ACAT

The ABET Course Assessment Tool



CSCI 5330 - Software Engineering

Joel Snider, Jorge Ruiz, Patrick Evans, Wyatt Landers, Stephan Maxi, Langdon Boxxe

Client: Dr. De

Table of Contents

[**Feasibility Report**](#_l29ps8h0lspb) **4**

[Introduction](#_3xfkspvx8b3s) 4

[Requirements](#_huezcqyogf1l) 5

[Similar Systems & Comparisons](#_jr2cb840a8x8) 9

[Risks & Costs Estimates](#_88g124hevyvr) 9

[**Software Requirements Specification Document**](#_vi6bc3otus0h) **10**

[Introduction](#_dpz7uxf9yc7r) 10

[I.I Purpose](#_rfbr6xv4dmv6) 10

[I.II Scope](#_5811wyu3q1vj) 10

[I.III References](#_zf02uzvdrost) 11

[I.IV Overview](#_lw6prs2uhu22) 11

[Overall Description](#_rqmvz5vyyjrt) 11

[II.I Product Perspective](#_kwczjwbo37r3) 11

[II.II Product Functions](#_rh1m056a6065) 12

[Specific Requirements](#_ozuanic38frb) 12

[III.I General Requirements](#_d6hbd9dieznc) 12

[III.II Assessment Report Requirements](#_hmps3udzf7oo) 12

[III.III Course Outcome Requirements](#_gad1gaqxv51j) 13

[III.IV Direct Assessment Requirements](#_dvxauxc1jfo0) 13

[III.V Report Functional Requirements](#_47pc2hlej3hk) 14

[III.VI Non-Functional Requirements](#_owtwzhctnv7p) 14

[Specific Requirements](#_shdhnnirzex7) 14

[**Software Design Document**](#_utxubrdt1wgf) **16**

[Abstract](#_ji9pfad03i6c) 16

[Introduction](#_u5jfid5et8rf) 17

[II.I Purpose of System](#_bfxc5d72bzs7) 17

[II.II Design Methodology](#_hphy9l4kwtvi) 18

[II.III Definitions, Acronyms, and Abbreviations](#_afp0hbcglp1) 18

[II.IV Overview of Document](#_uin7sat42bph) 18

[Proposed Software Architecture](#_pj2cdqpodzco) 19

[III.I Overview](#_3o2so7omfhnp) 19

[III.II Generative Architecture](#_6n5k8l5mhcy2) 20

[Object Design](#_c40pkmmw69qk) 20

[IV.I Overview](#_vlocxrgbx9dc) 20

[**Testing & Quality Assurance Document**](#_2o5ofezejbm) **22**

[Introduction](#_nvw5i5by6lsm) 22

[Testing Checklist](#_bvxxffl8r1bq) 22

# Feasibility Report

## **Introduction**

Purpose: The purpose of this feasibility report is to outline the function, purpose, requirements, method of creation, and risks and costs of the ABET Course Assessment Tool (ACAT). This document will also attempt to defend the idea that the project described here is both viable and acceptable according to the requirements set forth by the client.

Project Team: Our team is supervised by Dr. Andrew Allen and our client is Dr. Prapita De. The below table lists the members of the project team along with each member’s university email.

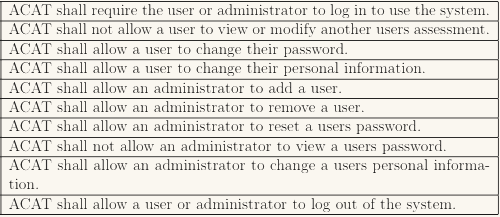
|  |  |
| --- | --- |
| Member | Email |
| Joel Snider | js13623@georigasouthern.edu |
| Stephan Maxi | sm08333@georigasouthern.edu |
| Wyatt Landers | wl00951@georgiasouthern.edu |
| Patrick Evans | pe00461@georgiasouthern.edu |
| Jorge Ruiz | jr07471@georgiasouthern.edu |
| Langdon Boxxe | db04511@georgiasouthern.edu |

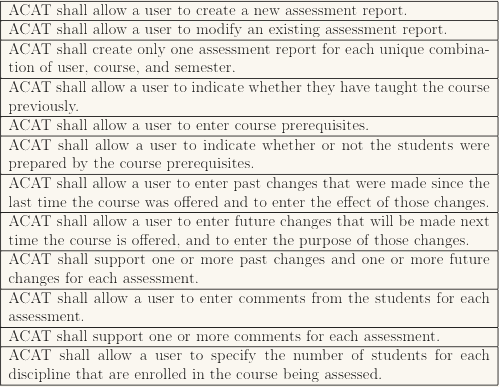
Project Description: The project itself is based heavily on the thesis published from the University of Reno Nevada titled “ACAT: ABET Course Assessment Tool” by Eugene O. Essa. This thesis outlines a web-based application that allows the user to enter a variety of data and information about a certain course, including assignment information, student information, and individual grades. The purpose of the tool is to simplify the process of creating assessment reports in order to maintain ABET accreditation.

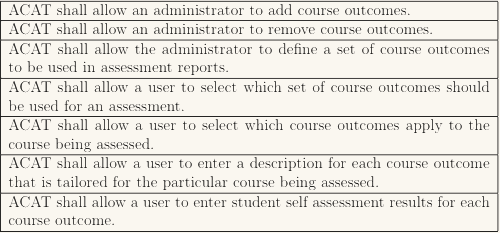
Functionality: This tool will be designed with simple usability in mind. We plan to allow the user a variety of options on how to input course data, whether that be manually, or via spreadsheet style files, such as .xlsx or .csv, or via .pdf files. Our user interface will be intuitive and simple to navigate in order to allow usability from anyone.

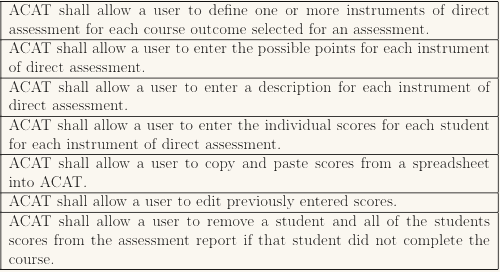
## **Requirements**

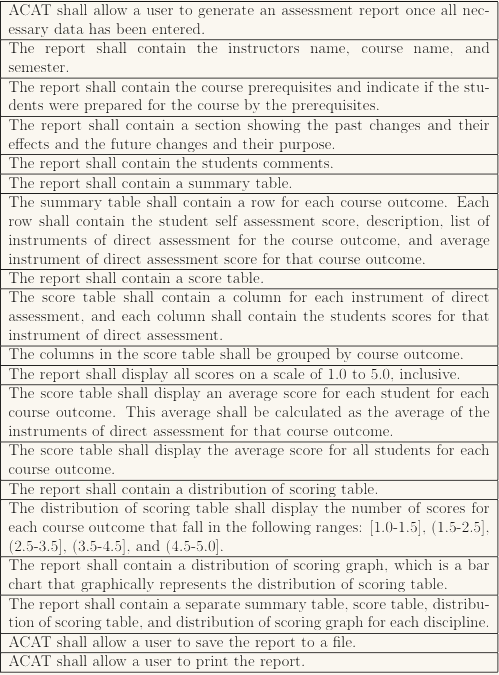
Technical: The previously mentioned thesis outlines in Chapter 4 a list of 63 functional requirements the ACAT system must have. As we aim to use the outline given in this thesis as the basis for our project, we also have made it our goal to satisfy these requirements. The tables containing these 63 requirements are given below.

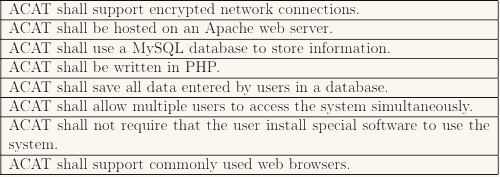












## **Similar Systems & Comparisons**

The University of Reno, Nevada uses ACAT software most similar to the one we are attempting to create. The primary differences between our proposed system and theirs is that our will allow for a variety of input and allow for individual weights of each assignment in the creation of course reports for accreditation purposes. Our system also allows the user to pull random samples from the lowest, highest, and middle thirds of grade ranges for each course or each assignment.

## **Risks & Costs Estimates**

Risks: With our system being web-based and making use of a database containing personal information of potentially thousands of student and university faculty members, data security is the greatest risk to our project. We aim to ensure that all access to contained data is encrypted using RSA encryption and to make sure that our website is secure from query injection methods in any of the input fields that the user has access to.

Costs: Since this project is being conducted fully by senior-level college students who work for free in order to receive a passing grade, our total cost estimate comes out to $0.00.

# Software Requirements Specification Document

## **Introduction**

### I.I Purpose

The purpose of this document is to present a detailed description of the ABET Course Assessment Tool (ACAT). It will outline the function and purpose of the tool and describe how the tool is intended to be used. This document is to be used by both users of the tool and those interested in its development.

### I.II Scope

ACAT is a tool intended for use by professors of universities approved by ABET accreditation. It is designed to assist faculty in the creation of course assessment reports for ABET accreditation. The tool allows the user to enter course learning outcomes, mission objectives, evaluation instruments and assignments, and grades in order to generate a report.

It will allows individual users to log in to the system and create a portal for any number of courses they need to create a report for. All files and information would be saved on a local database that can be backed up into a cloud based server.

### I.III References

*IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specification. IEEE Computer Science, 1998.*

### I.IV Overview

The following chapter provides a more detailed description of the ACAT Tool. Chapter 3 lists the specific requirements for our software. The final chapter will detail the process of how we created the software and how our team divided up the work.

## **Overall Description**

### II.I Product Perspective

ACAT is a self-contained software application designed for a user to load on their local machine. Upon start up, the user will enter their login credentials and will be given access to their personal portal. This allows the user to view previously made courses or update any necessary information. There is also a button which allows the user to create a new course from scratch. When a new course is created, the tool will ask the user for a variety of information. This includes the course name, semester, number of students, number and names of all the assignments for the course. Any of this information can be edited later if need be.

### II.II Product Functions

The tool allows the user to input any information and parameters required to create an ABET appropriate report and store any class information needed.

## **Specific Requirements**

### III.I General Requirements

The general functional requirements for ACAT are as follows:

1. ACAT shall require the user or administrator to log in to use the system.
2. ACAT shall not allow a user to view or modify another user’s assessments.
3. ACAT shall allow a user to change their password or personal information.
4. ACAT shall allow an administrator to add and delete users, and can reset passwords.
5. ACAT shall not allow an administrator to view a user’s password.
6. ACAT shall allow a user or administrator to log out of the system.

### III.II Assessment Report Requirements

The requirements for the assessment reports created by ACAT are as follows:

1. ACAT shall allow a user to create a new assessment report.
2. ACAT shall allow a user to modify an existing assessment report.
3. ACAT shall create only one assessment report for each unique combination of user, course, and semester.
4. ACAT shall allow a user to enter past changes that were made since the report information was last accessed.
5. ACAT shall support one or more changes to an assessment report at a time.
6. ACAT shall support one or more comments for each assessment, viewable only to the user.

### III.III Course Outcome Requirements

The requirements for the course outcomes are as follows:

1. ACAT shall allow a user to input desired course outcomes and mission objectives.
2. ACAT shall allow a user to edit course outcomes.
3. ACAT shall allow a user to enter comments for each course outcome.

### III.IV Direct Assessment Requirements

The direct assessment requirements for ACAT are as follows:

1. ACAT shall allow the user to define one or more instruments of assessment for each course outcome.
2. ACAT shall allow a user to enter a score associated with each assessment for each student in the course.
3. ACAT shall allow a user to input comments for each assessment.
4. ACAT shall allow a user to edit entered scores.
5. ACAT shall allow a user to remove any or all of the students’ scores from an assessment report.

### III.V Report Functional Requirements

The functional requirements for reports created using ACAT are as follows:

1. ACAT shall allow a user to generate an assessment report once all necessary has been entered.
2. ACAT reports shall contain instructor’s name, course name, semester, and CRN.
3. ACAT reports shall contain a summary table.
4. ACAT reports shall contain a score table.
5. ACAT reports will show all course outcomes, mission objectives, ABET requirements, assessment average scores, assessment standard deviations, and score averages per learning outcomes.
6. ACAT shall allow a user to save the report to a file.
7. ACAT shall a user to edit reports as necessary.

### III.VI Non-Functional Requirements

All other non-functional requirements for ACAT are as follows:

1. ACAT shall support local database integration using SQLite.
2. ACAT shall be a locally run and hosted computer application.
3. ACAT shall allow multiple users to have accounts in the system.

## **Specific Requirements**

Our team has chosen to use Trello as a team labor management tool. Joel Snider is the team lead and handles relating with the client, Dr. De. Wyatt Landers and Jorge Ruiz are handled the application interface and ensured that all data is processed correctly. Patrick Evans and Stephan Maxi were responsible for creating and managing the database that stores all necessary information for report creation.

# Software Design Document

## **Abstract**

The ABET Course Assessment Tool (ACAT) is a software designed with the intention to assist engineering professors of ABET accredited universities in the creation of mandatory ABET reports. It allows the user to input parameters for each course and assessment tool used throughout a semester, save this data, and generate an assessment report with relative ease. Within the tool, whenever a new course is created, the user is asked to enter the course’s learning outcomes, mission objectives, ABET learning objectives, number of evaluation instruments, the average score and standard deviation for each evaluation instrument, and program outcomes for the course. Once the user has input all data necessary to generate a report, they can do so at the click of a button.

The ACAT application itself will be locally hosted on the user’s device. It will run on Java so that it is easy to access on most devices. When the user first opens the program, it will establish a local database to hold all the data entered. It was also include an external database that will allow the user to access their course information from different devices. Each user will have a group of courses associated with their login credentials, which will be created by an admin account.

## **Introduction**

Many universities across the United States seek the approval of the Accreditation Board for Engineering and Technology, or the ABET, for their engineering related courses. In order to do this, professors of these engineering courses must create a report for each of the courses they teach that detail the learning outcomes, mission objectives, and assessment grades to be submitted each semester. This process can be long and tedious, and takes up much of the time that a professor could use to focus instead on instructing students. The ABET Course Assessment Tool, or ACAT, is created to help make this process more simple. This chapter provides an overview of what ACAT is and how it will work.

### II.I Purpose of System

The reason for the create of ACAT is to simplify the process of ABET accreditation reports for engineering courses. ACAT will allow a user to create a module for any number of different courses being taught in a certain semester and enter different parameters for each course that are required for create an ABET report. These include learning outcomes, mission objectives, assessments with grade samples ABET requirements, and averages for program outcomes. The user can edit of this information at any times as needed. Once all necessary information for a certain course has been entered, ACAT will generate an accreditation report at the click of a button.

### II.II Design Methodology

ACAT was designed with usability and security as two primary factors. We understand that accreditation report creation is tedious, but not difficult. As such, we understood that our software for generating these reports should also be as straightforward as possible. Also, since student records and grade are crucial to the use of our software, we understood that the security measures related to our content and ensuring proper encryption methods was extremely important.

### II.III Definitions, Acronyms, and Abbreviations

* ABET - Accreditation Board for Engineering and Technology, Inc.
* ACAT - ABET Course Assessment Tool
* Alphanumeric - A string consisting of letters and numbers
* API - Application Programming Interface
* Functional requirement - All absolutely necessary requirements our system must satisfy
* GUI/UX - Graphical User Interface
* Nonfunctional requirements - all constraints on the system that are not absolutely necessary
* System - The entirety of the ACAT software and all included code

### II.IV Overview of Document

The following chapters go into further detail over the specifications and design of our ACAT system. Section III details the overall architecture of ACAT and section IV discusses its object design in detail.

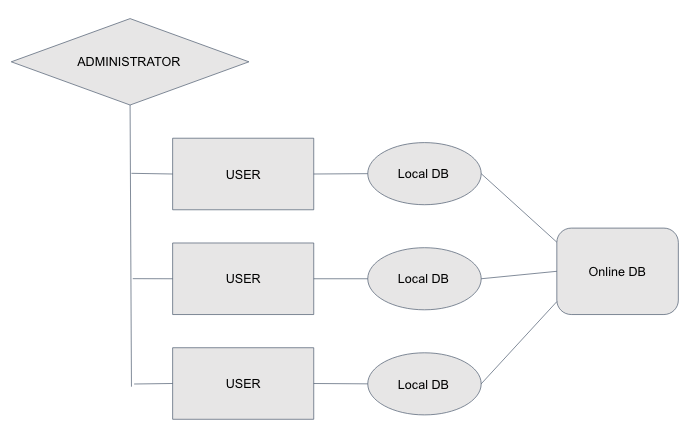
## **Proposed Software Architecture**

Software architecture describes the overall design of a piece of software. Architecture is by far the most important information when it comes to the creation of any software. The following sections detail the chosen architecture details for the ACAT system.

### III.I Overview

ACAT is designed using a simple three-tier architecture schema. The system is primarily a stand-alone application that allows users to generate reports, but also encompasses two databases to enhance the system design. The first tier is that application that allows the entering of information and generation of reports. The second is a local database to be loaded onto a user’s machine that allows any and all information necessary in the stand-alone application to be saved so that it can be used or modified at a later time. The third tier is a cloud-hosted database that the user can synchronize their local system to with relative ease. The cloud database shares the same schema as the local one, and the application allows for the two databases to stay in sync with one another whenever the user decides to back one up with the other.

### III.II Generative Architecture



This diagram breaks down the overall structure of the ACAT system. Administrator profiles oversee and create user profiles that each link to a database on their own machine. Each of these databases synchronizes to a cloud hosted database.

## **Object Design**

### IV.I Overview

Our system is an event driven program so each of our objects within the system is designed to fit this structure. The entry of each learning outcome, mission objective, ABET outcome, assessment with grade samples, and report is its own object within our system. Each of these are generated using a factory design pattern in C#. We manipulate the data within these objects to generate the necessary information to create an ABET report. For example, since each line in a report links back to a single learning outcome, we have to take the data from different combinations of mission objectives and assessments to generate an average score for each ABET outcome. We have to do the same with the standard deviation for each. As these objects are created they are stored in our local database. Upon report creation the user input the connections between these objects and this data is stored in a temporary dictionary data structure. This allows for easy access and partitioning of the data for each manipulation for report creation.

# Testing & Quality Assurance Document

## **Introduction**

This document outlines the methods and degrees to which the ACAT software system designed by our team was tested. This document was also used as a tool by our development team to ensure that all requirements for our system were met.

## **Testing Checklist**

The following checklist details the areas of our system that were tested with a description detailed where applicable.

SYSTEM LEVEL REQUIREMENTS VALIDATION

|  |  |  |
| --- | --- | --- |
| **Item** | **Response** | **Description** |
| Have the business context and justification for the system been properly developed? | Yes |  |
| Has overall feasibility been addressed? | Yes |  |
| Have all stakeholders been identified and polled for agreement? | N/A |  |
| Have the overall function and behavior of the system been defined? | Yes |  |
| Based on existing documentation/information, do you understand the system in the context of each of the views in the system engineering hierarchy? | Yes |  |
| Have system processes been adequately (unambiguously) and consistently defined? | Yes |  |
| Is system output and input adequately defined? | Yes |  |
| Have system-level assumptions, simplifications, limitations, constraints and preferences been explicitly and unambiguously stated? | Yes |  |
| Has simulation been done to demonstrate technological feasibility? | No | We created a feasibility document, but ran no real simulation. |
| Has a data architecture been identified? | Yes |  |
| Has an application (functional) architecture been defined? | Yes | Three-tier architecture |
| Has the required technology infrastructure for the system been adequately defined? | Yes |  |
| For business applications: Have ISP and BAA (SEPA, 5/e, p. 253) been performed? | N/A |  |
| Has requirements elicitation been performed at the system level? | No |  |
| Has the scope of the system been bounded? | Yes |  |
| Has business and technical feasibility been assessed? | Yes |  |
| Have usage scenarios been created at the system level? | Yes | Use cases are outlined in our design document. |
| Has a requirements management process been established for the system? | Yes |  |
| Is the allocation for software reasonable and well-defined?` | Yes |  |

SOFTWARE REQUIREMENTS SPECIFICATION

|  |  |  |
| --- | --- | --- |
| **Item** | **Response** | **Description** |
| Do stated goals and objectives for software remain consistent with system goals and objectives? | Yes |  |
| Have important interfaces to all system elements been described? | Yes |  |
| Have all data objects been described? Have all attributes been identified? | No |  |
| Do major functions remain within scope and has each been adequately described? | Yes |  |
| Have functions been refined (elaborated) to an appropriate level of detail? | Yes |  |
| Is information flow adequately defined for the problem domain? | No |  |
| Are diagrams clear; can each stand alone without supplementary text? | No | Our diagrams require captions. |
| Is the behavior of the software consistent with the information it must process and the functions it must perform? | Yes |  |
| Have events and states been identified? | No | I don’t know what that means. |
| Are design constraints realistic? | Yes |  |
| Have technological risks been fully defined? | Yes |  |
| Have alternative software requirements been considered? | N/A |  |
| Have validation criteria been stated in detail; are they adequate to describe a successful system? | Yes |  |
| Have inconsistencies, omissions or redundancy been identified and corrected? | Yes | Our software system has many omissions. |
| Is the customer contact complete? | N/A |  |
| Has the user reviewed the Preliminary User's Manual or prototype? | No |  |
| How are the Software Project Plan estimates affected? | N/A |  |

SOFTWARE DESIGN CHECKLIST

|  |  |  |
| --- | --- | --- |
| **Item** | **Response** | **Description** |
| Does the overall design implement all explicit requirements? Has a traceability table been developed? | Yes |  |
| Does the overall design achieve all implicit requirements? | Yes |  |
| Is design notation standardized? Consistent? | Yes |  |
| Is the design represented in a form that is easily understood by outsiders? | No | Nothing about this stuff would be easily understood by outsiders. |
| Does the overall design provide sufficient information for test case design? | Yes |  |
| Is the design created using recognizable architectural and procedural patterns? | Yes | Three-tier |
| Does the design strive to incorporate reusable components? | No |  |
| Is the design modular? | Maybe | Not sure what that means |
| Has the design defined both procedural and data abstractions that can be reused? | Yes |  |
| Has the design been defined and represented in a stepwise fashion? | no |  |
| Has the resultant software architecture been partitioned for ease of implementation? Maintenance? | Yes |  |
| Have the concepts of information hiding and functional independence been followed throughout the design? | Yes |  |
| Has a Design Specification been developed for the software? | No |  |