

Running Head: Documentation

Training and Testing Program Documentation

Kat Reynolds

Nathan Miller

Simon Choi

Geoff Dobson

Angel Lopez

Hiruy Sinote

Central Washington University

2/24/2024

Table of Contents

Project Overview and Requirements.....	3
Architecture Requirements.....	3
Software Architecture and Installation.....	4
Flask Python Application.....	4
Development Technologies.....	5
How to Set Up Email Configuration.....	6
Database Schema and Data Design.....	7
Data Hierarchy.....	8
Test Tables.....	10
Other Tables.....	11
User Manual.....	12
Login / Registration.....	12
Training Data Hierarchy Editor.....	14
Navigation.....	14
Modifying Data.....	15
Add Button.....	17
Edit Button.....	18
Topic Hierarchy.....	19
Learning Objective Hierarchy.....	20
Questions Hierarchy.....	21
Test Creation.....	23
Navigation.....	23
How the Filters Work.....	23
Random Test Creation.....	24
Manual Test Creation.....	27
Error Handling.....	31
Test and Answer Key Export.....	32
Test modification.....	33
Test Delete.....	34
Test Score Entry and Management.....	35
Navigation.....	35
How to manage tester's score.....	36
Test Score Metric Export.....	38
Development Lifecycle.....	41
Development Methodology.....	41
Initial Testing Feedback and Software Improvements.....	41
Hardships.....	42

Training and Testing Program Documentation

Project Overview and Requirements

This project focused on developing an application that could create and manage training and testing information, generate question lists for tests, and store historical and statistical information for test results. The hierarchical structure also should enable users to create and track training data. The requirements outlined for this can be split into two sections, architecture requirements and functional requirements as shown below:

Architecture Requirements

- The software shall be developed as a portable desktop application compatible with Windows
- The software shall implement a secure user authentication mechanism, allowing both general users and administrators to log in using unique credentials.
- The system shall have the ability to securely store data locally, ensuring the confidentiality and integrity of user information. This includes user authentication data, test scores, and other sensitive information.
- The system shall have the ability to securely connect with a remote database for synchronizing data and sharing between client computers.
- The system shall have the ability to import and export encrypted data for sharing between client computers without connecting to a database

Functional Requirements

Data Hierarchy

- Subjects: Store a hierarchical list of training subjects.
 - name
 - description
 - parent subject
 - subject reference list
 - is subject or is folder to differentiate between the organizational structure and the subjects within it. Maybe a visual cue to what is what?
- Topics: Store a list of training topics for each subject.
 - name
 - description
 - facility/location applicability (multi select) (Configurable)
 - topic references (select from or add to reference list)
- Learning Objectives: Store learning objectives for each topic.
 - description
 - methods of demonstration (bloom's taxonomy) (configurable)
 - training level applicability (configurable)(multi select) (applicant, apprentice, journeyman, senior, chief, coordinator)
 - tags/metadata - (configurable) emergency response, safety, compliance, regulatory, nerc, ferc, etc
 - objective references (select from or add to topic references)

- page/section in reference (free text))
- Questions: Store test questions for each learning objective
 - question (rich text, allowing images)
 - answer (rich text, allowing images)
 - applicable learning objective
 - max point value
 - history of scores

Manual Test Creation

- The system shall enable users to create custom tests by selecting questions from a list filtered by subject, topic, training objective, and/or qualification level.
- The system shall enable users to create randomized tests based on a selected subject, topic, training objective, and/or qualification level.

Automatic Test Generation

- For a given set of test questions, the system shall display the expected average score based on historical question scores.
- The software shall be able to adjust question inclusion in randomized tests based on historical average scores to target the test for a specific average score.

Score Tracking

- The system shall allow users to enter scores for test questions and implement tracking mechanisms to trend and average scores by question, category, by testee.

Reporting Functionality

- The system shall have the ability to generate, filter, and sort printable reports of metrics such as:
 - Average scores by test or tester
 - Average scores by subject, topic, training objective, or question.
 - Average scores by tag
- The system shall allow selection of a time range filter for reports.
- The system shall permit exporting reports to common file formats such as PDF and CSV.

Along with these requirements, the project should be delivered after undergoing comprehensive testing and quality assurance to identify bugs and ensure functional reliability. The user interface should be intuitive and easy to understand for all users. This application should also accommodate scalability for the client to implement future expansions while maintaining the project's performance. Lastly, this documentation with user manuals and system information was a deliverable.

Software Architecture and Installation

Flask Python Application

The application runs on Python version 3.10.12 and requires the following Python package dependencies:

Flask==2.0.3

Flask_Bcrypt==1.0.1

```
Flask_Mail==0.9.1
Flask_SQLAlchemy==2.5.1
itsdangerous==2.1.2
matplotlib==3.3.4
numpy==1.26.4
Pillow==9.1.0
python-dotenv==1.0.1
reportlab==3.6.8
SQLAlchemy==1.4.51
```

The software was built using Flask, a micro-web framework for Python. Our application was built similarly to a Model-View-Controller format. Primarily, the controller and main application driver is `wildcats.py`, which defines different url routes for page display and post requests. Database model operations are primarily handled in `data_retriever.py`, though some database operations occur directly in `wildcats.py`. View template files are stored in `/templates`, consisting of static and dynamic pages that are rendered by the controller based on database state and user actions. Static files such as JavaScript, CSS, and images not in the database are stored in `static`. Some javascript is written directly into HTML files.

The application has been validated to run correctly on Firefox, Chrome, and Microsoft Edge.

The application also requires a 50 char length `SECRET_KEY` and `SALT` (length of your choosing, used for new account verification links) defined in a `.env` file not present in the repository.

The Database schema must also be loaded onto a MySQL service. Upon deployment, an admin account manually inserted into the data as accounts cannot be registered without a logged in admin account. The schema creation script has been validated to work with MySQL Workbench's Server > import data feature.

Development Technologies

This software was primarily developed using the JetBrains Pycharm IDE. This included the main flask framework files including Python, HTML, CSS, and JavaScript. The database model and schema creation script were designed and generated in MySQL Workbench. Other supporting tools included PuTTY for production server CLI access, and FileZilla for server file updates. Git and GitHub were used for version control.

Production Server

- Single core of AMD Epyc 2.0 Ghz CPU
- 2GB RAM
- 50GB SSD Disk Space
- Ubuntu Linux 22.04
- Nginx Reverse Proxy

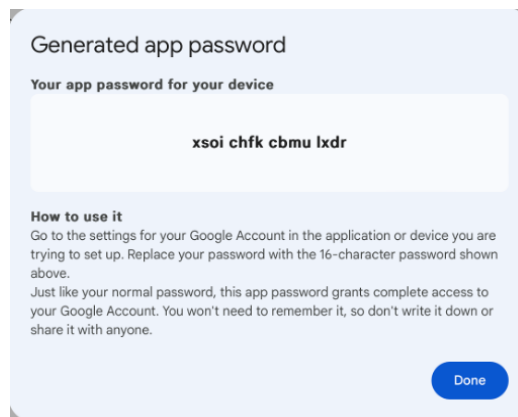
- Unicorn Web Server Gateway Interface
- Flask Web Server

We use a virtual private server that has the specifications above. It is connected to a 100 gigabit data connection in a data center in San Francisco, USA. It has local and network firewalls that allow connections over ports 22 (SSH), 80 (HTTP), and 443(HTTPS). Nginx is set up as a reverse proxy server that routes traffic over an HTTPS connection to the Unicorn Web Server Gateway that communicates with our instance of the Flask microframework.

How to Set Up Email Configuration

This email configuration is used for sending verification codes to users for two-factor authentication.

1. Access Google Account Settings
 - a. Sign in to your Google account and navigate to your Google Account settings.
2. Navigate to Security Settings
 - a. In the Google Account dashboard, click on the “Security” tab located in the left-hand side menu.
3. Enable Two-Step Verification (if not already enabled)
 - a. If you have not already set up two-step verification for your Google account, you need to enable it to generate an App Password
4. Generate an App Password
 - a. Click on the “2-Step Verification” option and scroll down to find the “App Passwords” section, or type ‘App Passwords’ in the search bar to navigate to the section.
 - b. Enter the App name and click the ‘Create’ button. This will generate and display a 16-character app password.
 - c. Copy the app password to paste it into the ‘config.py’ file



5. Input email account and app password in the ‘config.py’ file
 - a. Open the ‘config.py’ file and input your Google account and generated app password.

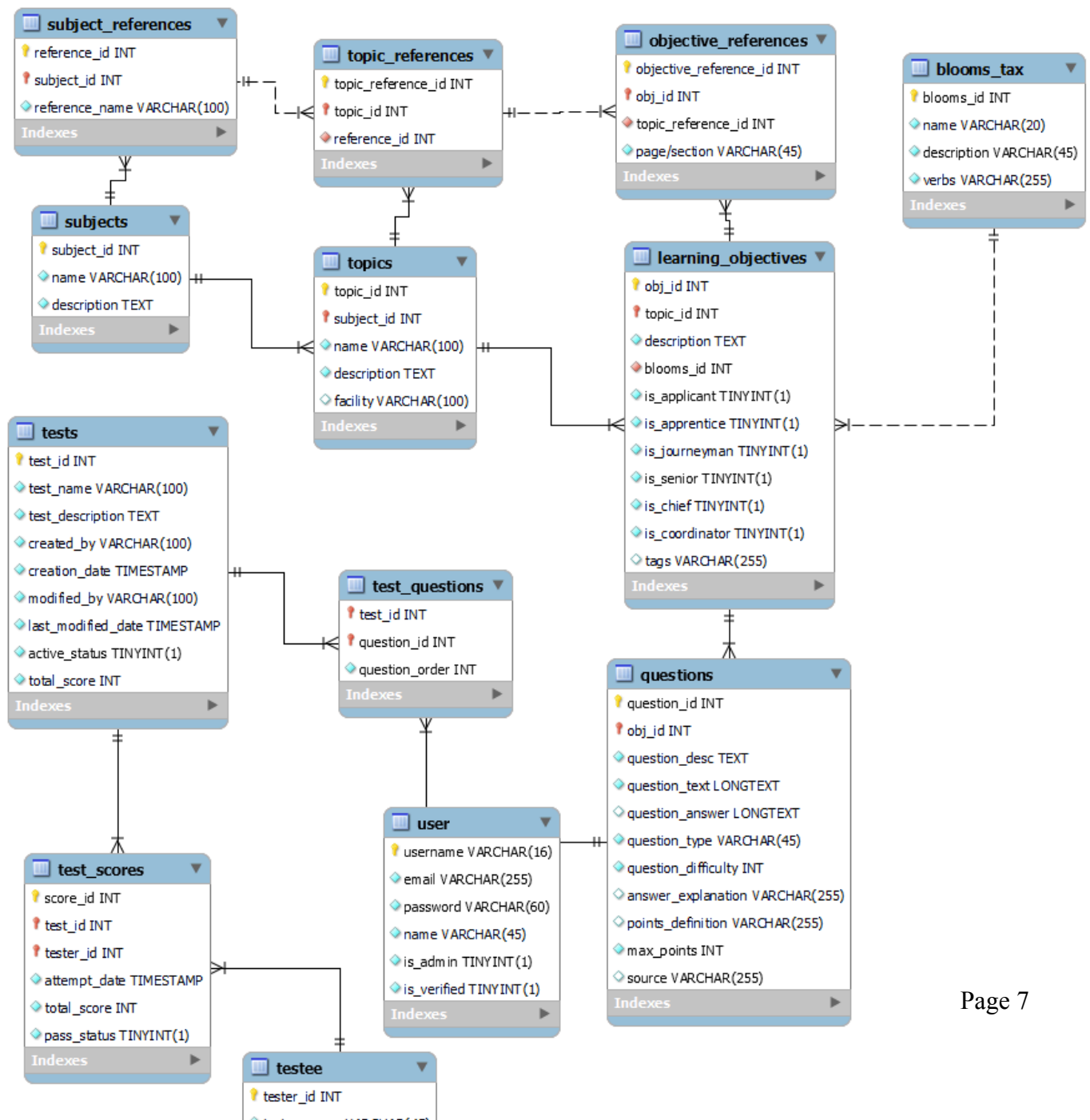
```

config.py — Edited
#config.py
#stores admin user's email information to send verification email

class MailConfig:
    MAIL_SERVER = 'smtp.gmail.com'
    MAIL_PORT = 465
    MAIL_USERNAME = 'TrainingTestingProgram@gmail.com'
    MAIL_PASSWORD = 'xs0i chfk cbmu lxdr'
    MAIL_USE_TLS = False
    MAIL_USE_SSL = True

```

Database Schema and Data Design



test_train_db Schema Entity Relationship Diagram

Data Hierarchy






The main training data hierarchy is made up of the subjects, topics, learning_objectives, questions, and blooms_tax tables.

The subjects table is the top level of the data hierarchy, and is referenced by its child topics and subject_references tables. Its identifying key is subject_id, a unique integer value associated with every subject. It also contains descriptive attributes including the name and description for the subject.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
 subject_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
 name	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 description	TEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>












subjects table data definitions

The topics table is the next level of the data hierarchy, containing topic information for every subject, and is referenced by its child learning_objectives and topic_references tables. Its identifying key is made up of the topic_id, a unique integer value associated with every topic, and its parent subject_id value. Descriptive attributes include the name, description, and applicable facility (optional) for the topic.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 topic_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 subject_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 name	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 description	TEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 facility	VARCHAR(100)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL





topics table data definitions

The learning_objectives table is the next level of the data hierarchy, containing learning_objective information for every topic, and is referenced by its child questions and objective_references tables. Its identifying key is made up of the obj_id, a unique integer value associated with every learning objective, and its parent subject_id value. The table references the blooms_tax table which stores a learning objective's associated bloom's taxonomy verb. Other descriptive attributes include a description, boolean flags for each skill level, and an optional attribute tag containing keywords to filter learning objectives by.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 obj_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 topic_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 description	TEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 blooms_id	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 is_applicant	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 is_apprentice	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 is_journeyman	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 is_senior	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 is_chief	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 is_coordinator	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 tags	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

learning_objectives table data definitions

The blooms_tax table is a table containing static bloom's taxonomy verb words and their descriptions, and is referenced by an associated learning objective. Each verb is identified by blooms_id, a unique integer, and contains static descriptive data.












Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
 blooms_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 name	VARCHAR(20)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 description	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 verbs	VARCHAR(255)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

blooms_tax table data definitions

blooms_id	name	description	verbs
1	Remember	Recall facts and basic concepts	Define, duplicate, list, memorize, repeat, state.
2	Understand	Explain ideas or concepts	Classify, describe, discuss, explain, identify, locate, recognize, report, select, translate.
3	Apply	Use information in new situations	Execute, implement, solve, use, demonstrate, interpret, operate, schedule, sketch.
4	Analyze	Draw connections among ideas	Differentiate, organize, relate, compare, contrast, distinguish, examine, experiment, question, test.
5	Evaluate	Justify a stance or decision	Appraise, argue, defend, judge, select, support, value, critique, weigh.
6	Create	Produce new or original work	Design, assemble, construct, conjecture, develop, formulate, author, investigate.

blooms_tax static table data

The questions table is the lowest level of the data hierarchy, which contains the questions for every learning objective. It is referenced by the test_questions table. Its identifier is made up of a question_id, a unique integer for every question, and its parent obj_id value. This table contains attributes used across many different features of the application. The question_desc attribute contains a plain text description of the question, used for display in the data hierarchy and test creation features. The question_text and (optional) question_answer attributes contain html encoded text for formatted display of question text and images in both test and answer key export. A question also has the attribute question_type, which is multiple choice, true / false, short answer, and any other question types manually defined by the user. Other descriptive attributes include the question_difficulty (1 - 5) and max_points integer attributes, and the optional answer_explanation, points_definition (rubric), and source text attributes.










Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 question_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 obj_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 question_desc	TEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 question_text	LONGTEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 question_answer	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 question_type	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 question_difficulty	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 answer_explanation	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 points_definition	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
 max_points	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 source	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

questions table data definitions

The aforementioned reference tables in the data hierarchy tables went unused and are not currently implemented in the application. Source references for individual questions can still be added and are displayed on answer keys.



Test Tables

The test table contains descriptive attributes for tests. Its primary identifier is the test_id attribute, a unique integer value assigned to every test. Other descriptive attributes include the test name, description, active status and total score. It also contains metadata for the original user who created the test, the last user to update it, and the original creation and last modified time and date.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
 test_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
 test_name	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 test_description	TEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 created_by	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 creation_date	TIMESTAMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 modified_by	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 last_modified_date	TIMESTAMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 active_status	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 total_score	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

test table data definitions

The test_questions table contains references to questions that make up tests. Its primary identifier is made up of the test_id of the test that the question is a part of, and the question_id of the actual question from the data hierarchy. It also contains an integer denoting which question number on the test it is.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
 test_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 question_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 question_order	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>







test_questions table data definitions

The testee table contains testees (people who take the tests). It includes a unique primary integer identifier and a testee name.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
 tester_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
 testee_name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

testee table data definitions







The test_score table contains test_scores for testees for referenced tests. Record primary identifiers are made up of score_id a unique integer for each score, the referenced tester_id, and the referenced test_id. Other descriptive attributes include the total score earned on a test, a binary flag indicating pass status, and the attempt date and time.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
 score_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
 test_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 tester_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 attempt_date	TIMESTAMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 total_score	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 pass_status	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

test_scores table data definitions

Other Tables

The user table contains identifying and authentication attributes for users that log in to the system. The table is referenced by test_scores and test_questions. Its primary identifier is the username attribute, which must be unique. It also contains the email of the user, a hashed 60-char length password, real name, and boolean flags for admin accounts and if the account has completed initial verification.

 username	VARCHAR(16)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 email	VARCHAR(255)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 password	VARCHAR(60)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 is_admin	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 is_verified	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

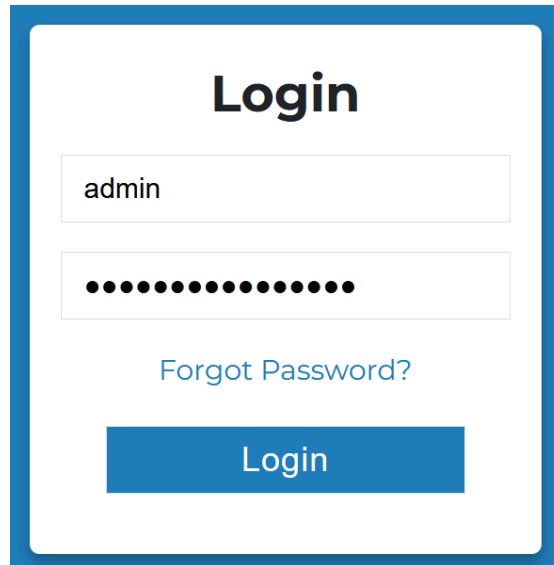
user table data definitions

The aforementioned reference tables in the data hierarchy went unused and are not currently implemented in the application. Source references for individual questions can still be added and are displayed on answer keys. The tables are ready should implementation work on the Data Hierarchy continue. In brief, subjects link to subject_reference table records. Topics link to topic_reference records which reference one of its parent subject records, and so on for learning_objective records. question sources as currently implemented do not have any relationships, and is a general text field for a specific page / section number in a reference.

User Manual

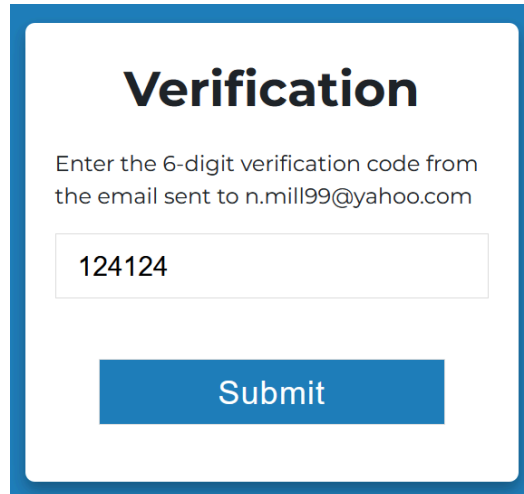
Login / Registration

To login to the application, navigate to the application url (wildcats.training or whichever domain the application is hosted at). If you aren't already logged in, the login form will display.

A screenshot of a login form. The form is titled "Login" in a large, bold, black font. Below the title, there are two input fields. The first input field contains the text "admin". The second input field contains a series of black dots, representing a password. Below the password field, there is a link that says "Forgot Password?" in a blue font. At the bottom of the form, there is a blue button with the text "Login" in white.

Enter your username and password and click Login. If you have forgotten your password, use the forgot password link and follow the instructions to reset your password.

When logging in, you will be prompted to enter a Two-Factor Authentication code. This is sent to the email attached to the account attempting to login. If you close the code entry window, please use the link the email sent to you on the same device you attempted to login on to reopen the page.



Verification

Enter the 6-digit verification code from the email sent to n.mill99@yahoo.com

124124

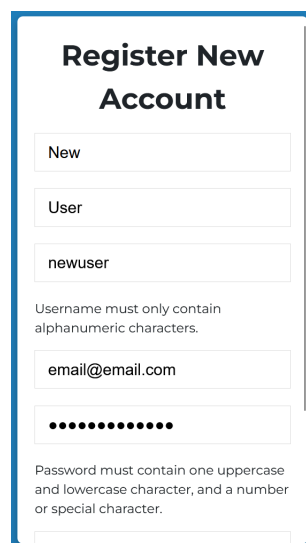
Submit

Upon providing the correct code, a user session will be enabled for your device and browser, and you will be redirected to the data hierarchy.

To register an account, you must be a logged in admin user. Navigate to the registration page using the top navigation bar.

Register Data Hierarchy Testing Test List Scoring Metrics Logout/Login

This will open a page to register a new account. Fill out the requested information, entering a starting password for the user you are signing up. Only select admin status for users you want to permit account creation on.



Register New Account

New

User

newuser

Username must only contain alphanumeric characters.

email@email.com

.....

Password must contain one uppercase and lowercase character, and a number or special character.

After completing the form successfully, you will be routed back to the data hierarchy page with a pop up success notification. The new user should navigate to their email, and click the provided link to activate their account. Successful verification will bring up the homepage with a success message. New users should use the forgotten password link to change their password from what was originally set by the creating admin.

Login

Account verified, you may now login

Training Data Hierarchy Editor

The Data Hierarchy Editor was designed to enable users to create and track training data in a hierarchical structure. This is split into subjects, topics, learning objectives, and questions which all store information regarding its parent in the hierarchy. In this section I will discuss how to navigate through the data hierarchy and how to use it.

Navigation

By selecting the Data Hierarchy button in the navigation bar you can be introduced to the home page of the Data Hierarchy Editor that displays all of the subjects available in the database including the subject ID, subject name, and the description as shown below.

Register Data Hierarchy Testing Test List Scoring Metrics Logout/Login



Data Hierarchy

Select a subject

Subject ID	Subject	Description
1	DOE - Mechanical Science	Pumps, valves, heat exchangers, filters, and other mechanical equipment.
2	DOE - Electrical Science	AC and DC fundamentals, circuits, batteries, generators, motors, reactive components, regulators, etc
3	Fish ladders	Fish ladder principles of operation, design, construction, and operating criteria
4	Bulk electric system	Principles of bulk electricity distribution.
5	Alarms, indications, and measurement devices	How pressure, flow, level, and other mechanical condition indicating devices work

Modifying Data

At the bottom of each page within the data hierarchy there are buttons to view and modify the data presented in the table. For buttons requiring a selector (View Topics, Edit, Delete) you must select the row entry prior to pressing the button. This can be seen below where the diagram is selecting a row, denoted by the blue highlight, and then you would click a button to perform that operation

Subject ID	Subject	Description
1	DOE - Mechanical Science	Pumps, valves, heat exchangers, filters, and other mechanical equipment.
2	DOE - Electrical Science	AC and DC fundamentals, circuits, batteries, generators, motors, reactive components, regulators, etc



Pressing the edit or delete buttons will display a browser popup that will include instructions for what is needed to perform the operation. Examples of occurrences can be seen below

[Register](#) [Data](#) [localhost:5000 says](#) [Logout/Login](#)

Are you sure you want to delete the selected record?

Data Hierarchy

Select a subject

Subject ID	Subject	Description
1	DOE - Mechanical Science	Pumps, valves, heat exchangers, filters, and other mechanical equipment.
2	DOE - Electrical Science	AC and DC fundamentals, circuits, batteries, generators, motors, reactive components, regulators, etc
3	Fish ladders	Fish ladder principles of operation, design, construction, and operating criteria
4	Bulk electric system	Principles of bulk electricity distribution.
	Alarms, indications, and	Flow, pressure, flow, level, and other mechanical

Add Button

Clicking the add button brings up a popup input field that allows you to enter a new Subject. After clicking OK another popup input field appears for inputting the description data associated with the subject.

The screenshot shows a web application interface with a blue header bar containing links: Register, Data, Topics, and Logout/Login. A modal window is open, titled "localhost:5000 says", with the prompt "Enter the new data:". The input field contains the text "Semiconductors". Below the input field are "OK" and "Cancel" buttons. The main content area is titled "Data Hierarchy" with the subtitle "Select a subject". It contains a table with the following data:

Subject ID	Subject	Description
1	DOE - Mechanical Science	Pumps, valves, heat exchangers, filters, and other mechanical equipment.
2	DOE - Electrical Science	AC and DC fundamentals, circuits, batteries, generators, motors, reactive components, regulators, etc
3	Fish ladders	Fish ladder principles of operation, design, construction, and operating criteria
4	Bulk electric system	Principles of bulk electricity distribution.
	Alarms, indications, and	How pressure, flow, level, and other mechanical

Below the table are four buttons: "View Topics", "Add", "Edit", and "Delete". The "Add" button is highlighted with a red square.

Edit Button

Using the edit button will first bring up a popup input field that contains the currently selected subject. After clicking the OK button a second popup input field will come up with the current description of the selected subject.

The screenshot shows a web application interface with a blue header bar containing links: Register, Data, Topics, and Logout/Login. The main content area is titled "Data Hierarchy" with the instruction "Select a subject". Below this is a table with three columns: Subject ID, Subject, and Description. The first row is highlighted in blue. Below the table are four buttons: View Topics, Add, Edit, and Delete. The Edit button is highlighted with a red square. A modal dialog is open over the table, showing a text input field with "DOE - Mechanical Science" and OK/Cancel buttons. The modal also displays "localhost:5000 says" and "Enter the updated subject:".

Subject ID	Subject	Description
1	DOE - Mechanical Science	Pumps, valves, heat exchangers, filters, and other mechanical equipment.
2	DOE - Electrical Science	AC and DC fundamentals, circuits, batteries, generators, motors, reactive components, regulators, etc
3	Fish ladders	Fish ladder principles of operation, design, construction, and operating criteria
4	Bulk electric system	Principles of bulk electricity distribution.
	Alarms, indications, and	High pressure, flow, level, and other mechanical

View Topics Add Edit Delete

Clicking the “View Topics” button after selecting a row will open its associated topics table, shown below.

Topic Hierarchy

DOE - Mechanical Science

Pumps, valves, heat exchangers, filters, and other mechanical equipment.

Topic ID	Topic	Description	Facility / Location Applicability
1	Diesel Engine Fundamentals	Provides information covering the basic operating principles of 2-cycle and 4-cycle diesel engines. Includes operation of engine governors, fuel ejectors, and typical engine protective features.	
2	Heat Exchangers	Describes the construction of plate heat exchangers and tube and shell heat exchangers. Describes the flow patterns and temperature profiles in parallel flow, counter flow, and cross flow heat exchangers.	
3	Pumps	Explains the operation of centrifugal and positive displacement pumps. Topics include net positive suction	

[View Learning Objectives](#)
[Add Topic](#)
[Edit Topic](#)
[Delete Topic](#)
[Back](#)

The topic hierarchy page introduces the “Back” button which will redirect back to the previous page that was accessed. Here, the same features rules apply will be available and row selection is required to view the learning objectives stored within each topic. After selecting a topic and clicking “View Learning Objectives” you will open the learning objectives stored within that topic as seen below.

Learning Objective Hierarchy

Diesel Engine Fundamentals Learning Objectives

Objective ID	Description	Bloom	Skill Applicability	Tags
1	DEFINE the following diesel engine terms: a. Compression ratio b. Bore c. Stroke d. Combustion chamber	Remember	View Training Level ▾	
2	Given a drawing of a diesel engine, IDENTIFY the following: a. Piston/rod b. Cylinder c. Blower d. Crankshaft e. Intake ports or valve(s) f. Exhaust ports or valve(s) g. Fuel injector	Understand	View Training Level ▾	
3	EXPLAIN how a diesel engine converts the chemical energy stored in the diesel fuel into mechanical energy	Understand	View Training Level ▾	

[View Questions](#)
[Add Objective](#)
[Edit Objective](#)
[Delete Objective](#)
[Back](#)

The learning objective table displays more data associated within each topic and has a “Skill Applicability” column which when clicked displays the training level associated with the rows objective.

Objective ID	Description	Bloom	Skill Applicability	Tags
1	DEFINE the following diesel engine terms: a. Compression ratio b. Bore c. Stroke d. Combustion chamber	Remember	View Training Level ▾	
2	Given a drawing of a diesel engine, IDENTIFY the following: a. Piston/rod b. Cylinder c. Blower d. Crankshaft e. Intake ports or valve(s) f. Exhaust ports or valve(s) g. Fuel injector	Understand	View Training Level ▾	



Skill Applicability

View Training Level ▾

- ☐ Applicant
- ☒ Apprentice
- ☒ Journeyman
- ☒ Senior
- ☒ Chief
- ☐ Coordinator

Questions Hierarchy

Lastly, when a row is selected from the learning objectives table and the “View Questions” button is clicked it will redirect to the questions table which displays questions for each learning objective.

Questions

Select a question to edit or create a new question

Question ID	Description	Type	Difficulty	Max Points
1		multiple choice	1	1
2		Short answer	2	5
3		Short answer	3	5
4		Short answer	4	5
5		Draw	5	5
6		Matching	1	8

Add
Edit
Delete
Back

The buttons work similarly to previous pages, but after selecting a row for a question and pressing the edit button another window will open and allow you to edit all content related to that question as seen below.

wildcats.training/editquestion?obj_id=1&question_id=1 - Google Chrome

wildcats.training/editquestion?obj_id=1&question_id=1

Edit question id 1 for learning objective:

DEFINE the following diesel engine terms: a. Compression ratio b. Bore c. Stroke d. Combustion chamber

Enter a question description (summary of a question without images, writing lines, and other content.)

Question Description

Enter the question contents. Displayed as is on tests including any other content. Use template buttons to get started, or start from scratch.

Multiple Choice Short Answer True/False

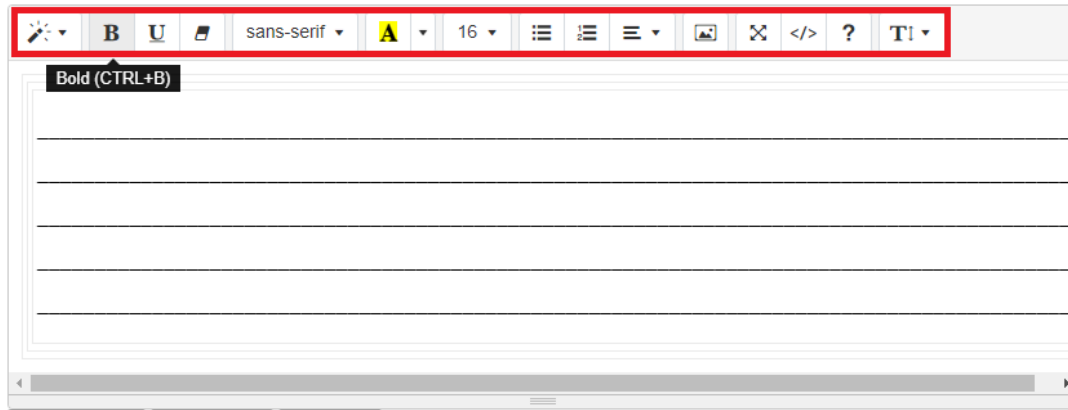
Enter the question answer. Displayed as is on test answer keys.

C) Nathan

Enter a question type (multiple choice, short answer, true / false, etc...)

multiple choice

You are able to change any data that will be populated based on the question selection. To edit the data you just need to select the input box for what you would like to edit and type what you want to change the data to. The summernote editor for question contents has many features that are explained by hovering the button that you would like information about as seen below.

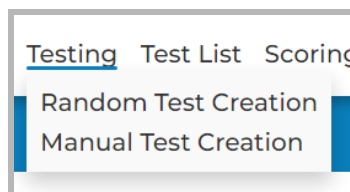
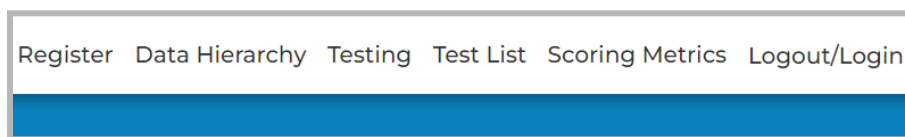


Test Creation

There are two different methods for creating tests: random and manual. The following section will go over how to navigate to the two different test creations and how to use them.

Navigation

Both methods are navigated to via the navigation bar, under testing. Hovering over “testing” on the navigation bar will show you the way to both “Random Test Creation” and “Manual Test Creation”. Simply click on the one you want to go to and it will take you there.



How the Filters Work

Both test creation methods utilize the use of filters to query the database for questions. The logic behind the filters is divided into two categories, ‘AND’ and ‘OR’. Think of the ‘AND’ filters as the requirements for a question to pass through. The ‘AND’ filters require the question to meet all criteria. On random test creation these filters include:

- Bloom's taxonomy (AND)
- Question type (AND)
- Question difficulty (AND)
- Training level (AND)

On manual test creation, these filters include the above with the addition of the "Keyword Search", which is an optional 'AND' filter.

On the other hand, the 'OR' filters are more like options. The 'OR' filters for both creation methods include:

- Subject (OR)
- Topic (OR)

Any question matching at least one of these criteria should be included. It's like browsing through different shelves in a library; you can pick any book from one shelf or another, and they're all relevant.

The decision behind the logic of the filters was to maximize usability so that anyone could use these filters to successfully query for questions, without having to have extensive knowledge of what topics and subjects are connected together while still incorporating strict filters to narrow the pool of questions down.

The queries are performed by a combination of the functions `handle_get_questions` (located in `wildcats.py`) and `get_questions` (located in `data_retrieval`).

Random Test Creation

Random test creation utilizes the strategy of creating a pool of questions via the different filters on the page. The filters include:

- Bloom's taxonomy (AND)
- Subject (OR)
- Topic (OR)
- Question type (AND)
- Question difficulty (AND)
- Training level (AND)

Filters are selected by clicking on the values you would like to apply to the query. Any number can be selected, including all or none. Selecting all or none does apply the same logic for the query, only selecting at least one value from each category will apply limitations to what is returned for that category. Below you can see an example of the filter selection. This styling is applied to all filters except for training level, which is a drop down located with the field inputs.

Random Test Creation

Select options to determine the pool of potential questions from the database:

Bloom's Taxonomy

Remember
Understand
Apply
Analyze
Evaluate
Create

Select All Categories

Subject

DOE - Mechanical Science

DOE - Electrical Science

Fish ladders

Bulk electric system

Alarms, indications, and measurement devices (mechanical)

Alarms, indications, measurement devices, and relays (electrical)

Relay protection schema

test

Once, the user has selected the filters they must also fill in the remaining fields:

- **Number of Questions:** This input determines how many questions will be on the randomly generated test. Only numerical inputs greater than 0 are accepted.
- **Test Max Points:** This input determines the maximum value the test can be. Generated tests will not exceed this value. Only numerical inputs greater than 0 are accepted.
- **Test Name:** Test names can be anything, as long as there is not already a test with that name.
- **Test Description:** Test descriptions can be anything.
- **Active:** Interact with that to set the test state to active or not (red is not active, green is active). This value is automatically set to not active.

Number of Questions

Test Max Points

Training Level

Select training level

Test Name

Test Description

Active?

Required fields and training level filter.

Generate Test

Button to start the test generation process.

All of the above fields are required to be filled out for the random test generation to proceed. Once the user has selected their desired filters and filled in the required fields, they must click the “Generate Test” button to the test generation process. On the previous page, you can see a screenshot of what these fields look like in the application. Optionally, you can choose to clear out all the inputs, this includes all filters and text/number fields. The button for this is located next to the “Generate Test” button at the bottom of the window, called “Reset All Inputs”. Note that there is no confirmation to check whether the user does indeed want to clear all inputs. Once clicked, all inputs are cleared.

Generate Test

Reset All Inputs

The next step of the process is querying the database using the selected filters and creating a pool of questions. From this pool of questions, x questions are chosen that have a combined sum less than or equal to y , where x is equal to the “Number of Questions” and y is “Test Max Points”. A series of validation checks are run on the pool of questions that is pulled using the selected filters. These checks include:

1. Check whether there are at least x number of questions in the question pool.
 - a. The user is told how many questions are in the pool, and then has the option to change their requirements.

2. If the first check fails, if there are less questions available in the pool than the user requests, an additional check is run to ensure that those questions' sum of points is less than or equal to y.
 - a. The user is told how many questions are in the pool and what their point sum is, and is then given the option to adjust their requirements.
3. Check whether there is a valid combination of x questions in the pool that will have a combined sum less than or equal to y.
 - a. Informs the user that there is no valid combination of the requested number of questions in the available question pool that will meet their test max points requirement and tells them to select different filters.

This validation and selection is performed in the functions `select_questions` and `validate_question_pool` found in `wildcats.py`.

After these validation checks are passed, x questions are chosen at random from the available pool of questions that have a combined sum less than or equal to y. The selected questions, their total score, the test name, test description, and active status is passed to the database and a test is created. This process is performed by a combination of the functions `handle_test_creation` (located in `wildcats.py`) and `test_creation` (located in `data_retrieval.py`). After successful completion of this process, the test is available whenever anyone with access to the application would like to view it, export it, or modify it. Test export will automatically launch after successful test creation in a new tab.

Manual Test Creation

Manual test creation works similar to random test creation, except instead of the computer randomly deciding what should be on the test, the user gets to decide this. The filters on manual test creation include:

- Bloom's taxonomy (AND)
- Subject (OR)
- Topic (OR)
- Question max point value
 - Returned questions cannot have a point value more than; All returned questions will have a point value equal to or less than this.
- Question type (AND)
- Question difficulty (AND)
- Training level (AND)
- Keyword search (AND)

To initiate the query for questions, the user must simply click on the "Search" button. Note that just like in random selection, a selection of one of the values must be selected in order to limit what is returned. Selecting all or nothing results in the same results, returning everything from that category. In addition, "Question Max Point Value" is a required field, and only accepts numerical inputs greater than 0.

Register Data Hierarchy Testing Test List Scoring Metrics Logout/Login

Manual Test Creation

Create Test >>

Bloom's Taxonomy
Select Bloom's Taxonomy

Topic
Select topic

Question Type
Select type

Question Point Max Value

Subject
Select subject

Training Level
Select training level

Question Difficulty
Select difficulty

Keyword Search

Search

Question query filters.

Point Value	Question Contents
<div style="border: 1px solid gray; padding: 5px; margin: 5px;"> <p>Button to query the database for questions that meet the filter criteria.</p> </div>	

The questions that meet the filter criteria are then populated on the table.

Register Data Hierarchy Testing Test List Scoring Metrics Logout/Login

Manual Test Creation

Create Test >>

Bloom's Taxonomy
Select Bloom's Taxonomy

Topic
Select topic

Question Type
Select type

Question Point Max Value

Subject
Select subject

Training Level
Select training level

Question Difficulty
Select difficulty

Keyword Search

Search

Table that populates with queried questions

Point Value	Question Contents	Select
<div style="border: 1px solid gray; padding: 5px; margin: 5px;"> <p>Button to query the database for questions that meet the filter criteria.</p> </div>		<div style="border: 1px solid red; padding: 2px; margin: 5px;">Add</div>

From the table the user can view the number of points that a question is worth and preview of the contents of the question. The user can then select any number of those questions via the checkbox in the third "Select" column, and then scroll to the bottom of the page and click the "Add" button.

Point Value	Question Contents	Select
3	Which of the following is NOT a factor that affects a diesel engine's compression ratio?	<input type="checkbox"/>
3	What is the typical range of compression ratios for diesel engines?	<input type="checkbox"/>
5	What is the main function of the combustion chamber in a diesel engine?	<input type="checkbox"/>
5	Describe the difference between bore and stroke in a diesel engine:	<input type="checkbox"/>
5	Explain how the size of the bore and stroke can affect the power output of a diesel engine.	<input type="checkbox"/>
5	What are one of the applications of heat exchangers?	<input type="checkbox"/>
10	Where is the piston on this engine?	<input type="checkbox"/>

Check boxes to select questions to be added.

Button to add selected questions to the side panel.

Add

Any added questions can be viewed on the side panel which is accessible by clicking on the “create test” button that is located in the top right corner of the page. The user can perform any number of queries for questions and add any number of questions from those queries to the side panel.

Register Data Hierarchy Testing Test List Scoring Metrics Logout/Login

Manual Test Creation

Bloom's Taxonomy

Select Bloom's Taxonomy

Topic

Select topic

Question Type

Select type

Question Point Max Value

Subject

Select subject

Training Level

Select training level

Question Difficulty

Select difficulty

Keyword Search

Search

Button to open the test creation side panel.

Create Test >>

Questions can be viewed and removed from the side panel. Removal simply requires clicking on the remove button for the corresponding row. There is no confirmation to confirm the user wants to remove it, but any removed questions can simply be re-added the same way they were added originally.

The screenshot shows a web interface for selecting questions. A table titled "Selected Questions" is highlighted with a red border. To its right, a callout box states: "Table where selected questions appear." Below the table, there are input fields for "Test Point Value" (containing 25), "Test Name", and "Description". To the right of these fields, another callout box states: "Buttons to remove a question from the corresponding row." This callout points to four red "Remove" buttons, each aligned with a row in the table.

Order	Point Value	Question Contents
<input type="text"/>	5	Describe the difference between bore and stroke in a diesel engine:
<input type="text"/>	5	Explain how the size of the bore and stroke can affect the power output of a diesel engine.
<input type="text"/>	5	What are one of the applications of heat exchangers?
<input type="text"/>	10	Where is the piston on this engine?

Test Point Value:

Test Name:

Description:

Buttons to remove a question from the corresponding row.

Once the user is satisfied with the questions they have selected and are present on the side panel, all they need to do is fill out the remaining fields:

- Order of questions: This is filled out in the first column of the table in the side panel. This will control the order the questions are presenting on the test. Valid inputs are one through the number of questions present on the table, no duplicate numbers. Example: If there are ten questions on the table, the user must fill in each input with a value one through ten.
- Test name: Test names can be anything, as long as there is not already a test with that name.
- Test description: Test descriptions can be anything.
- Active: Interact with that to set the test state to active or not (red is not active, green is active). This value is automatically set to not active.

The user must fill in all the fields in order to make a test.

Selected Questions

Order	Question Contents	
<input type="text"/>	the difference between bore and stroke in a diesel engine:	<button>Remove</button>
<input type="text"/>	5 Explain how the size of the bore and stroke can affect the power output of a diesel engine.	<button>Remove</button>
<input type="text"/>	5 What are one of the apangers?	<button>Remove</button>
<input type="text"/>	10 Where is the piston on	<button>Remove</button>

Test Point Value **Test Name** **Active?**

25 ☒

Description

Create Test

Button to start the test creation process.

Once the user has finished selecting questions and filled in the required fields, they must click the “Create Test” button to initiate the test creation process to save their newly crafted test in the database. After successful completion of this process, the test is available whenever anyone with access to the application would like to view it, export it, or modify it. Test export will automatically launch after successful test creation in a new tab.

Error Handling

The current code is equipped to deal with multiple possible errors. Both test creation methods are coded to handle the following errors:

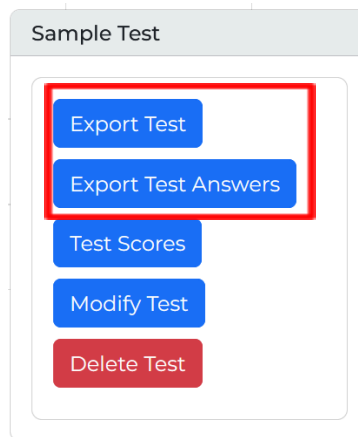
1. No questions meeting the query criteria.
 - a. Informs the user of this and tells them to change their filters.
2. Duplicate test name.
 - a. Informs the user of this and tells them to pick a different name.
3. Any error not explicitly planned for.
 - a. Generic message is given to the user. Mostly just used to stop the application from crashing. Future development of this application could improve and expand this.

Random test creation additionally has these three, mentioned in its section.

1. Handle the case that there are not enough questions in the question pool to meet the user defined requirement of the test having x questions.
 - a. The user is told how many questions are in the pool, and then has the option to change their requirements.
2. That there are not enough questions in the question pool and that those questions' max point sum in the pool exceed the user defined requirement of the test having a max point total equal to or less than y.
 - a. The user is told how many questions are in the pool and what their point sum is, and is then given the option to adjust their requirements.
3. That there is no valid combination of x questions in the pool that will have a combined sum less than or equal to y.
 - a. Informs the user that there is no valid combination of the requested number of questions in the available question pool that will meet their test max points requirement and tells them to select different filters.

Test and Answer Key Export

To access test and answer key exports, select the Test List from the top navigation bar. This will display a list of all tests, along with filtering options. To export: click on a test, then select Export Test or Export Test Answers.



This will pop up a new window and prompt your browser's print function. You can directly print the tests out, or save to PDF using the print to PDF option.

Name: _____

Points: _____ / 51

Date: _____

Sample Test**Question 1:**

Which of the following is NOT a factor that affects a diesel engine's compression ratio?

- ☐ Ⓐ The volume of the cylinder at its smallest size
- ☐ Ⓑ The volume of the combustion chamber
- ☐ Ⓒ The type of fuel used
- ☐ Ⓓ The length of the stroke

Points earned: _____ / 3

Question 2:

Where is the piston on this engine?



Points earned: _____ / 10

Answer Key**Total Points Possible: 51****Sample Test****Question 1:**

Which of the following is NOT a factor that affects a diesel engine's compression ratio?

- ☐ Ⓐ The volume of the cylinder at its smallest size
- ☐ Ⓑ The volume of the combustion chamber
- ☐ Ⓒ The type of fuel used
- ☐ Ⓓ The length of the stroke

Max Points Possible: 3

Answer:

- ☐ Ⓒ The type of fuel used

Answer explanation:

N/A

Points Rubric:

All or nothing

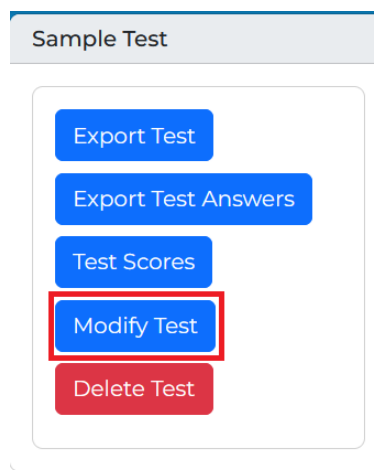
Source: N/A

Exported Tests contain a header with blank lines for Name, Date, and Points along with the test title and question contents.

Exported Answer Keys contain a simple header with the test name and total possible points, along with the full question contents including the original question, answer, answer explanation, points rubric, and reference source.

Test modification

This page allows testers to modify tests that are already in the database. Users can change the name of the test, its description, the order of questions, and add or remove questions. To access this page on "Test List" in the top navigation bar. Once on the Test List page, click on the test you would like to modify. This will display five options, including a "Modify Test" button. Click on the "Modify Test" button to navigate to the Modify Test page.



On the Modify Test page, you'll find similar features as the manual test creation page, with the addition of a side panel populated with questions from the selected test to be modified. Follow the steps provided in the Manual Test Creation User Guide to make changes to the questions listed. Once you are satisfied with the changes made to the questions, click on the "Update Test" button to modify the test in the database.

Test Delete

To delete a test, navigate to the 'Test List' in the top navigation bar to view a list of generated tests in a table. Click on the row corresponding to the test you wish to delete, then select the 'Delete Test' button.

This action will remove the test from both the database and the displayed list on the page. If any tester score records exist for the test, an error message will appear, indicating that these records must be deleted before the test can be removed. Once all the tester records for the test are removed, the test can be successfully deleted.

Test Score Entry and Management

This page is designed to allow users to manage a list of testers and their scores for specific tests by adding, editing, and deleting testers' records. This section will explain how to navigate through the pages and how to use its features.

Navigation

By selecting 'Test List' from the navigation bar, you can access a list of tests that have been already generated. When you hover the mouse over each test row, the background of the row changes to indicate that it is clickable. Clicking on a row opens a pop-up menu for that test, allowing you to navigate to the 'Tester List' page by clicking on 'Test Scores' in the pop-up menu.

Register Data Hierarchy Testing Test List Scoring Metrics Logout/Login



Test List

Subject

Topic

From

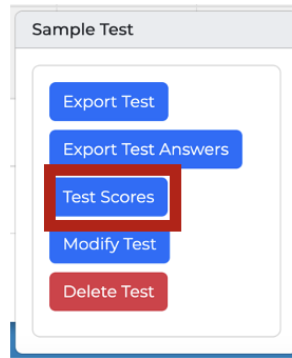
To

Search by name

No.	Test Name	Subject	Topic	Created Date	Modified Date
1	Sample Test	DOE - Mechanical Science	Diesel Engine Fundamentals, Heat Exchangers	0000-00-00 00:00:00	0000-00-00 00:00:00
2	testDima	DOE - Mechanical Science	Diesel Engine Fundamentals	2024-02-22 21:41:58	2024-02-22 21:41:58
3	picture test 10	Test Subject	Pictures	2024-03-05 06:30:49	2024-03-05 06:30:49

Rows can be clicked.





How to manage tester's score

This is the page for viewing, adding, editing, and deleting testers' score records.

Tester List

	/	No.	Tester Name	Attempt Date	Score	Pass Status	
5	▼	1	name1	2024-03-01 09:13:00	N/A	FAIL	Add New Record
	▼	2	name2	2024-03-05 20:13:00	90	PASS	Add New Record
	1		name2	2024-03-05 20:13:00	90	PASS	Delete
	2		name2	2024-02-29 22:13:00	60	FAIL	Delete

Add New Tester

1~2: Adding

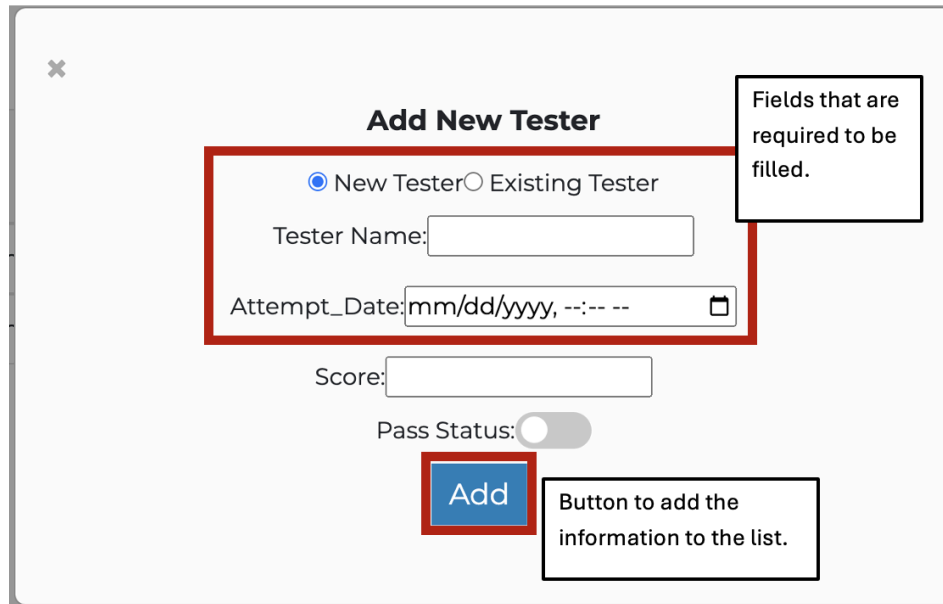
3: Deletion

4: Fields can be modified

5: Dropdown button

1. 'Add New Tester' Button

- a. When the button is clicked, a new page pops up for users to fill out the new tester's information.
- b. On the new pop-up page, users can choose between 'New Tester' and 'Existing Tester' options. 'New Tester' is for adding a new tester who has not been previously recorded in the system. 'Existing Tester' is for adding a tester who has been added before or is already present in other tests.
- c. If the score input field is left blank, it will be displayed as 'N/A' on the list and can be modified at any time.



Add New Tester

☒ New Tester ☐ Existing Tester

Tester Name:

Attempt_Date: mm/dd/yyyy, --:-- --

Score:

Pass Status: ☐

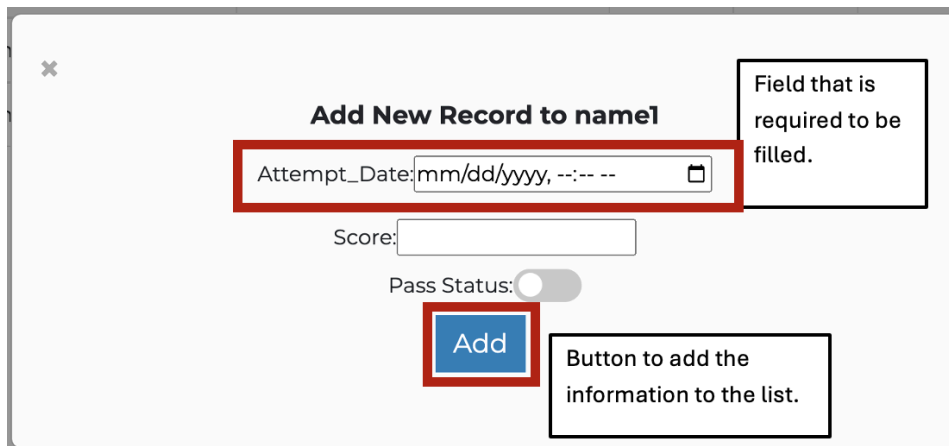
Add

Fields that are required to be filled.

Button to add the information to the list.

2. 'Add New Record' Button

- When the button is clicked, a new page pops up for users to fill out the selected tester's new record.
- It allows users to input attempt date, score, and pass status.
- It also displays 'N/A' on the list when the score input field has no input.



Add New Record to name1

Attempt_Date: mm/dd/yyyy, --:-- --

Score:

Pass Status: ☐

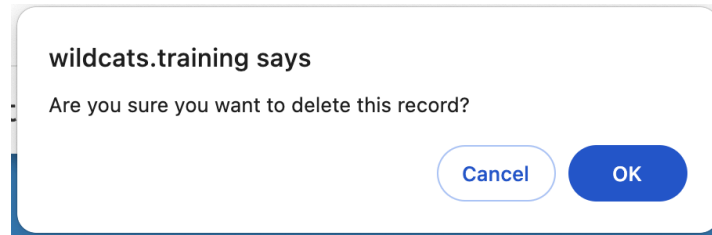
Add

Field that is required to be filled.

Button to add the information to the list.

3. 'Delete' Button

- Testers' records can be deleted by clicking the 'Delete' button, with a confirmation message appearing for double-checking before deletion.



4. Editing Attempt Date, Score, and Pass Status
 - a. Each field for 'Attempt Date', 'Score', and 'Pass Status' is editable.
 - b. When users hover mouse over the field, the background color changes to indicate it is clickable.
 - c. Clicking on the field allows users input their desired value for editing.
 - d. The fields can be edited individually.

1	name2	03/05/2024, 08:13 PM Edit	90 Edit	<input checked="" type="radio"/> PASS <input type="radio"/> FAIL Edit	Delete
---	-------	--	----------------------------	---	------------------------

5. Dropdown button to display the tester's all records
 - a. When the dropdown button is clicked, all of the tester's records are displayed from the latest attempt date to the oldest.
 - b. Tester records can also be modified directly within this dropdown table.

▼	2	name2	2024-03-05 20:13:00	90	PASS	Add New Record
1		name2	2024-03-05 20:13:00	90	PASS	Delete
2		name2	2024-02-29 22:13:00	60	FAIL	Delete

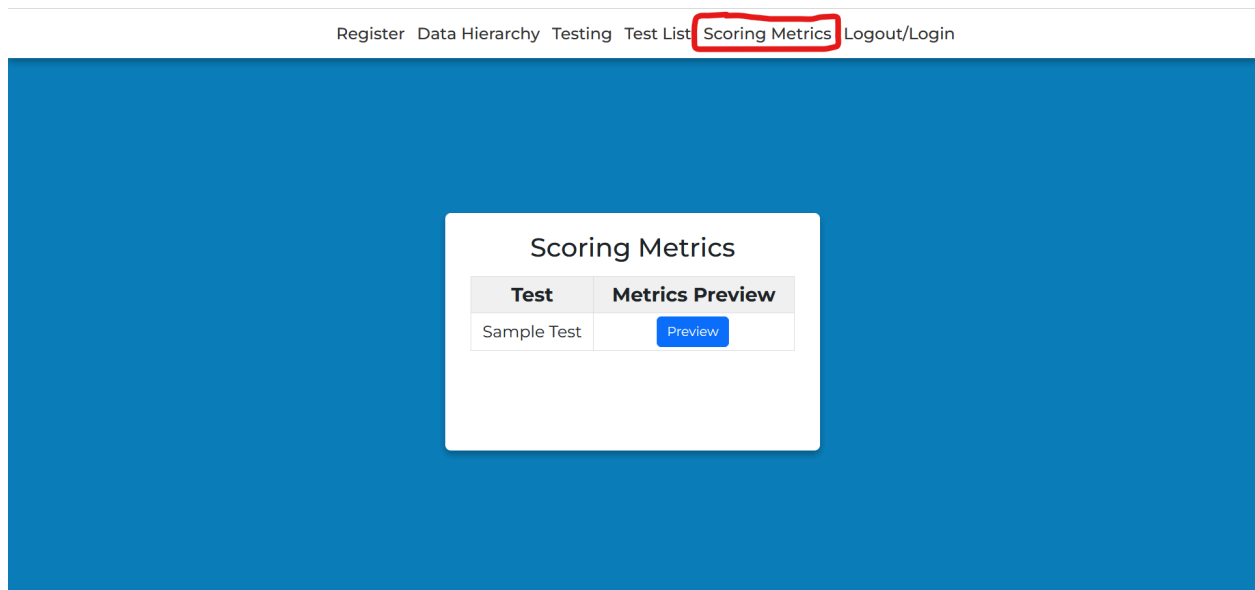
Test Score Metric Export

This page provides users with insights into how test takers scored in a particular test. It generates a PDF report including:

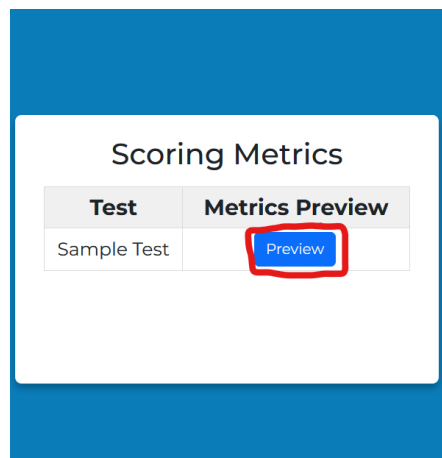
- Test name
- Test ID
- Number of time the test was taken
- Number of testees who passed the test

- Graph of score distribution
- Box plot of scores with mean score indicated
- Mean score
- Median score
- Upper quartile
- Lower quartile
- Highest score
- Lowest score

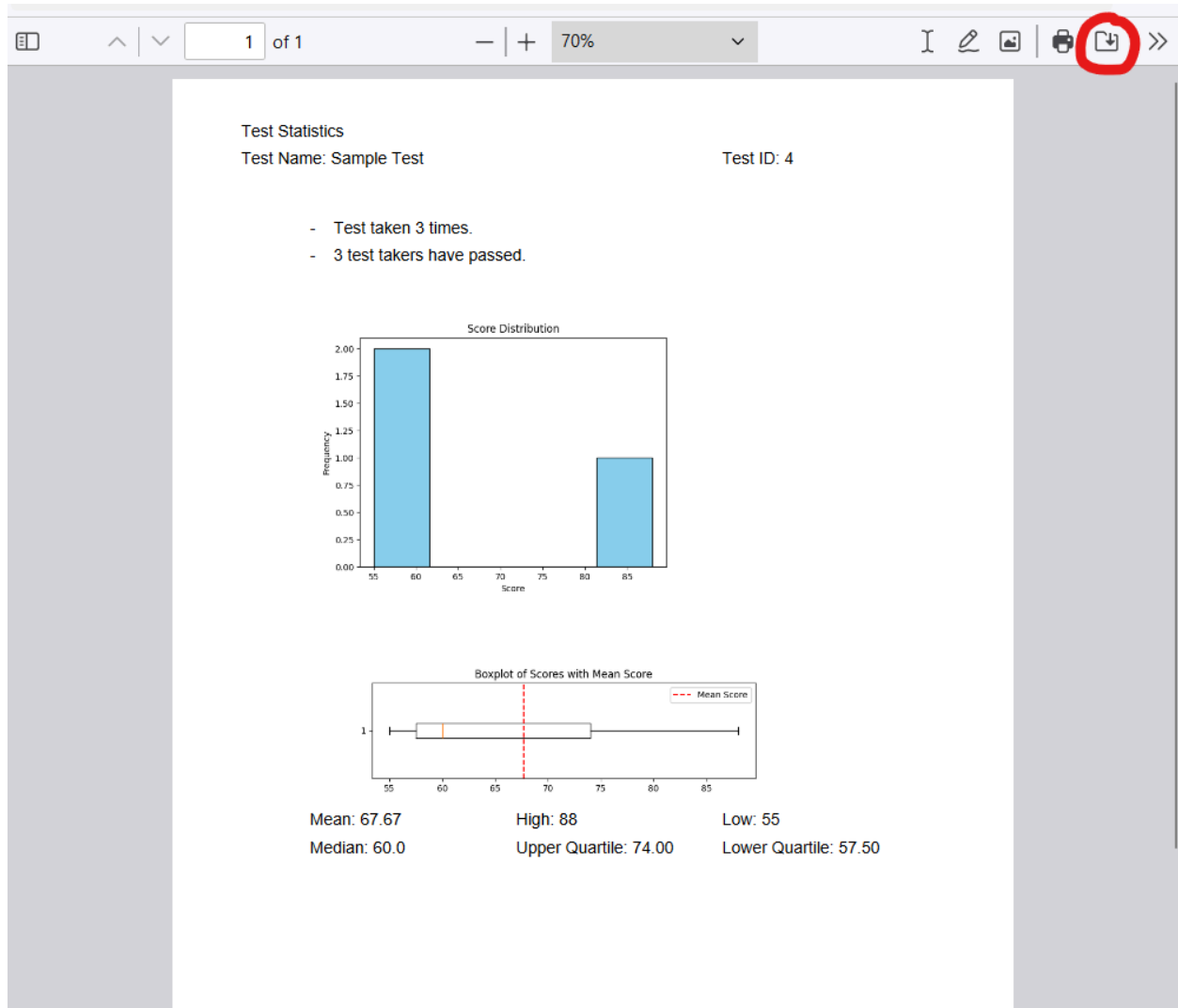
To access this page, users must click on "Score Metric" on the top navigation bar.



After clicking on "Score Metric," users will be directed to a page displaying a list of tests in the database. To preview the PDF containing statistics for a specific test, users must click on the "Preview" button next to the respective test.



The preview will display a PDF with the statistics, as shown in the figure below. To export the PDF, users should click on the save to file symbol within the PDF viewer and follow the instructions to save the file.



Development Lifecycle

Development Methodology

We didn't use a specific methodology. We basically broke into different sub teams:

- Database: Nathan and Kat
- Frontend: Angel
- Server Administration: Geoff
- Backend: Hiruy and Simon

Initially, we stuck to our respective assignments, but later the assignments mattered less and less and we just handed out tasks that needed to be accomplished. Each team member ended up essentially taking up and creating entire features:

- Login/Registration: Nathan, Simon, Hiruy
- Data hierarchy: Angel, Geoff, Nathan
- Test Creation: Kat
- Test Modification Hiruy, Kat
- Test Export: Nathan
- Scoring Metrics: Hiruy
- Test List: Simon, Nathan

We provided assistance to other teammates when requested and met frequently to touch base on how development was going and what we needed to do next in a weekly scrum meeting. We were also in constant communication through discord text channels providing feedback.

Initial Testing Feedback and Software Improvements

Originally, when sending progress to the client on the state of the project, we received feedback that was very important to the further development of the project that would tailor to the clients needs. The list of feedback can be summarized as:

- Popup confirmation for modifying data.
- Moving the button container to a location that doesn't require scrolling.
- Adding navigation breadcrumbs to revert to old pages.
- Solve user interface issues that result in text spilling, overlapping buttons, and diverse styling.
- Containers in user interface overlapping other features.
- Test creation not working with errors in requests.
- Pop up confirmation before deleting items for all pages.
- Enabling editing existing questions.

We took all comments into consideration and began working as a team delegating members to fix the issues outlined. Over a couple of weeks we successfully solved all issues highlighted and ran it by the client a second time towards the end of the deadline. We received approval for the final project and the client was happy with our changes from the suggestions.

Hardships

- Balancing other classes and work with all the time needed to put together the project.
- A few particularly difficult to find runtime errors that took multiple people multiple hours to solve.