A web application to assist in preparing for ABET accreditation

by

Saveri Pal

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Simanta Mitra, Co-major Professor
Gurpur Prabhu, Co-major Professor
Carl K. Chang

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this thesis. The Graduate College will ensure this thesis is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2021

DEDICATION

Dedicated to my parents and my little brother.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

ABET	Accreditation Board for Engineering and Technology
CRUD	Create Read Update Delete
JSON	JavaScript Object Notation
Canvas LMS	Canvas Learning Management System

ACKNOWLEDGMENTS

I would like to thank my co-major professors, Dr. Simanta Mitra and Dr. Gurpur Prabhu, and my committee member, Dr. Carl Chang, for their guidance and support throughout the course of this research.

In addition, I would also like to thank my friend, Tenson Cai, for his insightful feedback while debugging issues.

I want to offer my gratitude to Ananda, for being my pillar of support throughout my journey in computer science, without who, this thesis would probably not have been possible.

ABSTRACT

ABET accreditation is a certification of an educational program's quality and ensures that a program continually meets specific criteria. The process of creating an ABET accreditation report is burdensome and time consuming. It requires several pieces of information from multiple sources to be collected manually and included in the report. The current practice is error-prone and requires the effort of an entire faculty and administrative staff. While there have been previous efforts made to automate the evaluation of outcome assessments, little can be found on automating the data collection process. This work presents a web-based application software tool to automate data collection and report generation efforts in preparing an ABET accreditation document. Requirements for this application were gathered after several meetings with stakeholders of the ABET accreditation process. A combination of web crawling, data parsing and retrieving data from third-party applications such as Canvas LMS via API calls, have been used to gather information that is usually performed manually. Information collected include courses offered by the department, assignments of each course, student submissions, and their scores, among others. The web application tool features an intuitive user interface that allows users to add courses and assignments manually or pull them from Canvas LMS, add outcomes to courses and map them to ABET outcomes, and generate course reports and tables with one click of a button. The tool has the potential to automate a large part of the accreditation process, reducing manual effort by a considerable amount.

CHAPTER 1. INTRODUCTION

1.1 Background

Accreditation is a form of quality control that ensures educational institutions satisfy certain standards. ABET is a non-profit, non-governmental agency that accredits programs in applied and natural science, computing, engineering, and engineering technology [5]. Accreditation is based on a list of criteria that programs must meet to attain the accredited status. An accredited status indicates that a program offered by an educational institution meets global standards, and graduates of such programs are well prepared to enter the workforce. Today, ABET accreditation is sought after by programs worldwide and conveys the confidence of quality. ABET supports innovation and quality in education and accreditation increases trust and credibility of a program [5][7].

The ABET accreditation criterion for computing programs defines six outcomes, 1 through 6, that denote a specific skill that a student must possess when she or he graduates from a program. The courses taken by the student develop these knowledge and skills in her or him. A student in an educational program must be continually assessed throughout the duration of the study and the achievement of these outcomes be documented.

1.2 Motivation

The process of keeping a program accredited requires tedious documentation and presents unnecessary burden on staff [6]. It requires keeping track of fulfilling the ABET outcomes for all the mandatory courses offered by a program. Each course attempts to meets a subset of all the six outcomes and this is assessed via assignments, quizzes, exams, and projects completed by students. The degree to which a student achieves these outcomes are measured quantitatively from the scores of these assignments. All the information is included in a report that is evaluated by an ABET evaluator. The procedure of bringing together this report is time-consuming, expensive, and error prone when done manually.

1.3 Solution

We propose a web application that aids in creating and managing an ABET accreditation process. It is semi-automatic. The application pulls course information such as assignments, their grades from Canvas LMS and stores them in a database. It can fetch course information such as course description, credits, prerequisites from a department website. It can be used to store all courses offered by a department for their ABET accreditation and generate reports of all the courses with the click of a button. There is a course vs. outcome table that lists outcomes fulfilled by each course and indicates if a given outcome has been updated with a relevant assignment or not. This allows for easy identification of empty fields that should be filled before an ABET evaluator evaluates it. In addition, courses and assignments can be added manually to the application by filling out easy and intuitive input forms.

1.4 Organization

This document is organized as follows from here on : Chapter 2 discusses prior work on this topic, and in Chapter 3 the design and architecture of the application is described. Chapter 4 shows results of the application and Chapter 5 discusses future work on this topic.

CHAPTER 2. REVIEW OF LITERATURE

This section reviews existing automation efforts of the ABET accreditation process.

Essa et al. [1] developed a web-based software tool named ABET Course Assessment Tool (ACAT), that facilitates collection of data required for ABET and automatically generates standardized reports. The authors discuss the design, ease of use of the tool along with its implementation and operational details. The tool features an SSL encrypted authentication system and discusses features that make data entry easy, allowing copy-pasting from excel into ACAT. The four distinct components of the tool are authentication, data entry, report generation and different views for different users. Data entry is available via web forms with users having the ability to start, save, edit new or existing reports, and generate PDF of reports when complete. Both direct (exam, assignments, grades) and indirect (surveys) assessments are entered into the system. It allows copy pasting of data into the tool that adds flexibility to the data entry process. However, there is no detail of the data entry process described in the paper. The user interface contains a navigation panel to make different components of the tool accessible to the user. The user interface is made up of HTML web pages that are grouped into multiple layers that form the graphical user interface, and database interaction and web pages accessible to different kinds of users. The authors developed a functionally working prototype of ACAT and tested some features.

Course and Student Management System (CSMS) by Rahman et al. [2] is another web-based tool that aims to automate the learning objectives and program outcome assessment involved in the ABET accreditation process. The system aims to create a course assessment matrix and help faculty to assess students in an easy manner. The paper discusses an articulation matrix that maps each course outcome to ABET student/program outcomes. It uses impact values to determine how well a course is fulfilling the ABET criteria. The tool helps faculty members to build matrix and course reports and display assessment results graphically. Other features of the tool are adding users to the application, course details, assignment information and grades.

The web-based program outcome assessment (POA) system by Khwaja [3] is implemented to replace the manual effort involved in assessing ABET program/student outcomes. It describes the automation process and presents a prototype of this web-based tool. The tool adopts a direct-grade based student assessment system where rubrics for each program outcome can be defined in web-based interfaces. Assessments can be mapped to each rubric and assignment grades can be entered into the system by instructors using these interfaces. The grades and the rubrics are then used to automatically compute program outcomes. However, the data entry into the system described is mostly manual. Adding course information such as outcomes, assignments, grades are all done manually. Adding faculty profiles to the system is also manual. The tool automates the outcome evaluation part where it considers all the information entered in the system and generates program outcome rubric assessment results in color coded tabular and graphical forms.

Wang and Tang [4] uses Canvas LMS APIs to fetch course data from it and uses the information to help departments with multiple majors that need to seek accreditation from ABET CAC (Computing Accreditation Commission) and ABET EAC (Engineering Accreditation Commission). Canvas LMS does not allow downloading of individual rubric criteria of an instrument, hindering analysis to a more granular level. There is also no information to identify a student’s major, making data separation for different academic programs difficult. Professors do not share this information on Canvas LMS in fear of violating the FERPA (Family Educational Rights and Privacy Act).

The authors develop a web application, Multi-Program Assessment System (MPAS), that works on top of Canvas LMS to expand its functionalities and resolve the existing difficulties on data collection, separation, sharing, integration, and security. This helps faculty to assess student performance and generate instrument reports. The reports generated only show computed statistics and no other detail to safeguard data privacy.

While our ABET tool has several similarities with the above discussed works, we use the ideas to automate the manual effort involved in generating the ABET Self-Study report along with storing faculty information and providing an easy-to-use web interface for entering and reviewing the report. We use Canvas LMS APIs to pull course data from Canvas and then generate reports automatically using report templates stored in the application. Automated data entry is a feature that is missing from most available tools. While Wang and Tang does use Canvas APIs to retrieve data from Canvas LMS, their effort is mostly directed at calculating course outcome assessments. We have sought to concentrate on the automated data retrieval capabilities in our tool and examine the extent to which it can ease the process of preparing an ABET accreditation report.

Table 2.1 Features available in current ABET accreditation assistance tools

Features	**ACAT** **2010**	**CSMS** **2016**	**POA** **2018**	**MPAS** **2019**
Report generation	Y	Y	N	N
Automated data entry	N	N	N	Y
Navigation panel	Y	Y	Y	N
Outcome assessment	N	Y	Y	Y
Customize outcome description	Y	Y	Y	Y

CHAPTER 3. DESIGN AND IMPLEMENTATION

This chapter provides a detailed overview of the web based ABET accreditation support tool. The goal of the tool is to reduce manual effort required in creating an ABET accreditation report by an educational department of a university. This tool aims to create an application that automate data gathering by using a combination of web APIs, web crawling and data parsing techniques, creating a central repository where all the courses offered by the department is stored, and automate report generation of the various form required in ABET accreditation.

3.1 Requirements Gathering

Features of the application were decided after careful consideration and multiple meetings with the stakeholders, who create the ABET report every year on behalf of the department. ABET Self-Study reports from previous years were considered to create a list of the most important requirements of the application. Mock-up screens were created, and they were reviewed by the stakeholders. This was done in several iterations before the final look of the application was decided.

3.2 Features

The application is currently a prototype and can be viewed as a proof-of-concept. Crucial requirements are implemented that make up the essence of an application built to support the ABET accreditation procedure. The application works in conjunction with Canvas LMS, a tool that is used by Iowa State University to store course information such as assignment, exam, student enrollment, submissions, and grades. It pulls pertinent course information from Canvas via API calls. Courses and their assignments can be added into the application with hardly any manual effort.

Outcomes fulfilled by a course forms a major part of an accreditation report. This app allows adding outcomes to a course by specifying what assignment fulfills which outcome. It offers easily readable tabular views of all the courses that fulfill a certain outcome.

The application features easy and automatic report generation. Multiple reports can be generated by specifying only the template of the report with empty form fields. Completed and filled PDF reports are generated ready to be included in the accreditation document. The sections below discuss the features of the application in further detail.

3.2.1 Course

There are two ways by which a user can add a course to the application: (1) by manually inserting information about the course into a form and, (2) by providing only a Canvas course id and thereby prompting the application to automatically retrieve relevant course information from Canvas using the course id. The second option is a convenient feature that automates the practice of entering course details manually into the report. With the information automatically saved into the database, it can be re-used to generate multiple reports.

Courses stored in the database can be updated or deleted. There are form fields that display current information and allow changes that are immediately updated to the database with the click of the Update button.

All courses stored in the database are listed on the Course List page that comes with a search bar. Courses can be filtered by complete name or partial name or number. This is useful in cases where there are hundreds of courses saved in the application. A future work in this area could be to add filters that find courses grouped by year or semester they are offered.

The course page lists all the courses in the application. There is a search bar that facilitates easy filtering of courses by partial names. Selecting a course from the list displays basic information about the course in a panel on the left. This allows easy identification of a course when there are various courses with similar names. Clicking a course on the list, displays primary information of the course on a right-side panel with a button to Edit the course. This allows the user to identify the correct course before choosing to see or edit its details on a different page.

3.2.2 Assignment

Multiple assignments can be added to each course in the application. There are two ways by which a user can add an assignment to a course: (1) Add assignments to a course manually by inserting assignment information into a form and, (2) Retrieve all related assignments of the course from Canvas. When a course is added to the application with a canvas course id, it signals the application to retrieve all course information along with its assignments from Canvas. It retrieves canvas assignment id, name, description, and all student submissions to the assignment.

Assignments stored in the database can be edited or deleted. Deleting a course deletes all its assignments from the database. There are input fields provided which allows easy updates to an assignment. Mean, max and mean scores of all submissions are calculated and displayed as read-only information. This is intended to allow users to effectively decide which assignments to include in the accreditation report.

The application allows the user to select an ABET outcome for an assignment via a dropdown menu with the six ABET outcomes. In future, if a new outcome is added to the application, the dropdown will auto-update to include the new outcome number.

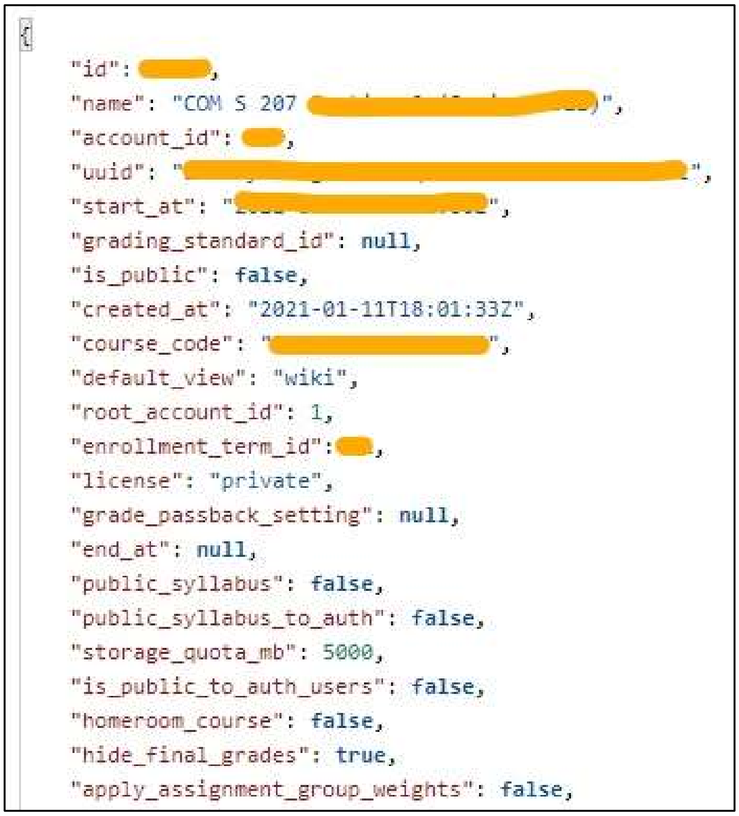
3.2.3 Outcome

Learning objectives and outcomes form an important part of any accreditation report. Each course fulfills a few outcomes that are measured from the outcomes fulfilled by its assignments and exams. The application allows the user to select an outcome for each assignment. Outcomes can be created by a user by adding a name, description, and number to the outcome. Once an outcome is created, it is available to be selected for an assignment. Outcomes from all assignments are combined to get the outcomes fulfilled by a course. In the current version of the application, an assignment can be assigned only one outcome, while courses can have multiple outcomes.

There is a course-outcome table that lists all courses and their respective outcomes. It presents an efficient way to visualize the outcomes fulfilled by all the courses of a department.

3.2.4 Canvas Data Retrieval

Canvas provides APIs to retrieve data. Bearer tokens generated in the Canvas application can be used as authorization to retrieve desired information from Canvas. The data returned is in JSON (JavaScript Object Notification) form. This application fetches courses and their respective lists of assignments from Canvas. Scores of all students for each assignment is also



```
{
    "id": [redacted],
    "name": "COM S 207 [redacted]",
    "account_id": [redacted],
    "uuid": "[redacted]",
    "start_at": "[redacted]",
    "grading_standard_id": null,
    "is_public": false,
    "created_at": "2021-01-11T18:01:33Z",
    "course_code": "[redacted]",
    "default_view": "wiki",
    "root_account_id": 1,
    "enrollment_term_id":[redacted],
    "license": "private",
    "grade_passback_setting": null,
    "end_at": null,
    "public_syllabus": false,
    "public_syllabus_to_auth": false,
    "storage_quota_mb": 5000,
    "is_public_to_auth_users": false,
    "homeroom_course": false,
    "hide_final_grades": true,
    "apply_assignment_group_weights": false,
```

Figure 3.1 Sample JSON response of a course object retrieved from Canvas LMS

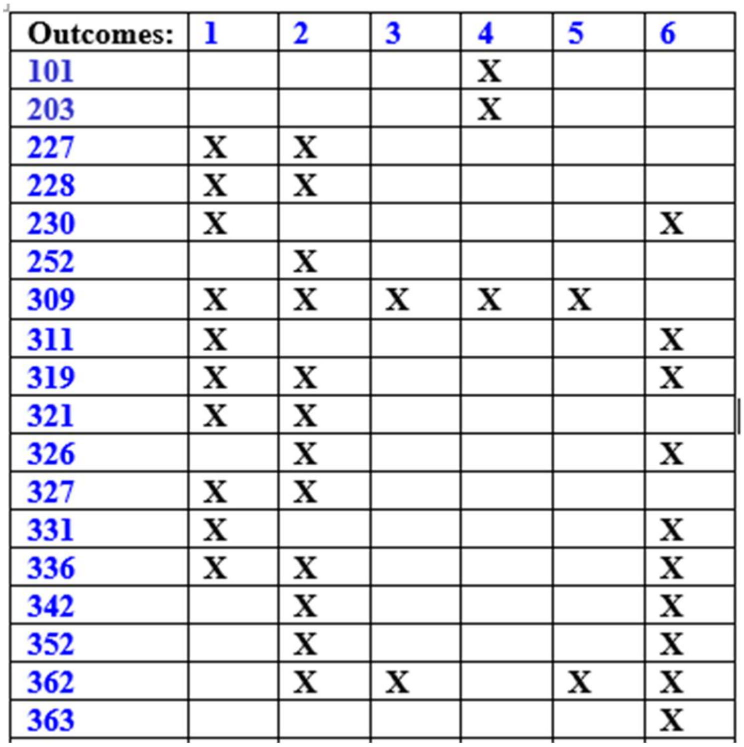
fetched from Canvas LMS to calculate mean, maximum and minimum score of each assignment. The JSON received from Canvas LMS is parsed to extract the required information and update it to the database.

3.2.5 Report Generation

There are various types of reports included in an accreditation document. Each depicts a facet of the quality of education provided by a department. These reports are time-consuming to create when done manually and are prone to mistakes. For example, a department offers 20 courses, and course detail reports need to be generated for each course. A major effort goes in copy-pasting information from different sources into a template form. Since the job is repetitive, auto generating the reports saves a lot of time.

ABET Self-Study report is a quantitative and qualitative assessment of a program's strengths and limitations. It provides a comprehensive overview of a program and guides the review team from ABET throughout its evaluation [5]. Course reports are included in Appendix-A of the ABET Self-Study document. ABET provides templates of this form. We use this course report template as a sample to depict the document generation automation capabilities of the application. This report provides general details of a course offered by the department namely, course name, number, credits, contact hours, instructor name, prerequisites, a brief description, and the outcomes fulfilled by the course. The information required to complete this information comes from not one but multiple sources. Normally, a manual effort includes copying information from three or four different sources and pasting them in the empty fields of this template. Then this is repeated for all the degree courses offered by the department.

There are many challenges faced while creating the course report. One has to regularly check the course catalog website for updates and make sure the information on the course report is up to date with the catalog. The outcome of each course is determined by the faculty teaching

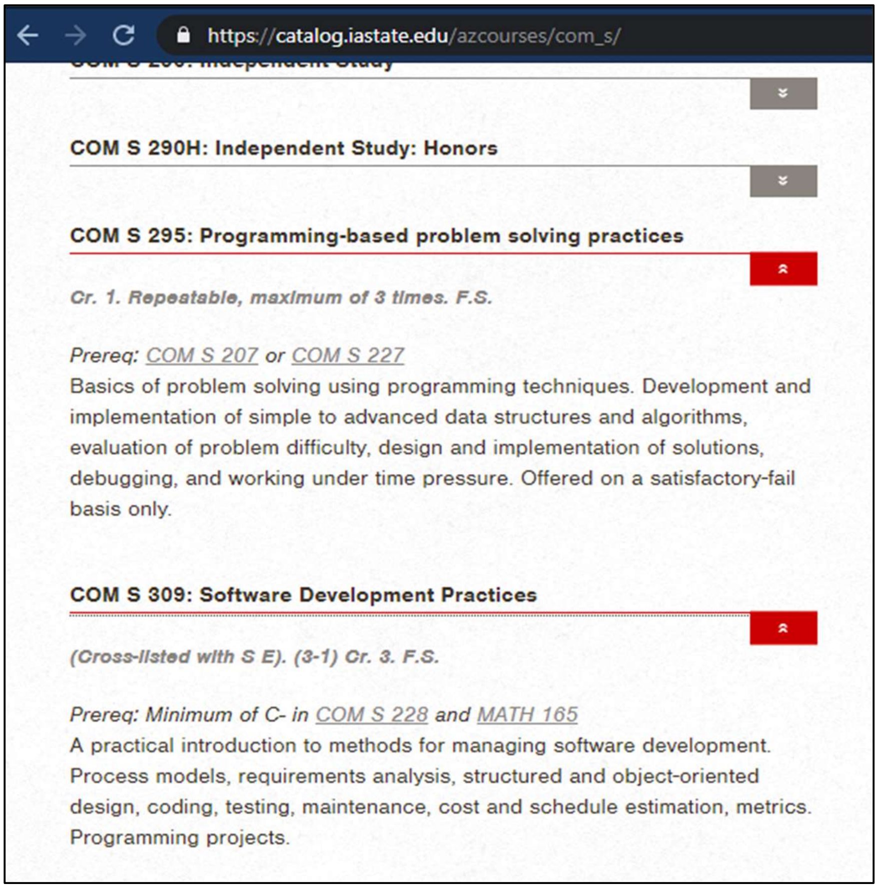


Outcomes:	1	2	3	4	5	6
101				X		
203				X		
227	X	X				
228	X	X				
230	X					X
252		X				
309	X	X	X	X	X	
311	X					X
319	X	X				X
321	X	X				
326		X				X
327	X	X				
331	X					X
336	X	X				X
342		X				X
352		X				X
362		X	X		X	X
363						X

Figure 3.2 Partial snippet of a sample course-outcome matrix

the course. It is a time-consuming effort to reach out to each faculty and send them timely reminders requesting them to send the course outcomes. After collecting outcomes, a course-outcome matrix is created for all the courses. This is another monotonous job that requires a lot of diligence. Figure 3.2 shows a course-outcome matrix that is created manually. Every row in this matrix denotes a degree course offered by the department and is used to complete the outcome table in the ABET Self-Study Appendix-A course report. Moreover, strict formatting must be maintained while completing each form which is often messed up while copy-pasting data manually into the form. The web-application addresses these challenges by generating and updating the course-outcome matrix automatically every time an outcome is assigned to a course.

This application applies a web crawler to gather relevant information about a course from the department course website and stores them in a database. Figure 3.3 shows the course listings on the course catalog of the department of computer science. The web crawler extracts required course information from this website. This eliminates the effort required to manually check the catalog website from time to time. Other required information to complete the form is also acquired and saved in separate documents. The application parses all the documents and tables to compete the template form. It adds relevant information to relevant fields of the template and generates PDF reports for each course. It can generate hundreds of reports at a time depending on CPU capacity. The whole procedure is completed in a few seconds, thus saving time, and eliminating considerable manual effort.



COM S 290H: Independent Study: Honors

COM S 295: Programming-based problem solving practices

Cr. 1. Repeatable, maximum of 3 times. F.S.

Prereq: COM S 207 or COM S 227
Basics of problem solving using programming techniques. Development and implementation of simple to advanced data structures and algorithms, evaluation of problem difficulty, design and implementation of solutions, debugging, and working under time pressure. Offered on a satisfactory-fail basis only.

COM S 309: Software Development Practices

(Cross-listed with S E). (3-1) Cr. 3. F.S.

Prereq: Minimum of C- in COM S 228 and MATH 165
A practical introduction to methods for managing software development. Process models, requirements analysis, structured and object-oriented design, coding, testing, maintenance, cost and schedule estimation, metrics. Programming projects.

Figure 3.3 Course listings in the Computer Science course catalog website

Future work in this area would be to expand the automated report generation to all the different types of reports necessary in an ABET study, such as faculty forms, course forms etc.

3.3 Architecture

The application is made up of three main interacting components – front-end, back-end, and database. It comes with Create Retrieve Update Delete (CRUD) capabilities. The front-end is made up of ReactJS and Bootstrap. The back end consists of Spring Boot with a MySQL database. The back-end server makes API requests to Canvas LMS to retrieve information, sends and receives data from front-end, and updates the MySQL database. Figure 3.4 describes the architecture of the application.

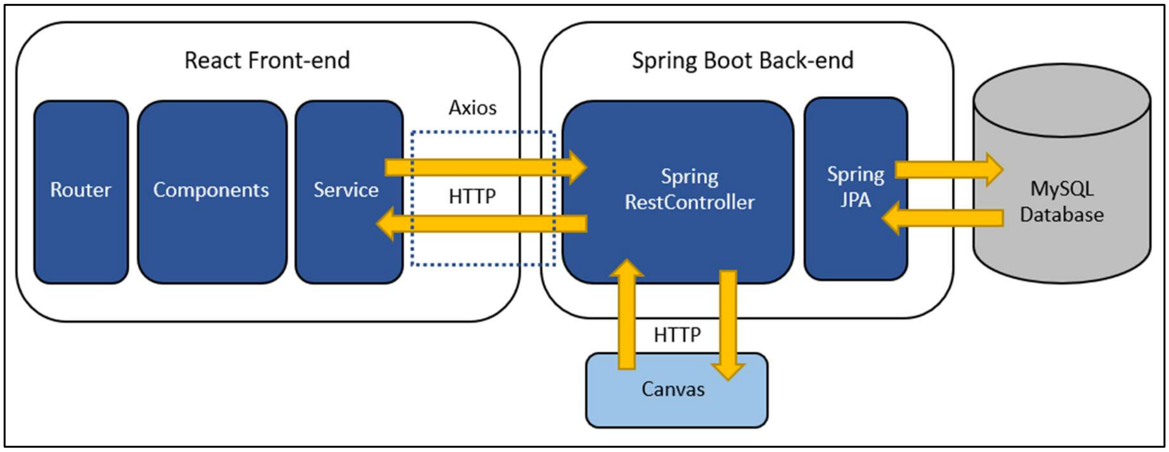


Figure 3.4 Application architecture

The front-end client side consists of React components, router, and services. The data is displayed using React components. React router is used to navigate the pages of the application and React services uses the Axios library to send and retrieve HTTP requests to the server.

The back-end server side is in Spring Boot and follows the Spring Model-View-Controller (MVC) structure. It uses REST APIs to interact with front-end and to retrieve data from Canvas. Each model has its own controller.

The database of the application is MySQL and all interactions with the database is done using Spring JPA.

3.3.1 Server Side Back-end

The server side consists of data models, controllers, and repositories. Models in spring boot define an object, its constructors, and variables. The models that correspond to tables in the database are annotated accordingly. Each instance variable that corresponds to a column in the database table is also annotated accordingly and one-to-many and many-to-one relationships are also specified. The three basic data models of the application are Course, Assignment, and Outcome. They are entities that have corresponding tables in the database. The entity relationship between the models can be seen in Figure 3.5.

Table 3.1 Sample APIs exported by spring boot for the course data model

Methods	URLs	Actions
GET	/abetapi/courses	Retrieves all courses
GET	/abetapi/courses/:courseId	Retrieves a course by id
POST	/abetapi/courses	Creates a new course
PUT	/abetapi/courses/:courseId	Updates an existing course by id
DELETE	/abetapi/courses/:courseId	Deletes an existing course by id

The controllers have request mapping functions for RESTful requests, for example *getAllCourses()*, *getCourseById()*, *createCourse()*, *updateCourse()* and *deleteCourse()*. Repositories are interfaces that extend the JpaRepository class for custom search methods and CRUD functions, such as *findAssignmentsByCourse()*. The JpaRepository comes with some functions that can be used without implementing them, such as *findById()*, *save()*, *findAll()* etc. In addition to that, the *application.properties* file contains configurations for JPA, Hibernate and database. The *pom.xml* file contains dependencies for Spring Boot and MySQL. Technologies used to build the server are Java 11, Spring Boot 4, Maven, MySQL.Table 3.1 shows APIs for the Course data model. In addition to that, the *application.properties* file contains configurations for JPA, Hibernate and database. The *pom.xml* file contains dependencies for Spring Boot and MySQL. Technologies used to build the server are Java 11, Spring Boot 4, Maven, MySQL.

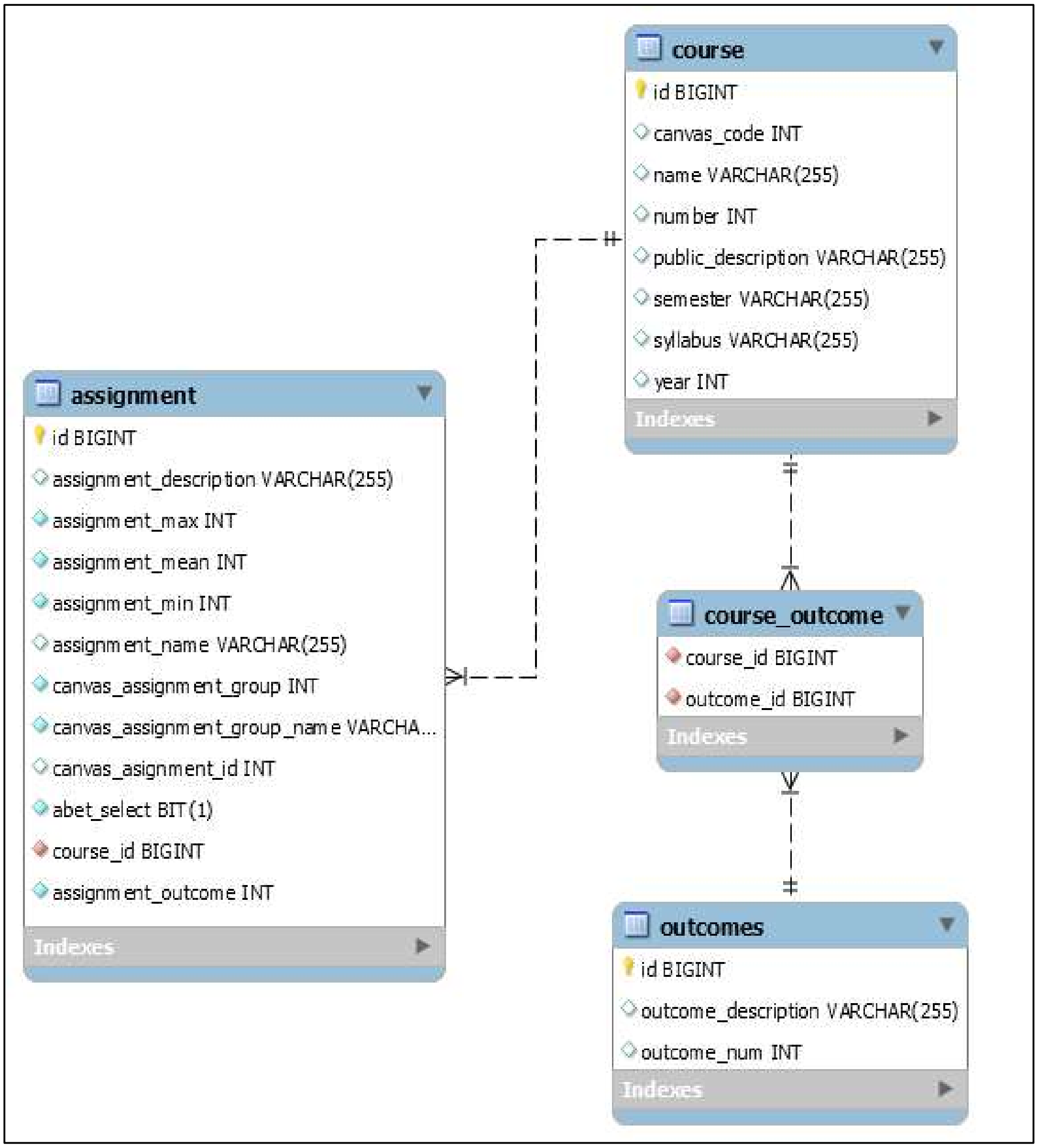


Figure 3.5 Entity-relationship diagram of the database

3.3.2 Client Side Front-end

The client side of the application is built using React.js and is made up of three main components namely, router, react components, and services. As shown in Figure 3.6, the Application component consists of the Router, which is a collection of navigational components and is responsible for specifying the path to each page of the web-application. There is a navigation bar that provides quick links to each page. The components are functions that return HTML elements via their *render()* method. *Course*, *CourseList*, *CourseAdd*, *AssignmentList, Assignment, AssignmentAdd, Outcome and DocGeneration* are the major components of this application and render pages that display various data from the database or contain input forms to edit, update and delete data from database.

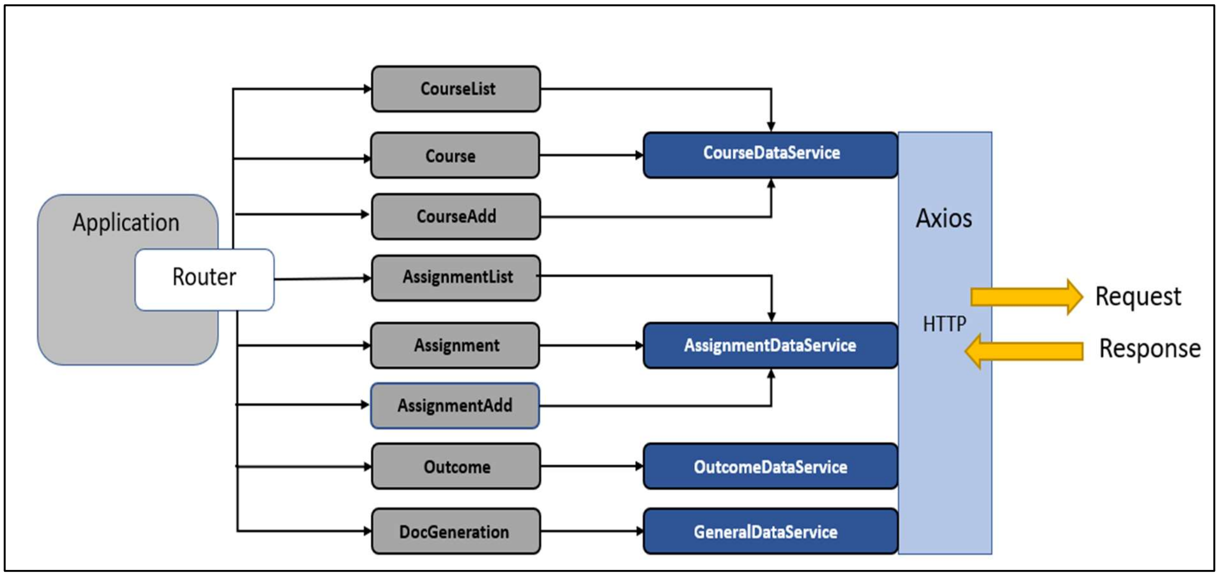


Figure 3.6 Front-end layout

The transfer of data between front-end and back-end takes place via HTTP requests and responses. The components call the data services such as *CourseDataService*, *AssignmentDataService* or *OutcomeDataService* to make those HTTP requests to the back-end and receive responses back. The technology used to build the client side is React 17, Axios and

Bootstrap. The services contain methods that specify URLs for API calls to initiate GET/POST/PUT/DELETE requests to the back-end. The data sent from the back-end is also received in the services. Separate services are used for the Course, Assignment and Outcome components to increase the readability and modularity of the code.

CHAPTER 4. RESULTS

The effort required to create an ABET accreditation report is multi-faceted and difficult to quantify. It typically takes three years and the effort of the entire department to gather relevant information and compile the report. This section discusses the features implemented in this application to reduce the manual involvement necessary to generate the accreditation report.

4.1 Features

The web-application offers features to reduce the manual effort involved in creating an ABET accreditation report. It contains a database to store course information from different semesters and compare the outcomes achieved. It features a navigation panel for easy access to the major web-pages of the application.

Table 4.1 Features of the application

Course	Course list and search/filter option
	Create, retrieve, update, and delete a course
	Add multiple assignments to a course
	Add multiple outcomes to a course
	Automated fetching of course
Assignment	Assignment list in a table
	Create, retrieve, update, and delete an assignment
	Add an outcome to an assignment
	View mean, max, min score of each assignment
	Automated assignment creation and grade retrieval
Outcome	Create, retrieve, update, and delete an outcome
	Course versus outcome table
Canvas API	Fetch course details
	Fetch assignment details
	Fetch student submissions
Report generation	Automated information retrieval from various sources
	Auto-generate multiple PDF reports at a time

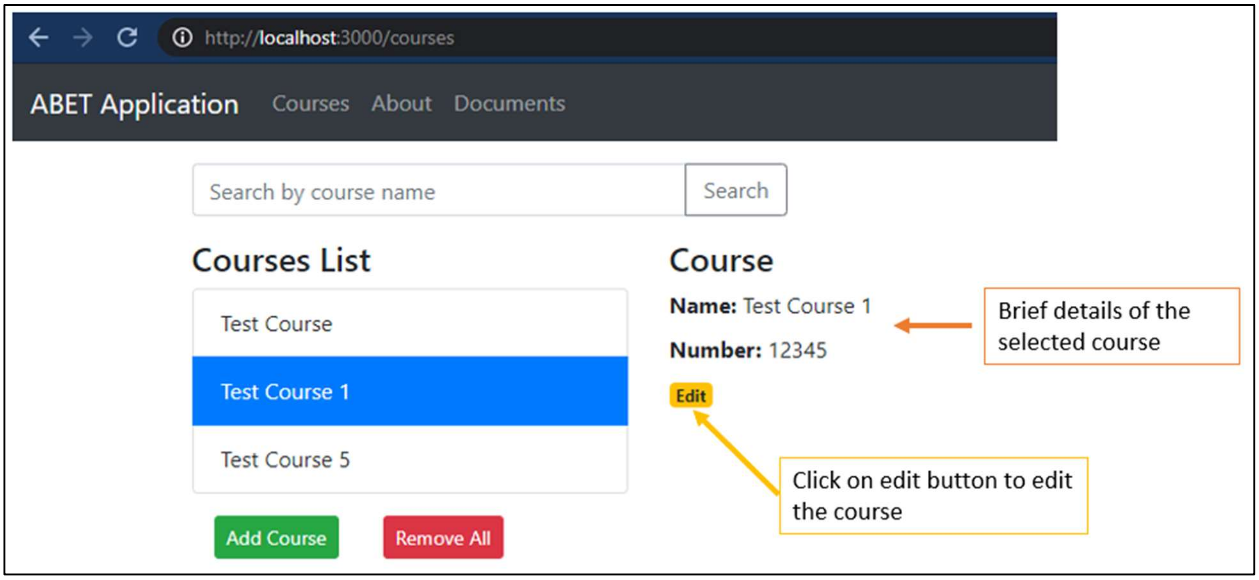
Table 4.1 lists all the features implemented in this version of the application, and Table 4.2 shows a feature comparison of this web-application ABET accreditation tool with the previously available similar tools ACAT [1], CSMS [2], POA [3], and MPAS [4]. The first three tools require the user to manually enter data into the system via input forms, while the last tool introduces automatic data retrieval from Canvas LMS. Our ABET assistance tool further explores the idea of automatic data retrieval by incorporating web crawling and document parsing in addition to automatic data retrieval from Canvas LMS. This lowers the amount of effort spent in copy-pasting data from multiple sources into the application.

Table 4.2 Feature comparison with other available tools

Features	ACAT 2010	CSMS 2016	POA 2018	MPAS 2019	Our Tool
Report generation	Y	Y	N	N	Y
Automated data entry	N	N	N	Y	Y
Navigation panel	Y	Y	Y	N	Y
Outcome assessment	N	Y	Y	Y	N
Customize outcome description	Y	Y	Y	Y	Y

4.2 Application Webpages

This section contains images of various web pages of the application. Figure 4.1 shows the course list page where all the courses stored in the application are listed. It features a search bar that allows filtering of courses by partial names or numbers. Figure 4.2 shows the course detail page. The information of a course can be edited or updated from Canvas on this page. It also contains a list of all the assignments associated with the course.



Courses List

- Test Course
- Test Course 1
- Test Course 5

Course

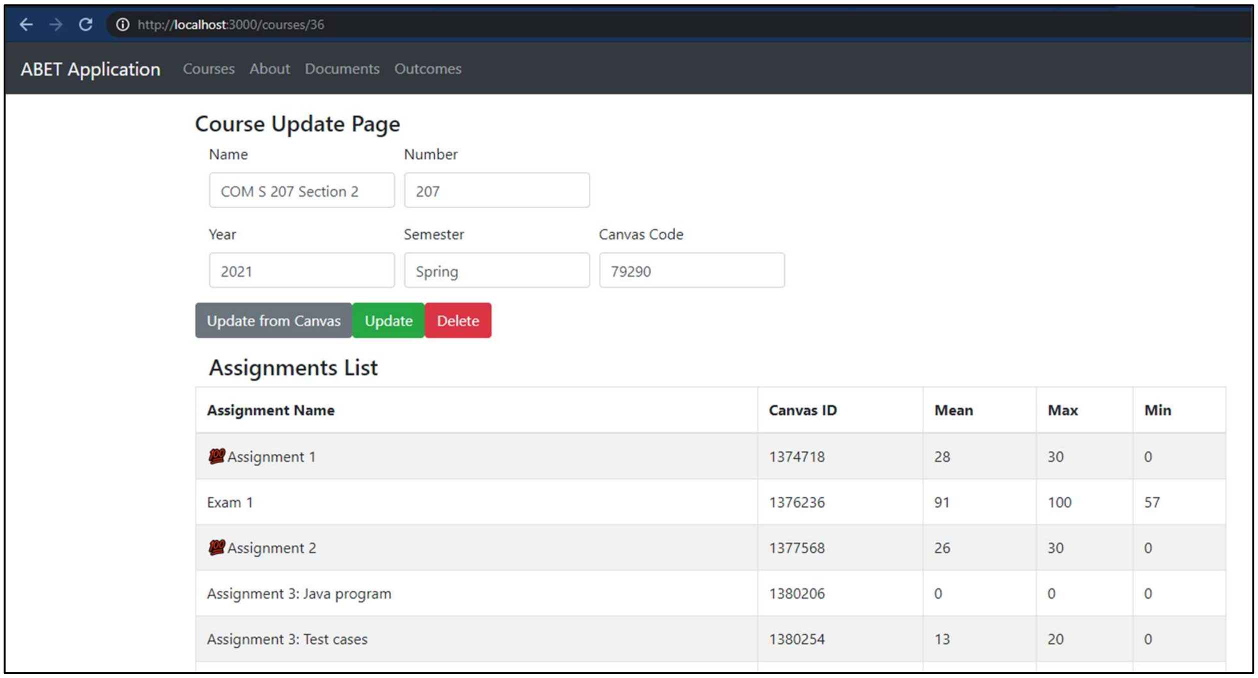
Name: Test Course 1

Number: 12345

Brief details of the selected course

Click on edit button to edit the course

Figure 4.1 Course list page with a search bar



Course Update Page

Name: COM S 207 Section 2

Number: 207

Year: 2021

Semester: Spring

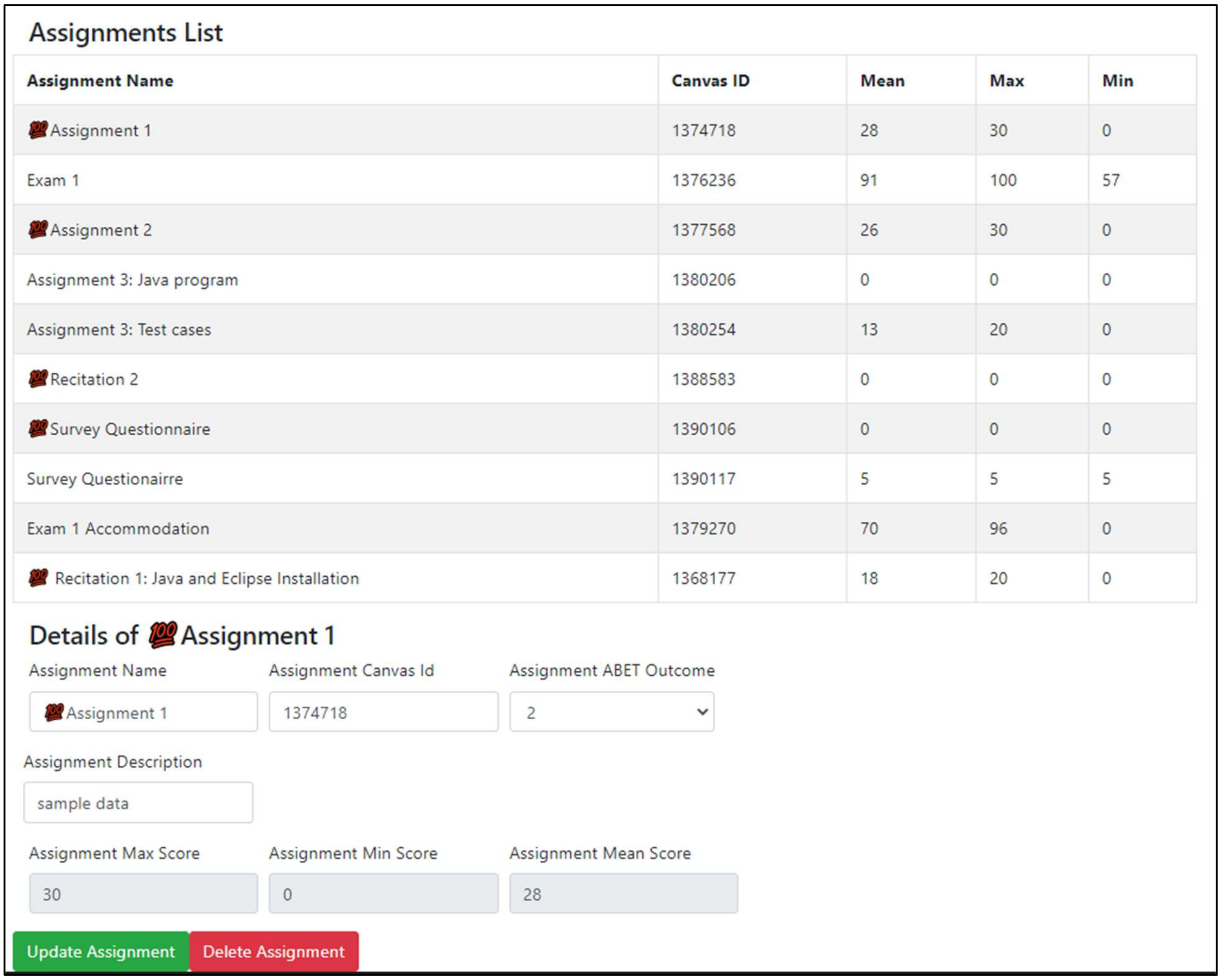
Canvas Code: 79290

Assignments List

Assignment Name	Canvas ID	Mean	Max	Min
Assignment 1	1374718	28	30	0
Exam 1	1376236	91	100	57
Assignment 2	1377568	26	30	0
Assignment 3: Java program	1380206	0	0	0
Assignment 3: Test cases	1380254	13	20	0

Figure 4.2 Course detail page with assignment list

Figure 4.3 shows a complete assignment table with the details of a selected assignment displayed below the table. The course-outcome table, as shown in Figure 4.4, is generated by the application automatically as new outcomes are added to a course. It allows comparison of outcome achievements for various courses across different semesters. It eliminates the manual effort required to create the course-outcome matrix. Figure 4.5 shows the outcomes page, where all the outcomes are listed, and each outcome description can be edited via an editable input field. This is useful when there is a need to customize an outcome description for a specific program.



Assignments List

Assignment Name	Canvas ID	Mean	Max	Min
💯Assignment 1	1374718	28	30	0
Exam 1	1376236	91	100	57
💯Assignment 2	1377568	26	30	0
Assignment 3: Java program	1380206	0	0	0
Assignment 3: Test cases	1380254	13	20	0
💯Recitation 2	1388583	0	0	0
💯Survey Questionnaire	1390106	0	0	0
Survey Questionairre	1390117	5	5	5
Exam 1 Accommodation	1379270	70	96	0
💯 Recitation 1: Java and Eclipse Installation	1368177	18	20	0

Details of 💯Assignment 1

Assignment Name: 💯Assignment 1

Assignment Canvas Id: 1374718

Assignment ABET Outcome: 2

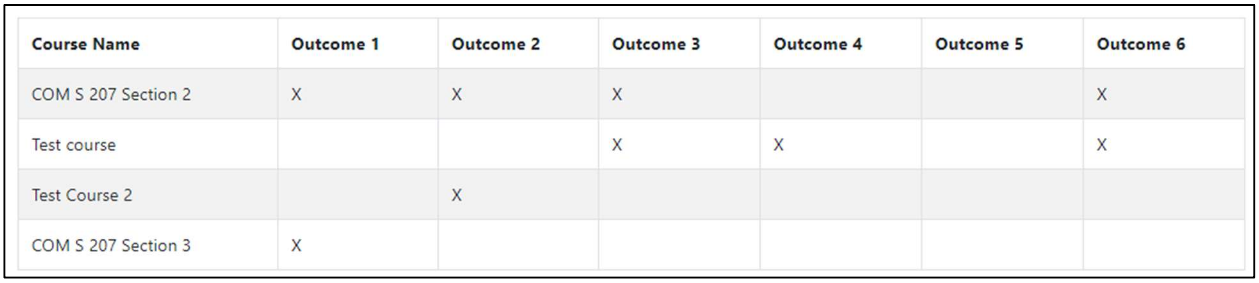
Assignment Description: sample data

Assignment Max Score: 30

Assignment Min Score: 0

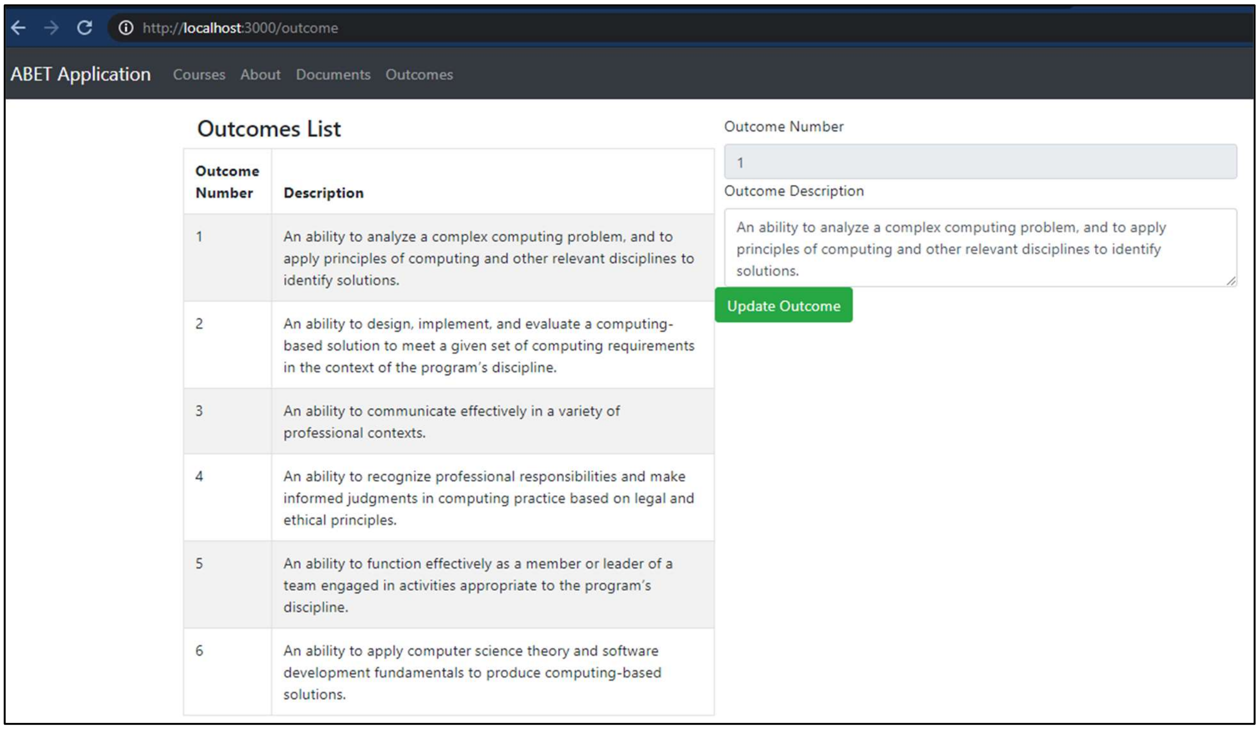
Assignment Mean Score: 28

Figure 4.3 Assignment list



Course Name	Outcome 1	Outcome 2	Outcome 3	Outcome 4	Outcome 5	Outcome 6
COM S 207 Section 2	X	X	X			X
Test course			X	X		X
Test Course 2		X				
COM S 207 Section 3	X					

Figure 4.4 Mapping of course outcomes to ABET program outcomes



Outcomes List

Outcome Number	Description
1	An ability to analyze a complex computing problem, and to apply principles of computing and other relevant disciplines to identify solutions.
2	An ability to design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.
3	An ability to communicate effectively in a variety of professional contexts.
4	An ability to recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles.
5	An ability to function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline.
6	An ability to apply computer science theory and software development fundamentals to produce computing-based solutions.

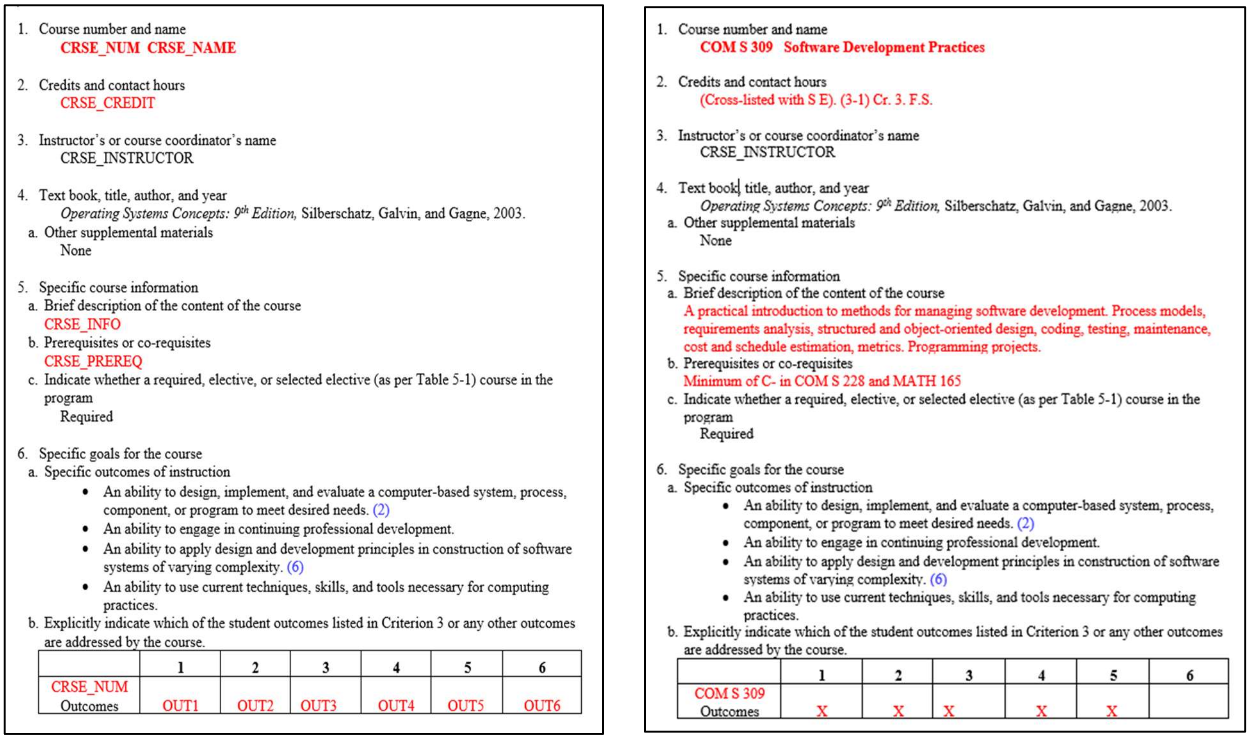
Outcome Number: 1

Outcome Description: An ability to analyze a complex computing problem, and to apply principles of computing and other relevant disciplines to identify solutions.

Figure 4.5 ABET outcomes page

4.3 Report generation

The web-application has the ability to generate hundreds of reports in a few seconds. This automation in report generation eliminates the manual effort of entering data into a template manually, saving the form, and then creating PDFs of the course forms. The user can initiate the generation of the course forms with a single click of a button. The automated report that is generated contains necessary information in the correct fields of the form in the correct format.



1. Course number and name
   CRSE_NUM CRSE_NAME
2. Credits and contact hours
   CRSE_CREDIT
3. Instructor's or course coordinator's name
   CRSE_INSTRUCTOR
4. Text book, title, author, and year
   Operating Systems Concepts: 9th Edition, Silberschatz, Galvin, and Gagne, 2003.
   a. Other supplemental materials
   None
5. Specific course information
   a. Brief description of the content of the course
   CRSE_INFO
   b. Prerequisites or co-requisites
   CRSE_PREREQ
   c. Indicate whether a required, elective, or selected elective (as per Table 5-1) course in the program
   Required
6. Specific goals for the course
   a. Specific outcomes of instruction
   - An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs. (2)
   - An ability to engage in continuing professional development.
   - An ability to apply design and development principles in construction of software systems of varying complexity. (6)
   - An ability to use current techniques, skills, and tools necessary for computing practices.

   b. Explicitly indicate which of the student outcomes listed in Criterion 3 or any other outcomes are addressed by the course.

	1	2	3	4	5	6
CRSE_NUM Outcomes	OUT1	OUT2	OUT3	OUT4	OUT5	OUT6

1. Course number and name
   COM S 309 Software Development Practices
2. Credits and contact hours
   (Cross-listed with S E). (3-1) Cr. 3. F.S.
3. Instructor's or course coordinator's name
   CRSE_INSTRUCTOR
4. Text book title, author, and year
   Operating Systems Concepts: 9th Edition, Silberschatz, Galvin, and Gagne, 2003.
   a. Other supplemental materials
   None
5. Specific course information
   a. Brief description of the content of the course
   A practical introduction to methods for managing software development. Process models, requirements analysis, structured and object-oriented design, coding, testing, maintenance, cost and schedule estimation, metrics. Programming projects.
   b. Prerequisites or co-requisites
   Minimum of C- in COM S 228 and MATH 165
   c. Indicate whether a required, elective, or selected elective (as per Table 5-1) course in the program
   Required
6. Specific goals for the course
   a. Specific outcomes of instruction
   - An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs. (2)
   - An ability to engage in continuing professional development.
   - An ability to apply design and development principles in construction of software systems of varying complexity. (6)
   - An ability to use current techniques, skills, and tools necessary for computing practices.

   b. Explicitly indicate which of the student outcomes listed in Criterion 3 or any other outcomes are addressed by the course.

	1	2	3	4	5	6
COM S 309 Outcomes	X	X	X	X	X	

Figure 4.6 Course report template and completed course report

Figure 4.6 shows the template of the report on the left and the PDF report generated on the right. The fields of the template to be completed are marked in red. The placeholders on the template such as CRSE_NUM, CRSE_NAME is replaced by actual information during the automated report generation process. The completed course report shows course specific information added, in red.

CHAPTER 5. SUMMARY AND FUTURE WORK

The tool shows promise that certain parts of creating an ABET accreditation report can be automated. The automation achieved via web-crawling in keeping course information up to date with the course catalog helps in eliminating the manual activity of comparing text information over and over again. Document parsing and data retrieval from Canvas LMS allows course-outcome matrix and course reports to be generated with as little manual effort as clicking a button. The front-end allows the user of the web-application to view, create, retrieve, and update data into the database effortlessly. The back-end of the application handles HTTP requests and stores data efficiently.

One limitation of this project is that the web crawling code is hard-coded for the specific Iowa state computer science course catalog page. If the HTML structure of the page is changed, then the code needs to be modified accordingly.

Possible future work includes calculating outcome assessments. This requires information of the total number of students taking an exam/assignment and scores obtained by each student. Since this application already has the platform in place to extract such information from Canvas LMS, it would require little work in setting up the formula and calculating the outcome assessments.

Another possible future work is to extended automated report generation to include all reports included in ABET accreditation. Authentication and the ability to add new users with different roles to the application is another important future work in this project.

REFERENCES

1. Essa, E., A. Dittrich, and S. Dascalu. “ACAT: A Web-Based Software Tool to Facilitate Course Assessment for ABET Accreditation.” In 2010 Seventh International Conference on Information Technology: New Generations, 88–93, 2010. https://doi.org/10.1109/ITNG.2010.224.

2. El Rahman, and Sahar S. Shabanah. “Course and Student Management System Based on ABET Computing Criteria.” International Journal of Information Engineering and Electronic Business 8, no. 3 (May 8, 2016): 1–10. https://doi.org/10.5815/ijieeb.2016.03.01.

3. Khwaja, A. A. “A Web-Based Program Outcome Assessment Tool.” In 2018 21st Saudi Computer Society National Computer Conference (NCC), 1–6, 2018. https://doi.org/10.1109/NCG.2018.8593170.

4. Wang, W., and M. Tang. “A Canvas Based Multi-Program Assessment System for ABET CAC and EAC Accreditations.” In 2019 IEEE Frontiers in Education Conference (FIE), 1–6, 2019. https://doi.org/10.1109/FIE43999.2019.9028609.

5. ABET official website at www.abet.org, accessed March 1, 2021

6. J. Enderle, J. Gassert, S. Blanchard, P. King, D. Beasley, P. Hale, & D. Aldridge, The ABCs of preparing for ABET, IEEE Engineering in Medicine and Biology Magazine, 22(4), 2003, 122-132.

7. R. Petrova, A. Tibrewal, T. M. Sobh, "An Electronic Web-based Assessment System," Journal of STEM Education: Innovations & Research, vol. 7, no. 3, 2006.