AbOut
MTM Program Product
Software Design Description

Version 1.3
05/01/2018

Applying Montana Tech SDD Standard Version 3.0

Project Client: *Faculty of CS Department*

Project Manager: *Celia Schahczenski*

2013 Project Team: *Foster Cade, Jeff Hall, Clint Hillerman, Jake Jones, Brian D. Knopp, Matt Morris, PJ Neary, Tom Powell, Frank Scholey, Alex Thompson, Logan Warner*

2014 Project Team: *Jon Wareham, Adam Cass, Justin Emge, Garrett Brown, Ben Butcher, Haythem Memmi*

Document Authors: *Cade J. Foster, Brian D. Knopp, Jon Wareham, Adam Cass*

Standard Version Number: 3.0
Standard Version Date: April 5, 2012

Version	*Date (mm/dd/yyyy)*	*Authors*	*Comment*
1.0	03/01/2013	Cade J. Foster, Brian D. Knopp	Initial version.
1.1	03/02/2014	Jon Wareham, Adam Cass	Made the document format more consistent and added details about the new UI and cosmetic features.
1.2	04/01/2014	Celia Schahczenski	Updates based on ESOF 326, Sp14

1.3	/5/01/2018	Students of ESOF 326	Updates based on changes made in ESOF 326

Montana Tech Software Engineering Students:
These Montana Tech Method software engineering standards encapsulate Dr. Ackerman's decades of experience in the software industry, the IEEE software engineering standards, and many suggestions from various texts. They have gone through many revisions and additions over the last several years. They are part of your software engineering studies so that (1) you may have the experience of developing software to a standard (which you may find you need to do if you take a job that requires high quality software), and so that (2) you will have the experience of developing high quality software. You are also invited to participate in the continuing evolution of these standards by studying them critically and making suggestions for their improvement and correction.

Table of Contents

1 Introduction

The purpose of AbOut's design is to help organize the drafting of the code for the software in a fluent, logical, seamless, efficient, and timely manner. It is therefore intended to be used by the programmers. The scope of AbOut's design can be found in AbOut's SRS document. This version was specifically written to be used by Montana Tech of the University of Montana, but a more widespread use of the product is a hopeful anticipation.

1.1 Purpose

In order to facilitate a common and consistent design during the implementation, refinement, and maintenance of AbOut, the proscriptions contained in this document are to be strictly followed. The intended audience of this document is the programmers and maintainers of the AbOut system.

1.2 Scope

See *AbOut SRS.*

1.3 Definitions, Acronyms, and Abbreviations

ABET	Accreditation Board for Engineering and Technology, Inc; the accrediting body for the Computer Science and Software Engineering programs.
CAS	Central Authentication Service
CORE	Course Outcome Review and Evaluation
SDD	Software Design Description
SRS	Software Requirements Specification

1.4 References

None.

2 Design Background

2.1 Background Information

Stakeholders:

Celia Schahczenski	cschahczenski@mtech.edu	406-496-4383
Frank Ackerman	fackerman@mtech.edu	406-496-4858
Jeff Braun	jbraun@mtech.edu	406-496-4206
Keith Vertanen	kvertanen@mtech.edu	406-496-4385
Michelle Van Dyne	mvandyne@mtech.edu	406-496-4855
Tamara Windham	twindham@mtech.edu	406-496-4366

As AbOut will be used to compile reports based upon student grades, which are indefinitely stored in a database, the system needs to be secure against intrusions or unauthorized access. Any actions or security flaws which jeopardize the confidentiality of student grades should be identified and rectified.

Access to AbOut depends on CAS and the interface between said entities.

AbOut utilizes a MySQL database to store information relating to administrators, faculty members, students, and student grades. Additionally, PHP is necessary to serve the dynamically changing contents of the AbOut web-based system.

2.2 Alternative Designs Considered

An alternative to a web-based user interface was a downloadable client coded in C#. The decision to reject this interface in favor of the current HTML and PHP-based system was one of pure utility. A web interface is significantly more accessible and portable in the current age of mobile applications.

3 User Characteristics

Potential users of this system include the faculty and administrative associate of the Computer Science Department. The assumed level of expertise needed to operate the system is a familiarity with web browsers and dynamic form webpages.

The initial webpage that a user will view is the login prompt, interfacing with CAS and providing a secure method to authenticate user credentials. The various pages which the user may access after login satisfy the requirements of faculty and administrators. These include adding, removing, and modifying outcomes, courses, offerings and assessments. Reports summarizing assessment data may also be generated.

4 System Architecture

The architectural design of AbOut is that it contains reusable modules. All essential files are located in their own logical 'units'. These files are located in the 'includes' directory and are named as follows: header.php, navBar.php, and footer.php. The default faculty and admin views (as accessed on login) and the reports view include these files at their logical locations. In code, which is to be tested, these files can be included as follows:

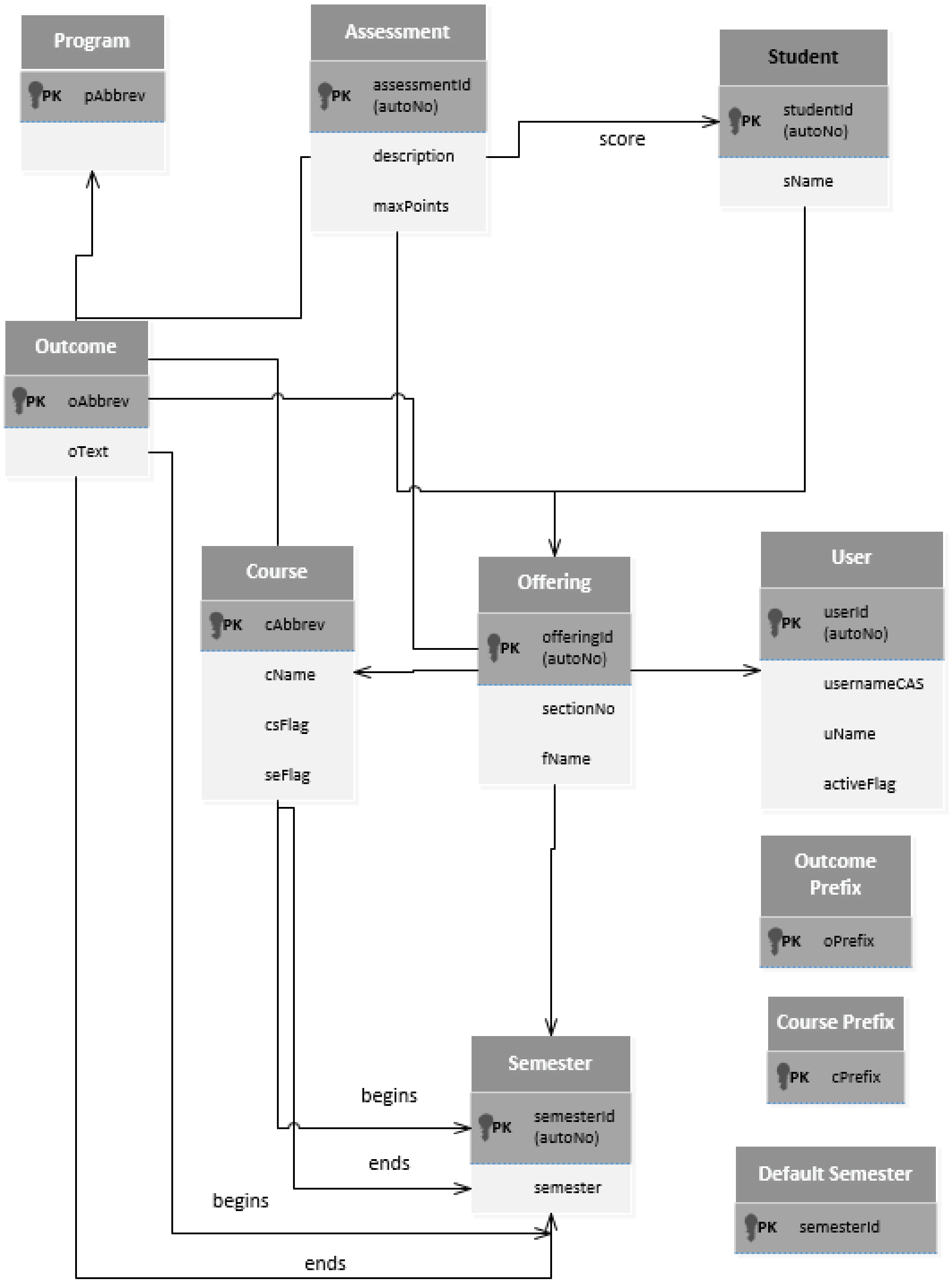
```
include( '../includes/header.php' );
include( '../includes/navBar.php' );
...
YOUR CODE HERE
...
include( '../includes/footer.php' );
```

5 Data Architecture

A database system is required by AbOut to store faculty, staff, and student information, as well as courses and their accompanying assessments. The entities and their interdependent relationships are documented in this section.

5.1 Data Analysis

A conceptual data model for AbOut follows.



A description of the fields in the data models is given next.

Field descriptions:

Program:	pAbbrev:	Program abbreviation.
Outcome:	oTest:	Outcome text.
	pAbbrev:	Program abbreviation.
	endSem:	Ending semester.
	beginSem:	Beginning semester.
AssessOutcome:	asessId:	Assessment identification.
	oAbbrev:	Outcome abbreviation.
Assessment:	asessId:	Assessment identification.
	description:	Assessment description.
	maxPoints:	Maximum number of points.
	offeringId:	Offering identification.
CourseOutcome:	cAbbrev:	Course abbreviation.
	oAbbrev:	Outcome abbreviation.
OffOutcome:	offeringId:	Offering identification.
	oAbbrev:	Outcome abbreviation.
Score:	asessId:	Assessment identification.
	studentId:	Student identification.
	Score:	Score.
Course:	cAbbrev:	Course abbreviation.
	csFlag:	Computer Science flag.
	seFlag:	Software Engineering flag.
	beginSem:	Beginning semester.
	endSem:	Ending semester.
Offering:	offeringId:	Offering identification.
	sectionNo:	Section number.
	fName:	Faculty name.
	userId:	User identification.
	cAbbrev:	Course abbreviation.
	semesterId:	Semester identification.
Semester:	semesterId:	Semester identification.
	semester:	Semester.
Student:	studentId:	Student identification.
	sName:	Student name.
	offeringId:	Offering identification.
User:	userId:	User identification.
	usernameCAS:	Username on Central Authentication System.
	uName:	Username.
	activeFlag:	Active flag.
CoursePrefix:	cPrefix:	Course prefixes.
OutcomePrefix:	oPrefix	Outcome prefixes.

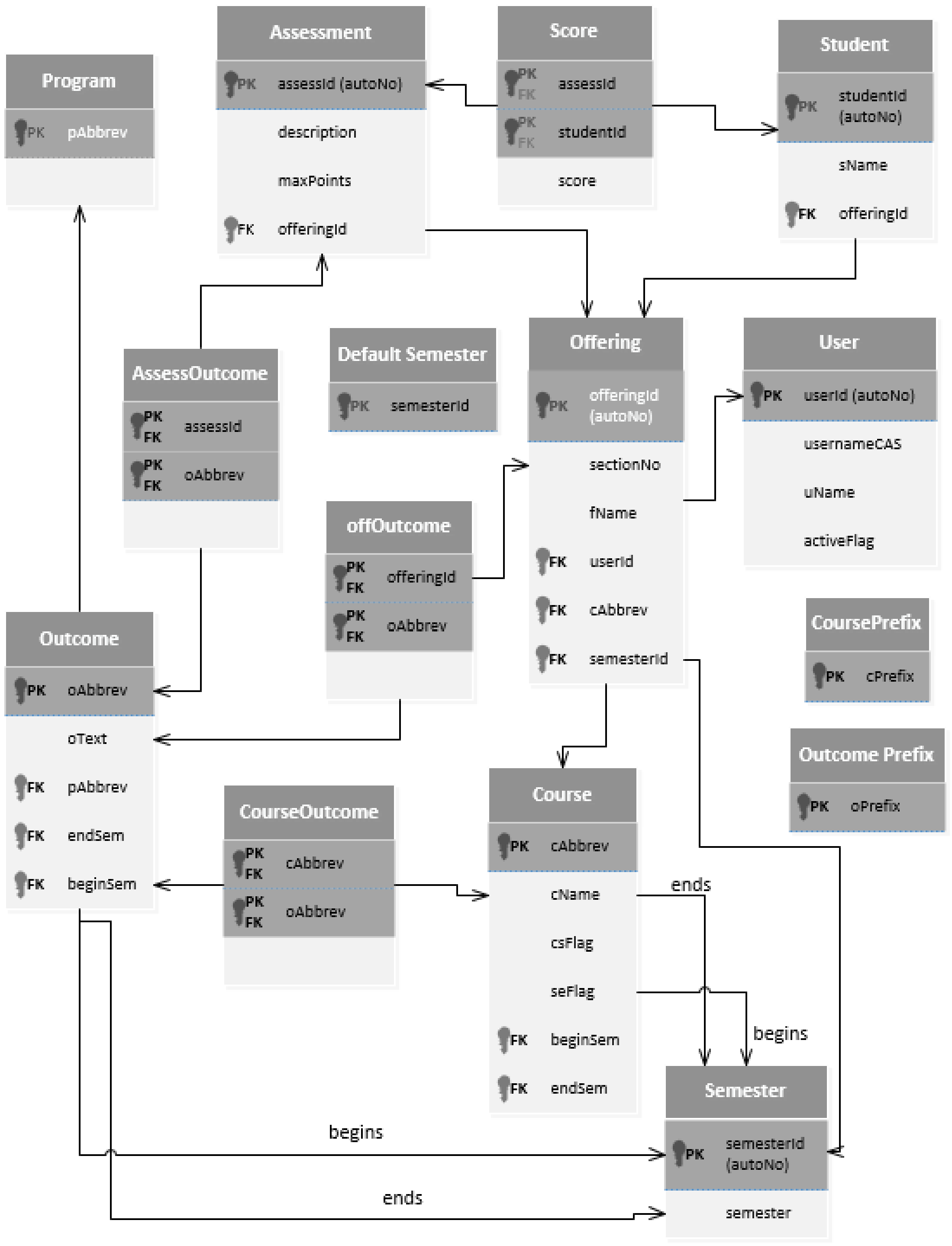
The data analysis activities that AbOut uses are the generation of reports. The data being analyzed is the student scores on the assessments, the max possible score on the assessments, the outcomes in the system, and the offerings in the system which consist of course data and semester data. This data is being compared against the percent required for a student to achieve on an assessment to be considered to have passed that assessment. The outcomes are tied to assessments by the faculty creating the assessments and scores are entered for the students on those assessments. All of this information is pulled from the database, manipulated in code, and output for the various users to see by generating various reports. The SRS document states that in order to change the percent required for students to achieve on an assessment to be considered to have passed the assessment, that change must be done directly in code. Therefore, changing this percent will alter the system design because that percent is going to be compared against when generating the reports.

5.2 Output Specifications

Separate PHP scripts will be created which support the reports generation and viewing functionality of AbOut. The database is to be queried by these scripts when the query is considered complicated, is used frequently in the system, or both. Otherwise the database is queried directly with SQL queries to access the necessary and relevant information to compile an outcome, CORE, or overview report.

5.3 Logical Database Model

The conceptual model translates to the following logical data model.



5.4 Data Conversion

All AbOut data is being added using the AbOut user interface. Therefore, no data conversion is necessary.

6 Interface Requirements

6.1 Required Interfaces

This system must interace with the Central Authetificatino System, CAS.

6.2 External System Dependencies

Aside from CAS mentioned above, AbOut does not have other external system dependencies. CAS is updated periodically, so some maintenance of the AbOut code may be required.

7 User Interface

The appearance of the AbOut website will be based on Montana Tech's school colors of green and copper. A Montana Tech banner will be displayed with a picture of the campus consistently across all sections of the site. The colors, banner, and other Montana Tech interface elements will match those from the Montana Tech website. Refer to the following Montana Tech style guides:

- General guidelines – http://www.mtech.edu/pr/style-guide.htm
- Visual Guidelines - http://www.mtech.edu/pr/visual-guides.htm
- Logon User & Downloads - http://www.mtech.edu/pr/logo-use.htm

Error messages may be in red even though red is not a Montana Tech color.

The AbOut functions are naturally divided between administrative functions, faculty functions, and reports. The interface should show such a division.

When a user logs in to AbOut, the screen displayed should be specific to that user. Faculty members should be able to perform administrative functions, but not vice-versa, since it doesn't make sense for an administrator to add assessments for someone else's course.

Once the user is logged-in, a welcome message such as "Welcome *first name last name"* should be displayed with a "logout" link close by. A "help" link should also be displayed.

AbOut must keep the users informed about changes in its state. When changes are not automatically clear, a message shall be displayed informing the user of the change.

Similarly, users must be informed about any errors. In case of errors, a message shall be displayed informing the user of the error.

Both error and informational messages should appear for around 5 seconds and automatically disappear. Alert boxes should not be used, as they need to be released and releasing them can be annoying to the users.

Error messages should contain red text, while informational messages can contain black text. The messages should appear in a box with a white background and so they will be visible in front of a variety of backgrounds. They should be positioned centered on the page horizontally, and centered between the AbOut box and the bottom of the top picture, vertically. (Coding information – JavaScript will be used to display HTML "div"s containing the messages.)

SAVE, RESET, and DELETE buttons only appear on pages where they are relevant. If they are relevant to a page they are always there but are faded when clicking them would not cause a change.

Buttons aside from those listed above, appear near the top of the page.

When saving an assessment, a "saved" message will appear at the top of the page. The "saved" message will disappear after 5 seconds or when the user begins to make changes to any field.

The system shall notify the user if an error occurs in the faculty section.

7.1 Login Interface Design

The login interface shall facilitate authorized access to AbOut by registered faculty or administrators. It shall serve as a standard login page interfacing with CAS to provide consistent and reliable login services. A button serves to redirect the user to the CAS login page. From here, the user is required to enter their username and password in separate fields. CAS proceeds with either a redirection to the AbOut main page or a notice of login failure.

A windowed component will work with CAS to validate the user's authenticity and permission to use the system. It will also determine whether the user is an admin, or a faculty/staff member. Since the validation process is done through CAS, it is beyond the scope of this document. However, all relevant user information (uName, usernameCAS, activeFlag, userId) will be determined after login and used in the system throughout the duration of time that the user remains logged in. This will determine which offerings (and therefore assessments tied to those offerings, and all information contained therein) will be viewable to the user (faculty and staff can only view offerings that they are teach). An administrator can see all offerings.

7.2 Reports Interface Design

Deleted. See section 8.3.

7.3 Main Page Interface Design

The main page serves as a central hub to access faculty, admin, and reports pages. These pages are accessed by their respective tabs in the central menu-driven interface.

The main page simply serves as a means of centralized access for faculty, admin, and overview reports, through hypertext links. No significant data entry or processing occurs in this object, aside from the determination of a user's faculty/admin status.

Upon login to AbOut, the user is redirected to the main page, which will automatically show the appropriate interface tab for the user's type (faculty see course offerings which they teach, etc…). A user may navigate to the other data views (faculty, admin, reports) using the tabs at the top of the courses list.

The faculty and administration sections will be navigated though a tree which displays the appropriate items for the corresponding section. Each part of the tree can be expanded or collapsed by the user to display more or less information. The links in the tree will change color when the mouse hovers over them.

A "help" button will appear beneath the user's name in the upper right corner of the screen with the logout button to the right of it. These buttons will appear in every section of the site.

7.4 Faculty Interface Design

Faculty may access the course offerings which they are assigned by clicking on the hyperlink of the name of their courses in the courses list. The main viewing pane will then display the relevant information for a course, including associated outcome and students. Faculty may add students to their course offerings, add assessments, and score assessments. A CORE Report may also be generated in the main viewing pane. A dash will always be used after the outcome abbreviation to separate it from the proceeding text.

When viewing or editing the assessment, the students, points, and percentages will be displayed in a table with the column headings: "Students", "Points", and "Percentage".

7.5 Admin Interface Design

Administrators have the ability to select a course from a list of all courses and offerings. The administrator may associate outcomes with a specific course, change which semester a course is offered in, and add course offerings. A course offering may then be associated with its respective faculty member. A dash will always be used after the outcome abbreviation to separate it from the proceeding text.

Course outcomes and their associated text may also be added or edited by the administrator in the latter subsection of the list panel.

When saving a course, offering, or outcome, a "saved" message will appear at the top of the page. The "saved" message will dissappear after 5 seconds or when the user begins to make changes to any field.

The administrator may also add and edit user information by changing the name and CAS login name of the user.

8 Detailed Design

AbOut has been entirely rewritten. The description of the re-write belongs here.

Section 7, User Interface, is not written well. Much of it needs to be deleted, however, some of the content there may be appropriate for here.

8.1 Description for Administration Component

Information from the previous sections needs to be placed here.

8.1.1 Processing for Administration Component

Information from the previous sections needs to be placed here.

8.1.2 Administration User Interface

Information from the previous sections needs to be placed here.

8.2 Description for Faculty Component

Information from the previous sections needs to be placed here.

8.2.1 Processing for Faculty Component

Information from the previous sections needs to be placed here.

8.2.2 Faculty User Interface

Information from the previous sections needs to be placed here.

8.3 Description for Report Component

A user is able to generate Outcome, Overview Course, Overview Outcome, C.O.R.E., and Matrix reports which summarize the various course data necessary to compile an ABET assessment.

Depending on the report being generated, this component will work with various data elements, as described below.

Overview Outcome Report: This report shows the percentage of students who earned 70% or higher on all assessments measuring each ABET outcome in the user-given semester range. Semesters in the given range appear across the top; outcomes appear down the left side.

Overview Course Report: This report shows the percentage of students in each course which was active in the user-given semester range who earned 70% or higher on all assessments measuring each ABET outcome. Outcomes appear across the top; courses appear down the left side.

CORE Report: This report shows the percentage of students who earned 70% or higher on each ABET outcome associated with a user-given course offering, as well as a description of each ABET outcome. The outcomes that the offering measures appear down the left side. The percentage of students in the offering who performed at 70% or higher on all assessments that measured that outcome appears next. Finally, the text of the outcome appears.

Outcome Report: This report shows the percentage of students who earned 70% or higher in every course offering(s) asscioated with a specific user-given ABET outcome. Semesters in the given range appear across the top; those courses that measure this outcome appear down the left side.

Matrix Report: This report shows all ABET outcomes and all courses in AbOut and places an ‘X’ to designate that an outcome is associated with a specific course. All outcomes associated with all courses are across the top. All courses are listed down the left side.

Main Page: This component will use the information obtained from the login component. Its main function is to facilitate a way to navigate between the various report generation panes and provide info to users on the reports page functionality.

8.3.1 Processing for Report Component

Reports are processed within ReportsFormProcessor.php. This form processor makes use of ReportsModel.php, which pulls data directly from the database. Based on the user inputs for each report described above, the form processor performs the necessary calculations for each report. The form processor determines which report to generate based on the value of variables obtained via POST from the various report panes. For each report, the form processor gathers data from the following database tables:

Overview Outcome Report: Semester, Outcome, CourseOutcome

Overview Course Report: Semester, Outcome, Course

CORE Report: Course, CourseOutcome, Outcome

Outcome Report: Semester, CourseOutcome

Matrix Report: Outcome, Course, CourseOutcome

8.3.2 Report User Interface

Users may select the type of report desired from the tree under the Reports tab. Each report is configured via various drop-down boxes, which allow users to specify data such as semester range and program, as well as whether raw data should be displayed on the reports. Once the report is configured, the ‘Generate Report’ button may be used to show the report within AbOut, or the ‘Generate CSV’ button may be used to immidiently download a copy of the report in a .CSV file format. If the users selects ‘Generate Report’, users may also save a PDF copy of the displayed report with the ‘Save’ button located below the report.

Appendix A – Requirements Traceability Matrix

I doubt that it will be helpful to detail design elements to the level that this traceability matrix can be completed. Therefore, probably delete this eventually.

Requirements	**Requirement Description**	**SDD Ref.**