faculty Advisor: Wants to keep accurate and up to date information on advisee records because warnings and recommendations are based on the accuracy of this information. Wants to easily retrieve advisee records and view retrieved records.
* Advisee: Wants advisors to have up to date and accurate information so advisee can be optimally advised.

**Preconditions**: Faculty advisor is authenticated. Advisor has been provided with returning advisee's grades from previous semester, or, advisor has been provided with transfer student advisee's unofficial transcript file.

**Success Guarantee (or Post conditions)**: Records are manually entered, and/or loaded from file. Existing records are retrieved successfully. Retrieved records are modified and updated successfully.

**Main Success Scenario** **(or Basic Flow)**:

1. A. Faculty Advisor creates/retrieves record for advisee.
2. A. Faculty advisor attempts to update record.
   1. Advisor enters requisite information and/or uploads transcript.
3. A. Faculty Advisor Check sheet UI is deployed.
   1. Alert is issued about possible graduation delays. Advisor must acknowledge receipt of warning to continue. Warning is displayed in UI until flagged class is added to advisee record as being taken and passed...
   2. Faculty Advisor prints form.
   3. Faculty Advisor adds note.
   4. Faculty Advisor prints recommended schedules.
   5. Faculty Advisor attempts to update record (Repeat step 2.A)

**Extensions (or Alternative Flows)**:

1. B. Advisor attempts to create student record.
   1. Record fails to be created.
      1. Repeat step 1.B.
2. B. Faculty advisor fails to retrieve advisee record from SAS.
   1. SAS signals failed retrieval.
      1. Repeat step 2.B.
3. B. Faculty advisor attempts to update record.
   1. Advisor enters requisite information and/or uploads transcript.
   2. Record fails to update.
      1. SAS signals user of failure.
         1. Repeat step 3.B.

**Special Requirements**: Advisee must email advisor unofficial transcript before/at meeting.

**Technology and Data Variations List**:

1. Advisee’s unofficial transcript file must be in PDF format.

**Frequency of Occurrence**: Two advisement sessions per school year, per student.

**Open Issues**:

* Data inconsistencies in unofficial transcript file pertaining to core competencies/university requirements for transferred courses.

**------------------------------------------------------------------------------------------**

**Use Case UC2**: Maintain University & Department Data

**Level**: User goal

**Primary Actor**: IT Maintenance

**Stakeholders and Interests**:

* Faculty Advisor: Needs system to keep accurate and up to date information on CSIT catalog, course offerings, department check-sheets and course prerequisite data to optimally advise advisee.
* Advisee: Needs system to keep accurate and up to date information on CSIT course catalog, course offerings, department check-sheets and course prerequisite data for optimal advisement.

**Preconditions**: IT maintenance has been authenticated. Kutztown university servers are online.

**Success Guarantee (or Post conditions)**: University and department data is successfully updated.

**Main Success Scenario** **(or Basic Flow)**:

1. IT maintenance enters course offerings, updated CSIT catalog, and any new checksheets, in the Maintenance UI.
2. A. University and department data is successfully updated in SAS.

**Extensions (or Alternative Flows)**:

1. B. University and department data is unsuccessfully updated in SAS.
   1. SAS signals user of failure
      1. Repeat step 1.

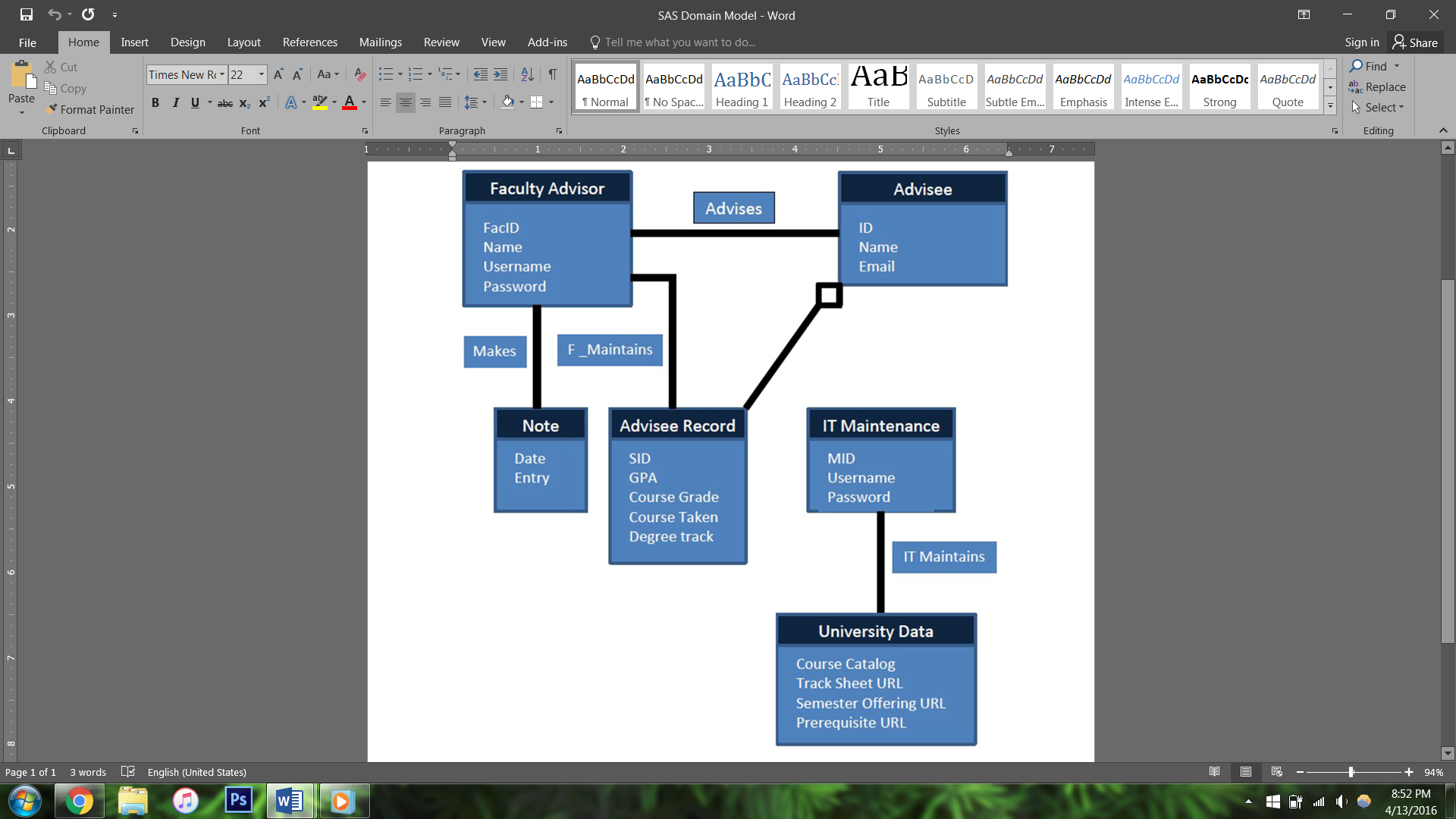
**Special Requirements**: n/a

**Technology and Data Variations List**: n/a

**Frequency of Occurrence**: Once per semester.

**Open Issues**: n/a

SECTION 12 DOMAIN MODEL



Software Requirements Specification

for

Student Advisement System

**Version 1. 2 approved**

**Prepared by: Geoffrey Pitman, Patrick Gagliano, Alan Duffy-Guy**

**Kutztown University of Pennsylvania: CSIT Department**

**9/22/2016**

**Table of Contents**

**Table of Contents**

**Revision History**

**1. Introduction**

1.1 Purpose

1.2 Product Scope

1.3 References

1.4 Document Conventions

**2. Overall Description 1**

2.1 Product Perspective 1

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Critical Assumptions 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 System Interfaces 3

**4. SAS Functionality 4**

4.1 Maintain Advisee Records 4

4.2 Maintain Advisor Notes 4

4.3 Maintain Universtiy and Department Data 5

4.4 Generate Schedule Recommendations 5

4.5 Generate Warnings 5

4.6 Generate Advisor-Form Data 6

**5. Other Nonfunctional Requirements 6**

5.1 Security Requirements 6

5.2 System Quality Attributes 6

**6. Disclaimer 6**

**Appendix A: Glossary 6**

**Appendix B: To Be Determined List 7**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Geoffrey Pitman | 9/22 | - Web scraping determined to be unreliable. | 1.1 |
| Geoffrey Pitman | 4/13 | -Resolved an open issue by discovering URL pointing to department course data.  -Removed section 4.4 ‘Maintain Sequence and Prerequisite data’ and changed 4.3 from ‘Maintain Course and Semester Data’ to ‘Maintain University and Department Data’ after discovering the above | 1.0 |

# Introduction

## Purpose

This SRS describes the software functional and nonfunctional requirements for release 1.0 of the Student Advisement System (**SAS)**. This document is intended for use by members of the project team that will implement and verify the correct functioning of the **SAS**, as well as the project sponsor in order to assess validity of requirements from the client perspective. Unless otherwise noted, all requirements specified here are high priority and committed for release 1.0.

## Product Scope

The **SAS** will assist **KU: CSIT** faculty advisors by recommending course sequences for the upcoming semester and providing warnings if the advisee is in danger of deviating from the advisee’s planned graduation date. A detailed project scope is available in section 1-A, on page 2 of the *Student Advisement System (Phase 1) – Project Plan* deliverable.

## References

1. Kirby, Ken. *Detailed SRS Outline*http://neon.airtime.co.uk/users/wysywig/srs\_mt.htm
2. Wiegers, Karl. *SRS for Cafeteria Ordering System* https://users.csc.calpoly.edu/~djanzen/courses/508W09/sampledocs/COS\_SRS.docOverall Description
3. Ian Sommerville. *Software Engineering 10e*

Course Text

## Document Conventions

Terms in **bold** will have definitions and/or an explanation of the term in the glossary section of the appendices.

# Overall Description

## Product Perspective

The need for this system came into being because the current advisement auditing system is not user friendly and does not provide sufficient warnings and/or recommendations for students and/or their advisors in regard to scheduling courses for the upcoming semester. The **SAS** will provide advisors with the tools to make the advisement process more streamlined and less prone to mistakes by visually demonstrating the recommended sequence of courses for the advisee and visual alerts when students are in danger of deviating from the recommended sequence of courses in such a way that would cause the advisee to graduate later than the advisee’s intended graduate date. Ultimately, the system will prove its worth if it can prevent even one student from having to enroll in an extraneous semester, saving that student months of time and thousands of dollars. If the system helps 10 students in the same way, it will save *years* of time and *over* 50,000 dollars.

## Product Functions

The following provides a high-level list of the system’s functions:

1. Provide graphical interface in check sheet form, for faculty advisor to view, edit and update advisee’s academic information.
2. Recommend course schedule for upcoming semester.
3. Provide alerts if advisee is at risk of deviating from advisee’s planned graduation date.
4. Keep track of advisee’s academic progress and store student academic records.
5. Generate information to help advisor’s fill out advisee forms.
6. Allow advisor to store notes about student.
7. Retrieve and update course information, semester offerings, university requirements, and department check sheets.

## User Classes and Characteristics

### Faculty Advisor

The main end-user of this system will be faculty advisors. This is any faculty member that has one or more advisees assigned to them

### IT Maintenance

The system may occasionally need to be updated or adjusted. As such, a secondary user, IT Maintenance, will perform the necessary updates and/or adjustments. This type of user must have the required privileges to access the system.

## Operating Environment

This system will run on a **Red Hat LAMP stack backend** and a web browser based **frontend**. It will be accessible from any device capable of downloading email attachments and running a browser.

## Design and Implementation Constraints

The major constraint on the system is that it is not allowed access to the **myKU** system, due to privacy concerns over student information.

## User Documentation

Both the Faculty Advisor user and the IT Maintenance user, will be provided with readily accessible documentation, available from the website hosting the browser based interface. Website **URL** to be determined at later date. See appendix B.

## Critical Assumptions

1. Requisite system data will be made publically available on **KU** website; course catalog, check sheet information, university requirements, and semester course offerings.
2. Said requisite data will not be modified by the University once posted for the upcoming semester.

# External Interface Requirements

## User Interfaces

### Faculty Adviser UI:

The user interface for the Faculty Advisor user will be displayed in the browser. The document will appear in the form of a **KU** check sheet. For existing advisee’s, there will be a search field for the advisor to locate the advisee’s records. There will also be a button to upload a parse-able file of advisee’s student records. The data from the file will then be populated into the check sheet fields. Advisors will be able to modify fields. There will also be a button to analyze the check sheet, which will generate the schedule recommendations, advisor forms information, and warnings. There will also be a button to clear data. Additionally, there will be a field for the advisor to add notes. A text field where advisors can enter notes will also be present.

### IT Maintenance UI:

The user interface for the IT Maintenance will be displayed in the browser. It will require authentication from a username and password. This **UI** will allow the user to change, via form fields, the **URLs** that point to the data needed by the system, and acquired from the **KU** website. Additionally, the user can make changes to course attributes, such as prerequisites and competencies. It will also have an update button to activate the data acquisition process, which will bring the system’s database up to date.

## System Interfaces

### SAS Interfaces with Browser

The system will interface with any standard web browser such as FireFox, Chrome, Edge, Safari, Explorer, Opera, etc. This will be accomplished with a **LAMP stack backend**, and **JavaScript/HTML/CSS** **frontend.**

# SAS Functional Requirements

## Maintain Advisee Records

This section describes the maintenance of advisee student records.

### Store Records

1. Student information is entered into the input fields of the check sheet form.

1. Can be entered manually by the Faculty Advisor user.
2. Can also be populated by loading file of advisee’s records.
3. Advisor clicks ‘Update’ button.
4. Advisee records are sent to the **SAS** database.

### Modify Records

1. Existing advisee records can be modified by the Faculty Advisor user
2. Once the check-sheet fields have been filled in to the user’s satisfaction, the ‘Update Records’ button can be clicked, which will update the advisee’s records in the **SAS** database.

### Retrieve Records

1. Existing advisee records can be retrieved by the Faculty Advisor user.
2. Advisor enters advisee’s student ID into the search field and clicks the ‘Retrieve’ button. The student data will then populate into the form fields.

ii. Advisor will be prompted to modify fields and/or upload current unofficial transcript

## Maintain Advisor Notes

This section describes the maintenance of the ‘Notes’ tool in the **SAS**. Faculty Advisors will be keep notes about their advisees.

### Store Notes

1. There will be a ‘Notes’ text field section in the Faculty Advisor **UI**
2. Advisor clicks “Add Note” button to add a note.

### Retrieve Notes

1. Existing notes can be retrieved by the Faculty Advisor user.

1. When advisee’s student records are retrieved, existing notes will populate in the ‘Notes’ text field.

## Maintain University and Department Data

This section describes the maintenance of the data derived from the university course catalog. upcoming semester course offerings, department check-sheets, and prerequisite course data.

### Update System Data

2. Accomplished from the IT Maintenance **UI** by entering and/or uploading the data and

clicking the ‘Update Data’ button.

1. Once a semester an IT Maintenance user must update the system
2. Must happen before advising starts, but after necessary data has been posted for the upcoming semester.

## Generate Schedule Recommendations

This section describes the scheduling recommendations that are generated by the system for advisees for the upcoming semester.

1. Faculty Advisor user clicks ‘Get Recommendations’ in the Faculty Advisor UI
2. User is taken to another screen
3. Schedule recommendations are presented in printable and visually pleasing manner
4. Multiple recommendations are given

## Generate Warnings

This section describes the warnings that the system generates when an advisee is in danger of delayed graduation.

1. When Faculty Advisor user clicks ‘Update Records’ (updates the advisee’s records) the system will generate popup warnings if the system identifies the advisee as a risk case.
2. Warnings can be overridden by Faculty Advisor user.

## Generate Advisor-form Data

This section describes information that the system will generate to help automate the task of filling out advising forms.

### Alert On Complete

1. When all possible data for a form has been generated, an alert is generated for the Faculty Advisor user.
2. Faculty Advisor user can choose to print out the form.
3. User can also download the form.

### Display Data

1. Form data is displayed under check sheet when advisee academic records are retrieved.

# Other Nonfunctional Requirements

## Security Requirements

Users must take standard privacy precautions when working with student data. Additionally, Faculty Advisor users and IT Maintenance users must provide login credentials to access their respective interfaces into the system.

## System Quality Attributes

### Provide Reliable Information

1. The **SAS** is, in large part, a data analysis system. As such, end users will be dependent upon the results provided from said data analysis. Reliable results are paramount to the system.
2. Reliable scheduling recommendations.
3. Reliable graduation delay warnings.
4. Reliable advisement form information.

# Disclaimer

The **SAS** is only to be used to assist Kutztown University **CSIT** department faculty advisors, and is intended solely for academic use.

**Appendix A: Glossary**

1. backend – server side of the system
2. CSIT – computer science and information technology
3. CSS – cascading style sheets used to help style websites
4. frontend – user side of the system
5. HTML – hyper text markup language is used to give content to websites
6. JavaScript – client side web scripting language
7. KU – Kutztown University of Pennsylvania
8. LAMP stack – Linux Apache MySQL PHP web development stack
9. myKU – academic software system used by Kutztown University
10. Parse-able – data that is in a format which can be processed algorithmically
11. Red Hat – proprietary implementation of the Linux operating system
12. SAS – Student Advisement System
13. UI – user interface

**Appendix B: To Be Determined List**

1. **URL** where the **SAS** interface can be accessed from.

**DEPENDANCIES CHART**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Activity | Duration (wks) | Dependencies |
| 10 | Plan project | 10 | - |
| 20 | Collect raw data | 2 | 10 |
| 30 | Analyze raw data | 4 | 20 |
| 40 | Create ER diagram | 4 | 30 |
| 50 | Analyze ER diagram (advising algorithm) | 4 | 40 |
| 60 | Analyze ER diagram (scheduling algorithm) | 3 | 40 |
| 70 | Design adviser UI | 6 | 40 |
| 80 | Design advising algorithm | 6 | 50 |
| 90 | Design scheduling algorithm | 6 | 60 |
| 100 | Implement advising algorithm | 4 | 70 |
| 110 | Implement scheduling algorithm | 3 | 80 |
| 120 | Implement adviser UI | 6 | 90 |
| 130 | Test advising algorithm | 6 | 100 |
| 140 | Test scheduling algorithm | 4 | 110 |
| 150 | Test adviser UI | 4 | 120 |
| 160 | Connect modules | 2 | 130, 140, 150 |
| 170 | Test SAS | 3 | 160 |
| 180 | Deploy SAS | 2 | 170 |

**\*SEE SCH2.1 IN TEAM REPOSITORY FOR DETAILED DEPENDANCIES\***

**PROJECT ESTIMATES METHOD**

The Expert Judgment Method was chosen as the time estimate technique of choice for this project. Because we have no previous work to take information on our past projects, we can’t use many of the algorithmic models as they would just be based off of our own estimations anyway. Therefore, the Expert Judgment Method is the best for the estimation of our project. It’s based on our own estimations and gives us a way to refine our estimations without using methods that take a lot of time to get the same result.