Accreditor Software Applications

Software Design Document

for Software Engineering Final Project

Prepared by: Jesus G. Villafranca
Jesus Jasso
Diana Orozco
Sandra Briseno

Instructor: Dr. Mahmoud K. Quweider

Version: 1.0 Approved

November 12, 2014

TABLE OF CONTENTS

Contents

Revision History

Version	Date	Author(s)	Description
Alpha 1.0	11/9/2014	JV, JJ, DO, SB	First Draft created

1 INTRODUCTION

1.1 Purpose

The purpose of this document is to describe the implementation of the Accreditor Application Software. The Accreditor Software Application is an all-in one application for colleges and universities that generates user specific reports based on various queries. These reports contain all the institution's information regarding its colleges, courses, faculty, staff, students along with its vision and mission for every said department. This allow to generate on the fly reports in order to meet standard compliance with the institution accreditation guidelines.

1.2 Scope

This document describes the implementation details of the Accreditor Application Software. The Accreditor Application Software has three major functions: they are to receive the data from the institution, display the data on screen for the user and generate reports. This document will not include contain prototype nor testing information as it not readily available at this point. However it will added later on to both the Software Requirement Document as well as the Software Design Document.

1.3 Definitions, Acronyms and Abbreviations

NetBeans IDE 8.0 – an integrated development environment for developing primary with Java programming language. We use NetBeans IDE to develop the Accreditor Software Application. It runs on Windows.

GUI – Graphical user interface. The primary medium the end-user interacts with the application software.

XML Database – An XML database is a data persistence software system that allows data to be stored in XML format. These data can then be queried, exported and serialized into the desired format. XML databases are usually associated with document-oriented databases.

Setters and Getters – A highly efficient form of programming that allows data to be encapsulated and self-contain within its class. The only way to access the data is to properly call the said function. It allows codes to be highly modular and crash resistant when bad or redundant data is encounter.

ANSI/IEEE Std. 830-1984 –This document is mainly drafted and design using the IEEE Recommended Practice for Software Requirements Specifications standard. Our Software Requirement Specification is standard compliant, as an extension, the Accreditor Software Application is also standard complaint with all IEEE and ANSI specifications and standards.

Customer – The University of Texas at Brownsville faculty and staff that will operate and interact with the software application. They are the end-user.

2 Design Overview

2.1 Description of Problem

The customer has requested the development of a software application that will allow them to handle the mass amount of data for the whole intuition and create on the fly-reports with specific queries on such data. In addition, the application should read an already built database as well as allow the user to input the data manually.

2.2 Class Structure

This system is designed to be as simple and modular as possible. It allows different programmers and coding to work on many aspects of the application individually. The initial design calls for the main class and the subclasses with different aspects of code self-contain within. The following is a breakdown of the Accreditor Software Application classes and its functions:

2.2.1 Colleges.java Class

A minor class that holds all the data pertaining to all the colleges. It queries the user on how many colleges exist in order to create various records. In addition, it has setters and getters that allow the data to be encapsulated and protects the data contents from outside classes.

2.2.2 Courses.java Class

A minor class that holds all the data pertaining to all the courses. It queries the user on how many course exist in order to create various records. In addition, it has setters and getters that allow the data to be encapsulated and protects its contents from outside classes.

2.2.3 Department.java Class

A minor class that holds all the data pertaining to all the departments. It queries the user on how many departments exist in order to create various records. In addition, it has setters and getters that allow the data to be encapsulated and protects its contents from outside classes.

2.2.4 Faculty.java Class

A minor class that holds all the data pertaining to all the faculty within a giving department. In addition, it holds the CV files which is a file format that stores resume like information. It queries the user on how many faculty exist in order to

create various records. Lastly, it has setters and getters that allow the data to be encapsulated and protects its contents from outside classes.

2.2.5 Program.java Class

A minor class that holds all the data pertaining to all the programs. It queries the user on how many programs exist in order to create various records. In addition, it has setters and getters that allow the data to be encapsulated and protects its contents from outside classes.

2.2.6 Student.java Class

A minor class that holds all the data pertaining to all the students. It queries the user on how many students exist in order to create various records. In addition, it has setters and getters that allow the data to be encapsulated and protects its contents from outside classes.

2.2.7 *OBJECT*Xml.java Class

Various minor classes exist that end with Xml on its name. These classes are essentially the data stored in a local XML database. This allows the main program to directly read, write and query the data all while protecting it and avoid any corruption on the Xml database. These classes are created at run time with the provided database information. In addition, data can be appended to the database when the end user enters record data manually.

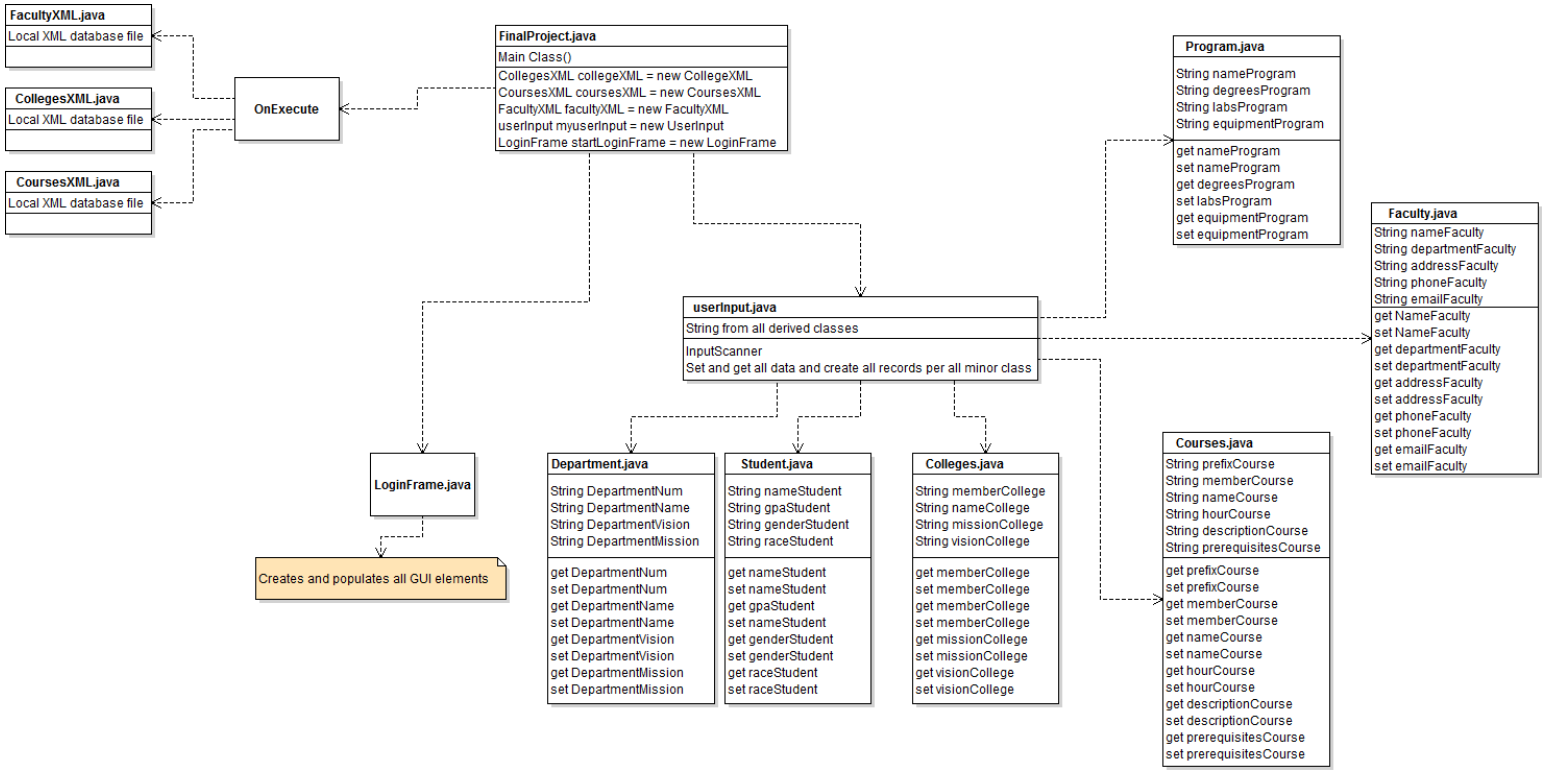
2.2.8 userInput.java Class

A major base class that calls all the other input classes into one major function. It allows the class to be call in and invoke from the program main section.

2.2.9 LoginFrame.java Class

A major base calls that create all the GUI elements. In addition, it reads from other classes in order to display data. Lastly, it has button listeners that will execute code depending on the users' interaction with the application.

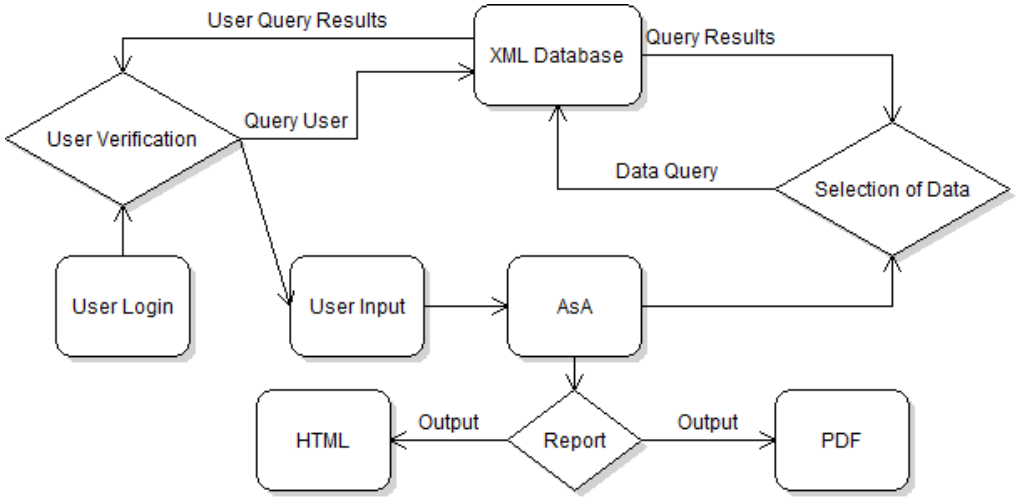
2.2.10 Class Structure Diagram



3 System Operation

3.1 Accreditor System Software Schematic Diagram

An overall schematic of the application main logical components. The overall diagram is shown below:



3.2 Major Workflow Outline

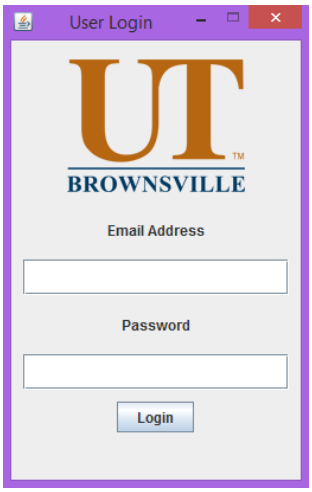
The following is an outline explaining and elaborating on the main functions of the Accreditor Software Application:

- User Verification:
  End user enters email address and password for verification and authentication.
- XML Database
  Local XML base database is created using predefined data. This allows the main program to have easily read and write to the database file. In addition, it will also be able to append to the file in order to add more records as per the user request.
- Selection of Data
  Allows to perform advance queries on the local XML database. In return, it will display results base on those queries both on the end users' screen and when generating reports
- Report
  It will generate a report based on the queries or lack thereof and will display the data either on the user or generate either a PDF or HTML file

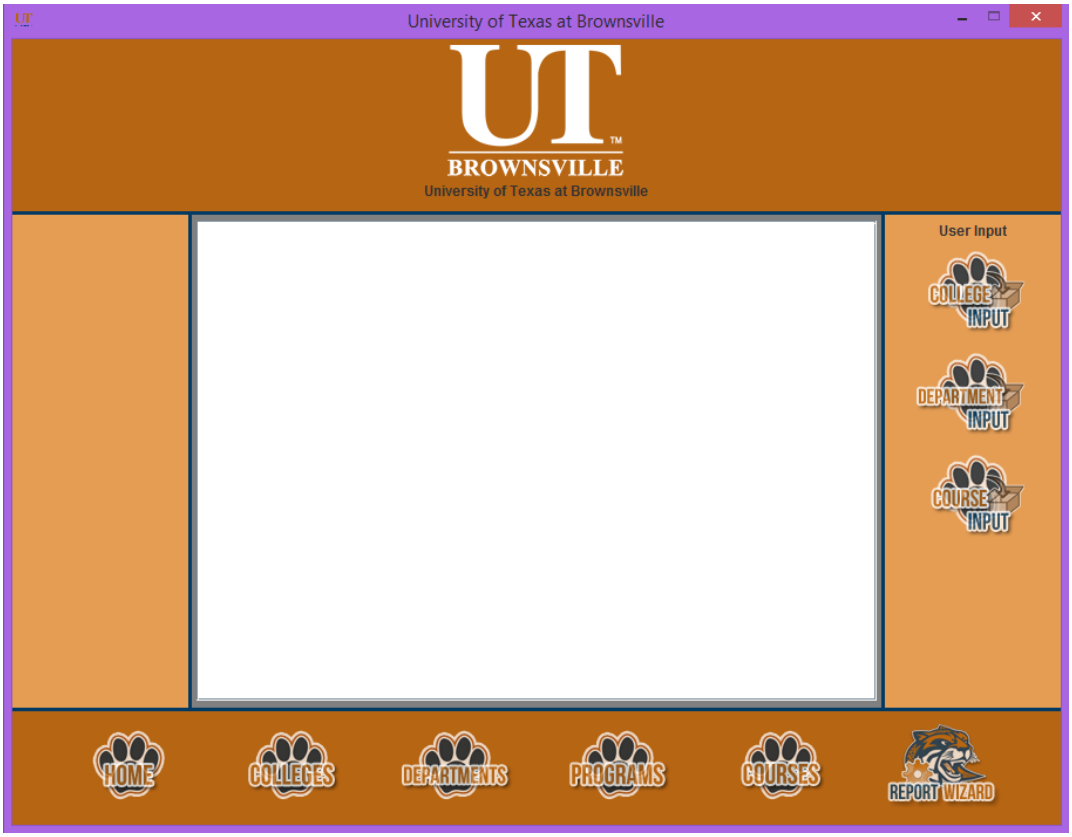
3.3 Interface

The interface will include various dialog box along with buttons that will allow the user to interact with the application software. Here are the screenshots of some of the GUI elements already developed:

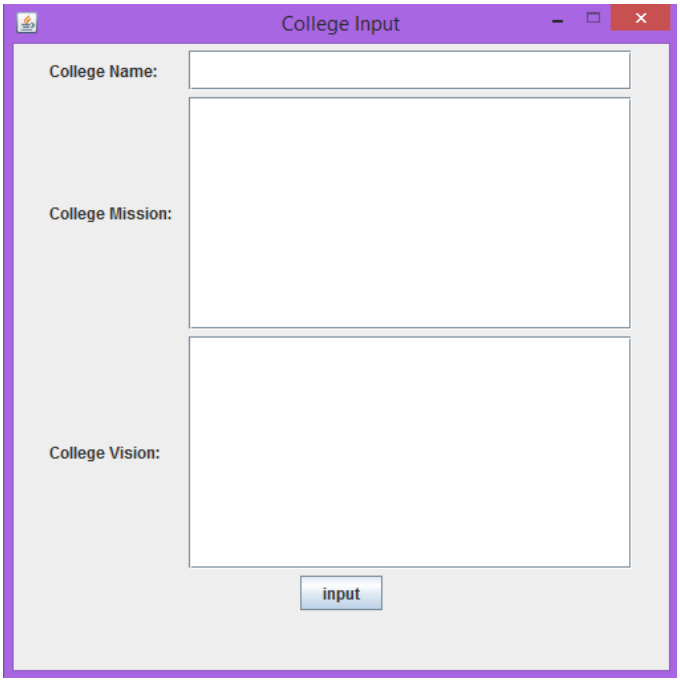
Accreditor Software Application Splash Screen



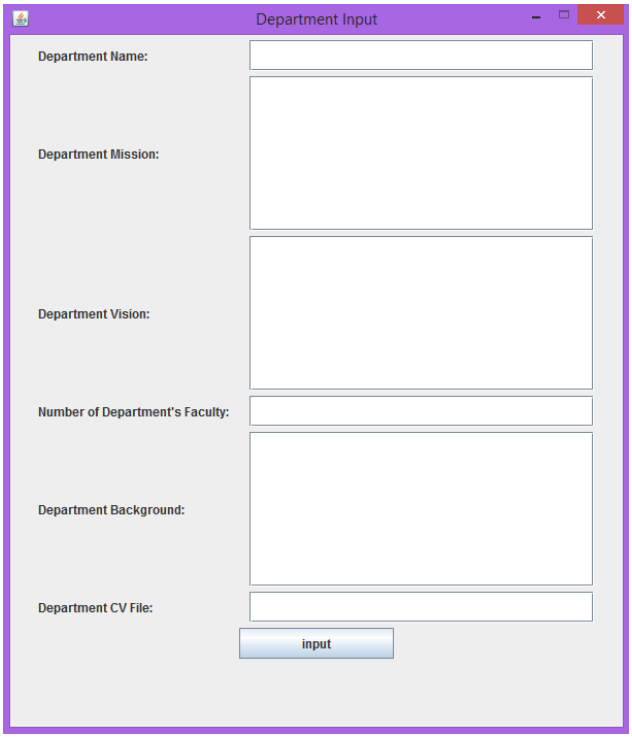
Initial Dialog Box authenticating credentials



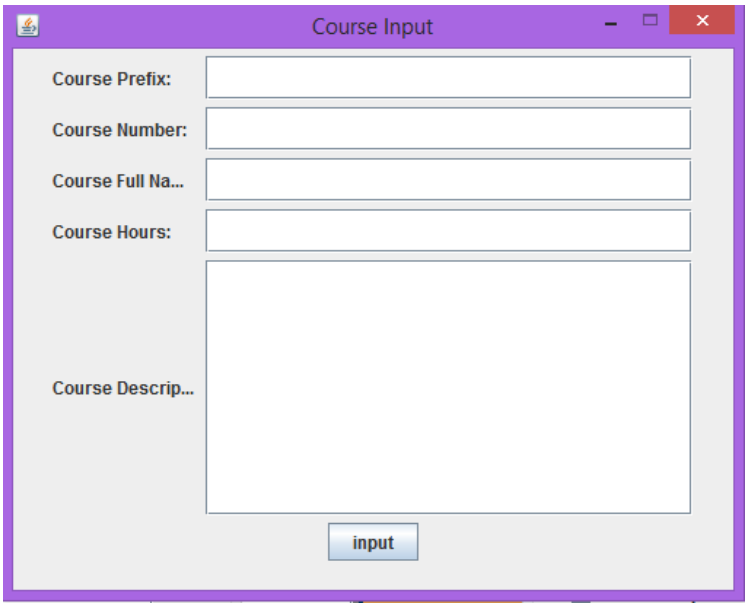
Main Screen that will allow the user to access the data on the local database as well as add new data using the following dialog boxes:



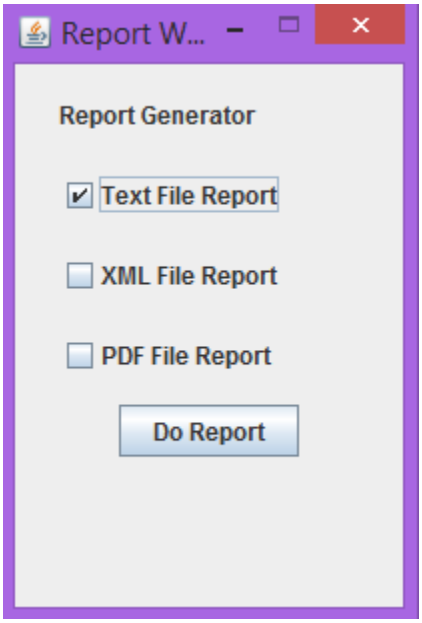
College Input Dialog Box



Department Input Dialog Box



Course Input Dialog Box



Report Generator Dialog Box

4 TIMELINE

The following table is a tentative timeline of the individual and group contributions to the project. It is tentative and subject to change. As new functions are added/remove/improved to the project, this table will be updated accordingly:

Meeting Number	Date	Medium	Details
1A	10/26/2014	Face to face	• Plan and discuss project • Set team roles and functions.
2A	10/29/2014	E-mail/Phone	• Created GitHub accounts, created prototype code and upload to GitHub. • Discuss on using XML local database. • Discuss on GUI design and composition • Combined all SRS parts into SRS document final, proofread and upload to GitHub.
3A	10/31/2014	E-mail	• Discuss on current design and future design changes
1B	11/2/2014	Face to face	• Added code to create local XML database
2B	11/9/2014	Face to face	• Added code to create GUI screens • Improve local XML database implementation
3B	11/16/2014	Face to face	• Pending: Have fully working prototype • Pending: Test local XML functionally: to be able to read/write/append to
4B	11/23/2014	Face to face	• Pending: Read from database, output the data to main screen Pending: Perform advance queries on database, output the data to main screen
5B	11/26/214	Face to face/Email/Phone	• Pending: Read from database, output the data to main screen • Pending: Perform advance queries on database, output the data to main screen
6B	11/30/2014	Face to	• Pending: Generate reports to screen • Pending: Generate reports to HTML/PDF files
1C	12/1/2014	Face to face/Email/Phone	• Pending: Check, update and improve Software Requirement Document to match final version • Pending: Check, update and improve Software Design Document to match final version • Pending: Thoroughly test software, make final revisions
2C	12/2/2014		• Pending: Deliver fully working software that satisfy all requirements Pending: Deliver full documentation