Software Design Description (SDD) for the Computer Science Question Input and Delivery Template Manager (C-SQID-TM)

Document Number: CMSC447-FA2020-G02-SDD-01A
Revision A

27 October 2020

Authors:
Ryan Appleby
Charles Varga
Aleya Mayo
Amrithya Balasubramanian
Chris DeVoe
Danada Amalage Don

Version History

Date	Version	Description	Authors
10/27/2020	1.0	Initial Version for Submission	Team

1. Scope

1.1 Identification.

This Software Design Document pertains to the development of the Computer Science Question Input and Delivery Template Manager (C-SQID-TM) application and backend question database powering said application. The intended audience includes the engineers tasked with developing the application and its database as well as the test conductors responsible for the verification and acceptance of the application.

1.2 System overview.

The C-SQID-TM application can interface with the backend C-SQID-TM database to store, retrieve, and update the information within the database. The user can manually enter information to both store and retrieve information. The user may also submit a local document to be parsed and stored in the database. In addition, the application allows the user to input values into retrieved questions and save said questions locally without storing them in the database, and will allow the user to export this output to a separate file. The primary user for this application would be Computer Science professors and teaching assistants.

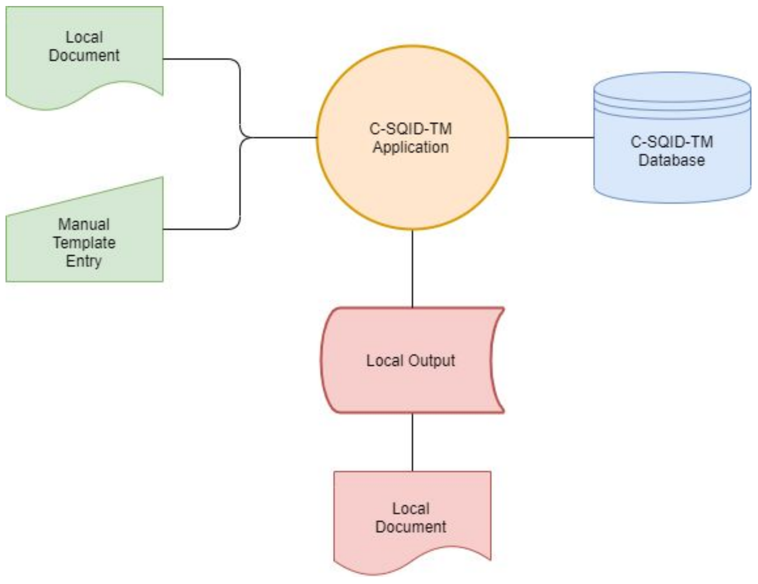


Figure 1. C-SQID-TM Interface

The software outlined in this SDD is responsible for interfacing with the user's local machine as well as the C-SQID-TM database in order to store and retrieve information. The software's overall state transition is shown in Figure 2.

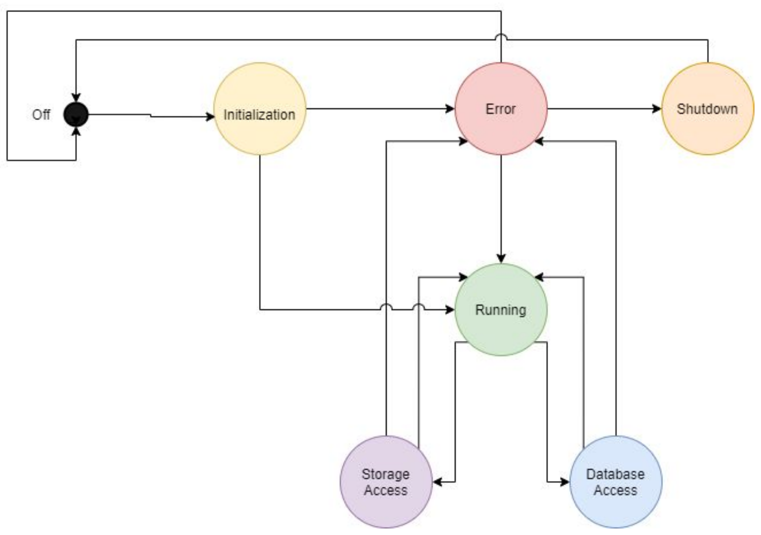


Figure 2. C-SQID-TM State Diagram

1.3 Document overview.

Section 1 describes the scope of the project. Section 2 provides the documents references. A brief overview of design decisions are detailed in Section 3. Section 4 details architectural design choices, while Section 5 goes into greater detail about design decisions. Requirements Traceability is included in Section 6.

2. Referenced Documents

MIL-STD-498	Military Standard Software Development and Documentation
CMSC447-FA2020-G02-SDP-01A	Computer Science Question Input Template and Delivery Manager - Software Development Proposal
CMSC447-FA2020-G02-SRS-01A	Computer Science Question Input Template and Delivery Manager - Software Requirements Specification

3. CSCI-wide design decisions

3.1 User Access Mode Selection

To provide a user friendly interface for the customer, C-SQID-TM is expected to operate on a web-based server-client architecture. The code that runs the front end and the back end of the application shall be stored on a remote server if possible. Otherwise, it shall be stored on a local server. The database shall be stored in a remote server that takes advantage of cloud computing services such as Amazon Web Services, Google Cloud, etc.

3.2 User Characteristics

Users of C-SQID-TM are assumed to possess basic familiarity with web browsers. They are also assumed to be either TAs or professors at the receiving university. All user interaction shall occur through browser based windows which inform the user of the purpose of the pages in view.

3.3 Constraints

The purpose of this web application is to provide a cross-platform, online interface to upload, pull, and manage question bank templates from a centralized source database. There are limitations and user commitments regarding the usability of the templates produced. The constraint factors are summarized below:

- An uploaded template shall require users to manually fill in blanks within the question banks. Since the product is not meant to be a means of version control, multiple users/professors cannot edit the templates collaboratively.
- Students shall not be able to download templates or question banks uploaded by professors without the required permissions.
- The question difficulty classification feature (if implemented) may not provide accurate data sets as the application will not support question answering features within the application as of the current status of the project. Any performance metrics and/or feedback shall be self-reported by the students and may consequently prove biased.

Authentication and security constraints:

- Creation of administrative professor accounts cannot be strongly validated by the application. Professors/instructors shall be added by other pre-authorized staff. The application does not support security features to implement academic identity verification through formal means.
- Even if students do not have share or download permissions, question bank templates and questions can still leak. Professors shall manually enforce academic integrity for sensitive documents and incorporate appropriate protocols to conduct secure release of the documents if this application will be used for examination purposes.
- As such, the authentication and security will be largely reliant on the proprietors administering this web application.

Proprietary control:

- Professors may be limited in the way they can use question banks/templates committed to the system by the management setting the committing professor/staff has set. Their submission shall be treated as intellectual property and their proprietary rights will be protected by limiting the use of their submitted template by other professors according to whatever terms of use conditions are set as constraints.

3.4 Assumptions and Dependencies

This application is built as a web application using the Flask framework and stylistic scripting languages such as HTML and CSS. As such, not many dependencies are present from the user end other than an internet connection. This includes assuming that users have an adequate web browser, such as Chrome or Firefox. A Google account/myUMBC login info is also required for user authentication. The application will also require a connection with a remote database, either structured like SQL or unstructured like NoSQL. The exact database platform to be used shall be determined further in the design process.

4. CSCI architectural design

4.1 CSCI components

There are several software units that make up the C-SQID-TM CSCI. Each software component satisfies one or more of the CSCI's requirements. This traceability is provided in Section 6. Several of these components inherit from each other. This inheritance is demonstrated in Figure 3.

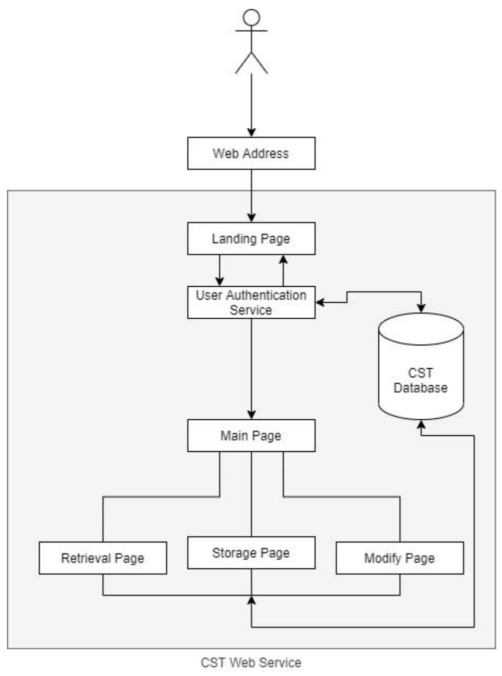


Figure 3. C-SQID-TM Software Unit Interface Inheritance

The software components are as follows.

4.1.1 CST Web Service

The CST Web Service is responsible for maintaining a connection to the remote CST Database as well as the web server that the main software units will run on. It handles errors and exceptions that occur with the web connection, and provides basic security features to prevent malicious code from being executed within the CST software.

4.1.2 CST Database

The CST Database is a MySQL database used for both user verification and question storage and retrieval. The CST database interfaces with the Retrieval Page, the Storage Page, the Modify Page, and the User Authentication service. The primary interactions between these units and the CST Database are illustrated in Figure 4.

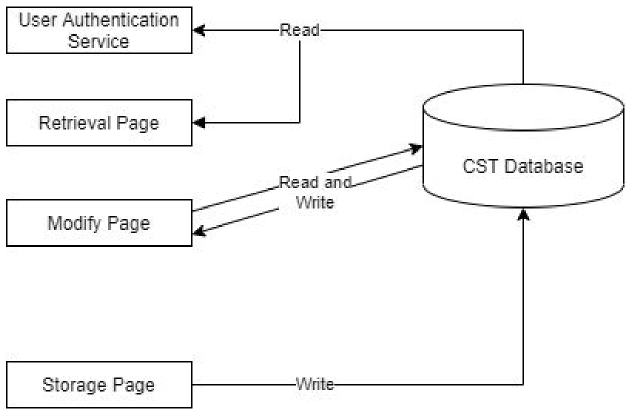
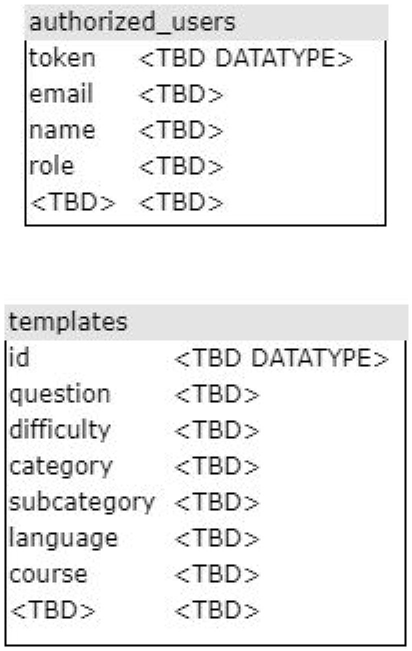


Figure 4. Primary CST Database Interactions

The primary tables and fields of the database are shown below in Figure 5.



authorized_users	
token	<TBD DATATYPE>
email	<TBD>
name	<TBD>
role	<TBD>
<TBD>	<TBD>

templates	
id	<TBD DATATYPE>
question	<TBD>
difficulty	<TBD>
category	<TBD>
subcategory	<TBD>
language	<TBD>
course	<TBD>
<TBD>	<TBD>

Figure 5. CST Database Design

4.1.3 CST Landing Page

The CST Landing Page is the first page the user sees when accessing the C-SQID-TM web address. The page will provide a way to login to the CST system. In addition, it will be the page that users are redirected to should they be unauthorized or otherwise unable to access the C-SQID-TM database and software. The Landing Page will also provide instructions for how to contact the system administrator so that access can be granted to the database and software.

4.1.4 CST User Authentication Service

The CST User Authentication Service will authorize users after they log in to ensure that the database is accessible to them. This is a two layer process. First, the user logs in with their UMBC Google Account credentials through the Google Authentication API. Then, a token from their user account will be sent to the CST Database. This token is verified to be in the authorized_users table of the database. If either of these tests fail, the user is sent back to the CST Landing Page. If both succeed, the user is sent to the CST Main Page. This interaction is illustrated in Figure 6.

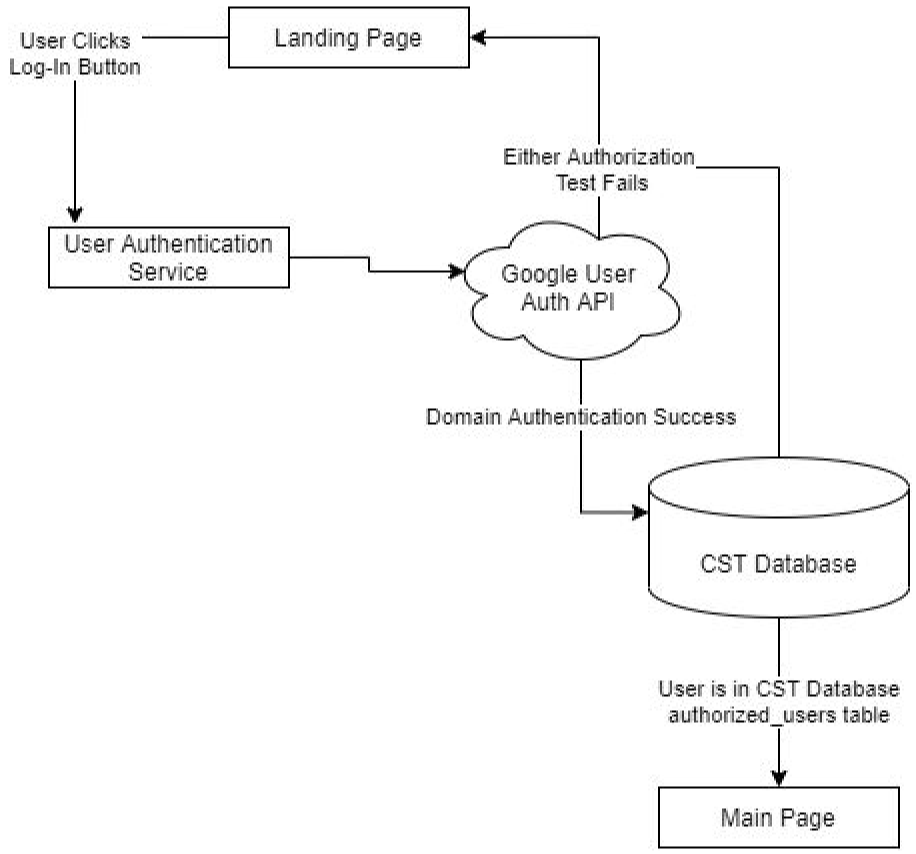

Landing Page
User Clicks Log-In Button
User Authentication Service
Google User Auth API
Either Authorization Test Fails
Domain Authentication Success
CST Database
User is in CST Database authorized_users table
Main Page

Figure 6. CST User Authentication Service

4.1.5 CST Main Page

The CST Main Page is the parent software unit for the CST Retrieval Page, the CST Stoage Page, and the CST Modify Page. These four software units can be navigated to from the CST Main Page's Side Navigation panel. All three child software units inherit this side bar and can use it to navigate between pages, much like the parent software unit. This is illustrated in Figure 7.

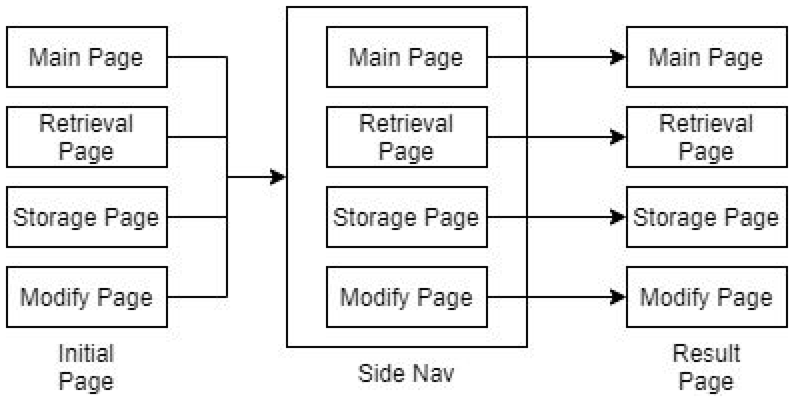


Figure 7. CST Main Page Side Navigation Panel (Side Nav)

4.1.6 CST Retrieval Page

The CST Retrieval Page is a child software unit of the CST Main Page. This software unit provides a way for the user to interface with the CST Database and retrieve question templates. It allows the user to enter values into the template fields, and copy the retrieved template to the clipboard for later use.

4.1.7 CST Storage Page

The CST Storage Page is a child software unit of the CST Main Page. This software unit provides a way for the user to interface with the CST Database and store new question templates. It allows the user to either manually enter template(s) to upload to the database, or to upload a file to the web application to parse, interpret, and enter into the database.

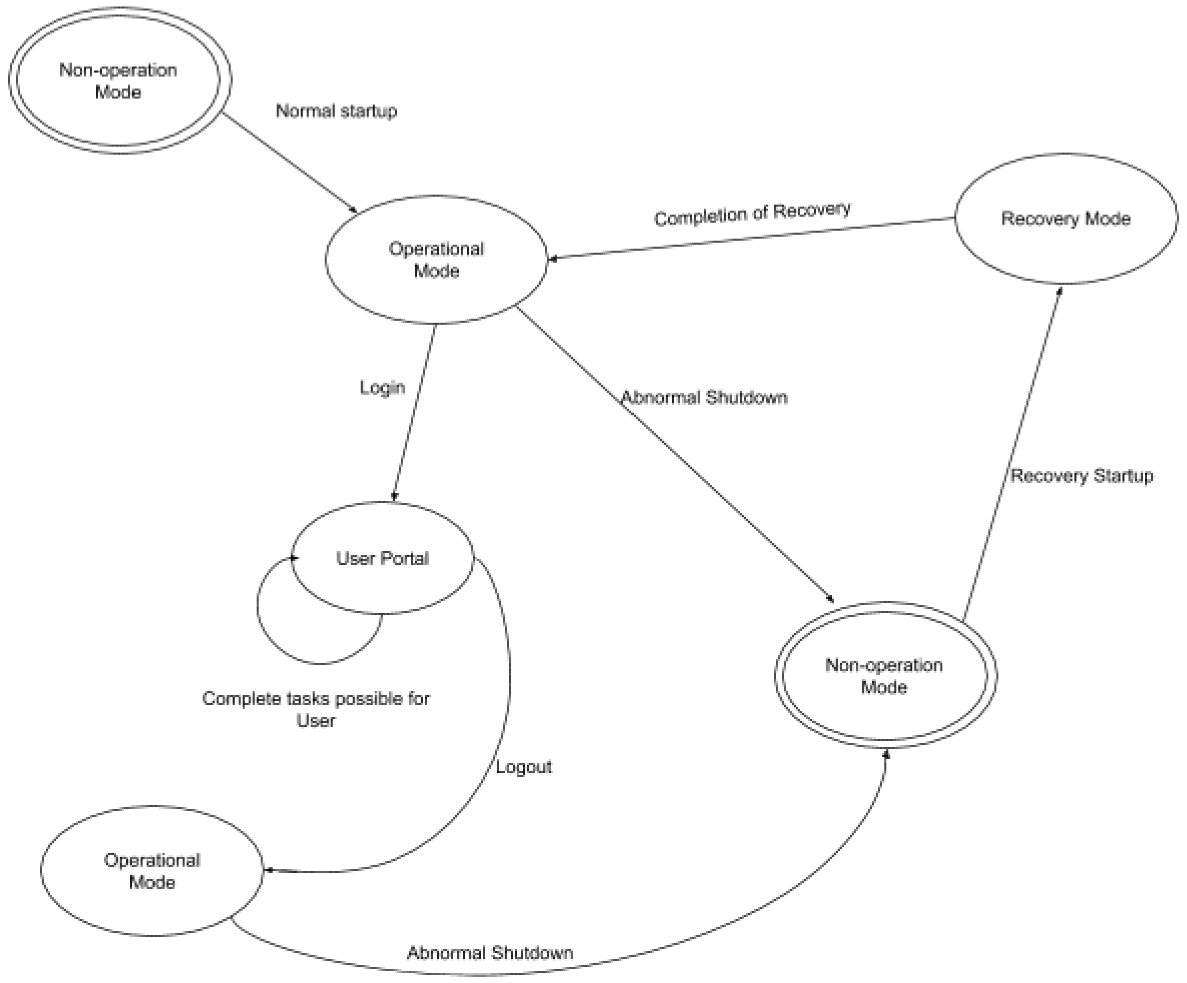
4.1.8 CST Modify Page

The CST Modify Page is a child software unit of the CST Main Page. This software unit provides a way for the user to interface with the CST Database and modify existing question templates. The user will be to edit the question text, the question's difficulty, the question's category and subcategory, and the question's programming language. The user will also be able to delete and remove templates from the database.

4.2 Concept of execution

This section will analyze application C-SQID-TM as a concept of execution by defining and analyzing the states and modes the application would be on during its operational phase.

- Startup of the C-SQID-TM
- Runtime Operation
- Dynamic Relationship between components
- System Monitoring
- Shutdown of C-SQID-TM



4.2.1 Startup of the C-SQID-TM

- Normal Startup
  Normal startup is when the application runs as intended and all components of the application are working correctly. Components such as the CST database are active and interact with the web application. It also means users can login to the application and interact with template files without encountering any errors.

- Recovery Startup
  Recovery startup occurs when the web application fails to function properly, which can be a result of a glitch or bug in the application. This would likely lead to an abnormal shutdown, and once the issue is resolved the application is returned to the last active state.

4.2.2 Runtime Operation

Once the application has startup and is in idle phase, it can continue with following operations.

1) Login phase where user authentication is needed to access their private portal. Token from the user's account will be sent to the CST database to be verified.

2) Once login is complete, the user will be navigated to the CST main page, the parent software unit of the child software units CST Retrieval page, CST Storage Page, CST Modify Page. The user can utilize the side navigation panel to access parent software unit and child software units. By accessing these units the user will be able to perform operations such as.

    - Retrieve template file: In the CST Retrieval page users can retrieve template files from the CST Database.
    - Store template file: In the CST Storage page users can create new template files.
    - Modify template file: In the CST Modify page users can modify existing template files.

4.2.3 Dynamic Relationship between components

- Database interaction for when logging in to the web application.
- User interactions within their respective portal
- Database interaction to retrieve, store, and modify template files.

4.2.4 System Monitoring

Throughout the lifecycle of the web application, there will be patches and bug fixes that would be integrated into the application to create a better experience with login and user portal.

4.2.5 Shutdown of C-SQID-TM

Shutdown of C-SQID-TM is when the web application is closed.

4.3 Interface Design

4.3.1 Interface identification and diagrams

The goal of this section is to narrate the characteristics of the interface for C-SQID-TM.

As seen in Figure 9, C-SQID-TM is built using a client-server architecture with the help of a database. In this model, the user uses a browser to talk to the Web API server, which in return, responds with data from the database. Figure 10 demonstrates the user authentication system used in C-SQID-TM.

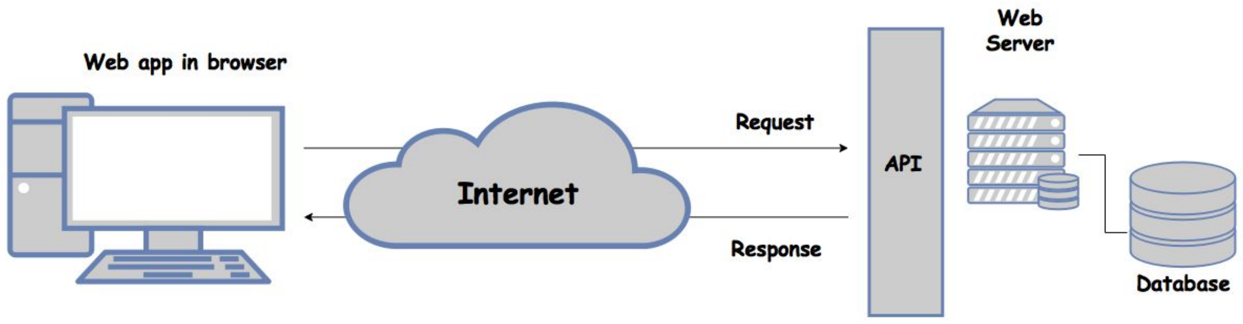


Figure 9. Flow Diagram From User's Browser to Database

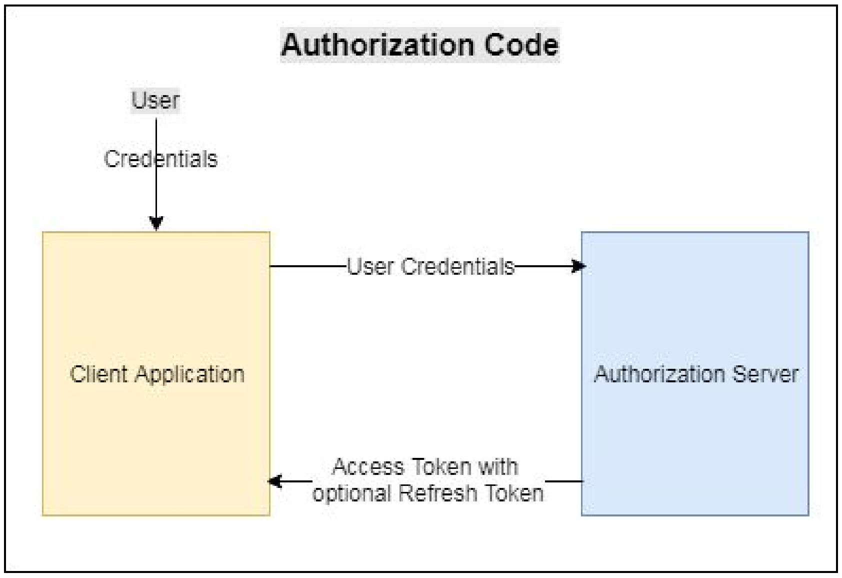


Figure 10. User Obtaining Authorization From Client

Figures 11 and 12 are an activity diagram and a sequence diagram, respectively, to demonstrate the data and control flow in a typical user's interaction with our software.

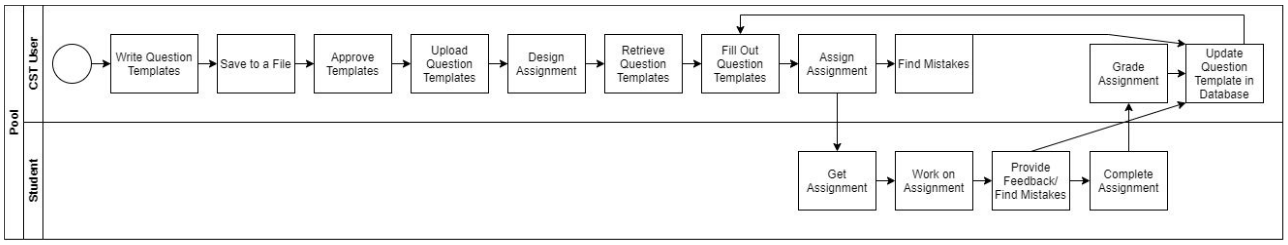


Figure 11. CST Typical Use Case Activity Diagram

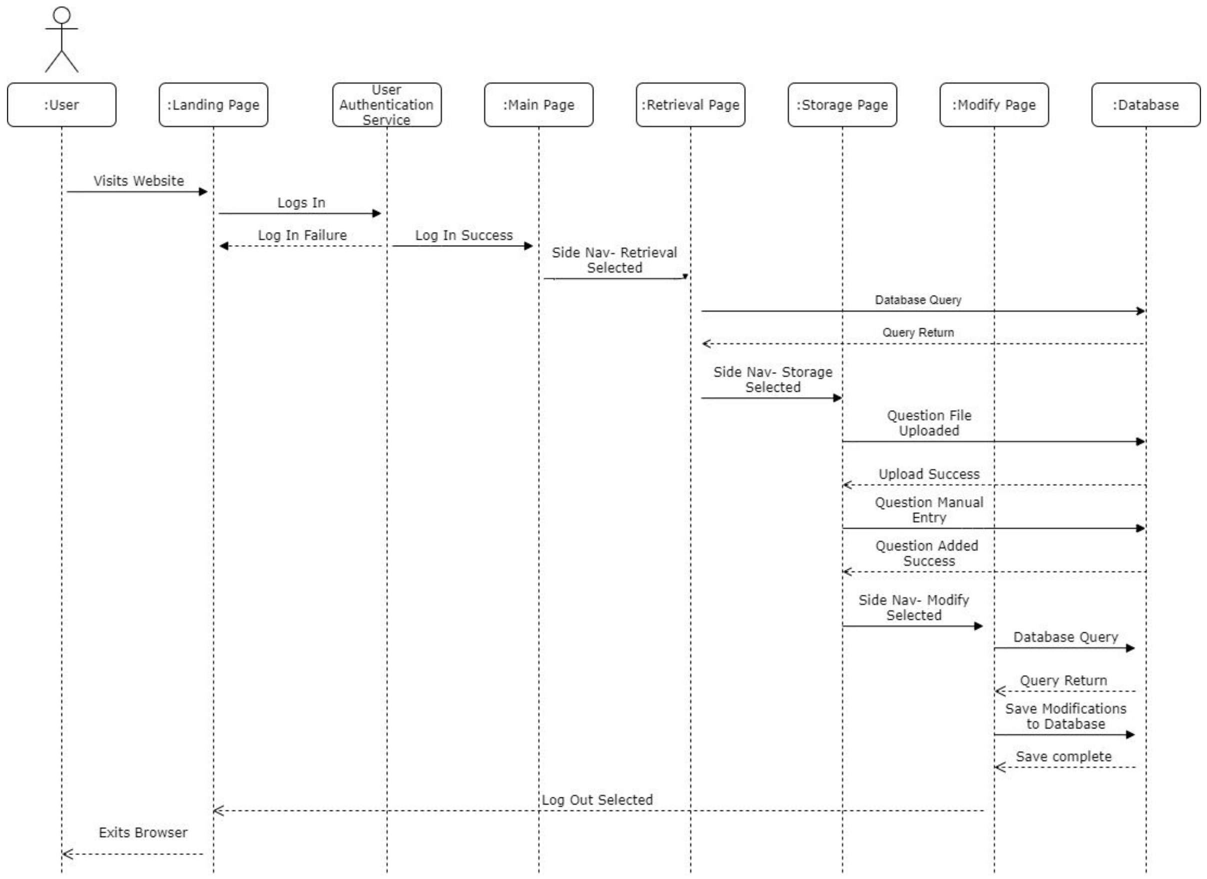


Figure 12. CST Typical Use Case Activity Diagram

The interface is split up into 3 major components:

- The client
- The server
- The database

4.3.2 The Client

In this component, the user talks to the client to obtain access to the website. According to C-SQID-TM, the user must be a University of Maryland, Baltimore County professor or an admin user; therefore, the user will be able to login to the site using their UMBC username and

password. The client then sends the credentials off and in return, it receives a token of access or denial of access. Figure 10 gives a good visual of this.
The client is also in charge of sending requests from the user, such as adding, retrieving and modifying the questions, to the server and also retrieving/displaying the responses that come back from the server.

4.3.3 The Server

The server's job is to handle requests from the client and give proper responses. In terms of this web application, the server will be asked to talk to the database and provide the necessary information back to the client, as far as, retrieving the questions to be used or modified and/or providing the database with a new question to be added.

4.3.4 The Database

C-SQID-TM hosts a database full of previously used or new questions given by professors at UMBC to be recycled and reinvented.

5. CSCI detailed design

5.1 CST Database

The main design requirements for the CST Database are explained above in section 4.1.2.

For the CST Database the programming language that will be used is MySQL. This software unit will contain, receive, and output data. This data that will be contained is shown in figure 5 above. The database will also output this data when requested by the user and will receive or add this data to the database when the user requests to add to the database.

5.2 CST Web Service

The main design requirements for the CST Database are explained above in section 4.1.1.

For the CST Web Service the programming language that will be used in Python and HTML using a Flask API. The main constraints that will be shown more in other software units but do affect the database has to do with the authentication. Only authenticated users should have access to the database of information. This software unit will allow authenticated users to have the ability to upload, download and modify information to the database as well as allow authenticated users to explore the application. The authenticated user will be able to use the web service to retrieve templated questions as well as upload if needed.

This software will be tasked with handling any error that it receives as well as avoiding from allowing users to upload their own code into the application.it shall also expose any unused memory in a secure method and to avoid memory leakage. It will also catch any errors including compromise confidentiality, integrity, or availability and return the appropriate errors.

5.3 CST Retrieval Page

The main design requirements for the CST Retrieval Page are explained above in section 4.1.6.

For the CST Retrieval page this will be programmed in Python and HTML using a Flask API for web development. The main constraint involved with this unit has to do with authentication of the user. For a user to be able to use this page to retrieve data from the database they must be authenticated properly so the software does not have any unwanted users trying to access the database.

This unit will allow users to receive a copy of the templated question from the database. This copy of the templated question will allow the authenticated user to download and modify it. These modifications shall not affect the actual saved templated question in the database but only the copy on the users side. Error handling and logic, the software unit will continue to accept new requests even if there is an internal error. Memory wise the shall give the user a copy of the data from the database but will not change or affect the stored information of memory in the database.

5.4 CST Storage Page

The main design requirements for the CST Storage Page are explained above in section 4.1.7.

For the CST Storage page this will be programmed in Python and HTML using a Flask API for web development. The main constraint involved with this unit has to do with authentication for the user. For a user to be able to use this page to upload templated questions to the database they must be authenticated properly so that the software does not have any unwanted users trying to access the database.

This unit will allow users to upload templated questions to the database. This will be done either by a file or a copy and paste into a text box.

5.5 CST Modify Page

The main design requirements for the CST Modify Page are explained above in section 4.1.8.

For the CST Modify page this will follow similar to the previous web service and be programmed in Python and HTML using a Flask API for web development. The main constraint involved with this unit has to do with authentication for the user. For a user to be able to use this page to upload templated questions to the database they must be authenticated properly so that the software does not have any unwanted users trying to access the database.

This section will allow users to modify already existing templated questions to the database.

5.6 CST User Authentication Service

The main design requirements for the CST User Authentication Service are explained above in section 4.1.4.

For the CST User Authentication Service this will use Google Sign-In to manage the users that should be allowed to use the web application. This unit will work with the web service as it needs to allow users to login in from the web application. The main constraint involved with this section would be limiting the access to only Professors, TA's and admins. This would require some manual settings on the admins side of who has what permissions.

This unit would be to figure out who is allowed to enter the web application. If a user is not allowed onto the web application as they do not have the permissions to access then that user will not be given access to the web application. If a user does have the permission to access the application must keep track of what permissions they have once on the application. All the other software units will keep track of the users permissions.

5.7 CST Landing Page

The main design requirements for the CST Landing Page are explained above in section 4.1.3.

For the CST Landing page this will follow similar to the previous web service and be programmed in Python and HTML using a Flask API for web development. The main constraint involved with this unit has to do with authentication for the user. The landing page will work with the user authentication service software unit to make sure the user has the correct authentications to enter the rest of the web application.

5.8 CST Main Page

The main design requirements for the CST Main Page are explained above in section 4.1.5.

For the CST Main Page this will be programmed in Python and HTML using a Flask API for web development. The main constraint involved with this unit has to do with authentication of the user. For a user to be able to use this page to access the other pages they must be authenticated properly so the software does not have any unwanted users trying to access the database and web application.

6. Requirements Traceability

Requirement	Description	Software Component
CST_F_1	The application shall store question templates in a database for later access and use.	CST Database
CST_F_2	The application shall allow the user to query the database to retrieve question templates.	CST Web Service
CST_F_3	The application shall categorize and sub-categorize question templates to allow the user to query the database for a specific type of question.	CST Database
CST_F_4	The application shall provide a way to store question difficulty for each question template.	CST Database
CST_F_5	The application shall allow the user to upload a file of question templates to add to the database.	CST Storage Page
CST_F_6	The application shall allow the user to save question templates to their devices.	CST Retrieval Page
CST_F_7	The application shall allow the user to fill out question templates with data.	CST Retrieval Page
CST_F_8	The application shall provide a way to manually enter a new question template in a text box and add it to the database.	CST Storage Page
CST_F_9	The application shall allow the user to modify question templates already in the database.	CST Modify Page
CST_F_10	The application shall recommend restrictions for assignments based on the question template type choses.	CST Database

CST_P_1	Application shall be able to perform main tasks such template retrieval and database management in an efficient manner.	CST Web Service
CST_P_2	Application shall access the correct user's choice of template and let the user make appropriate changes without changing the template document.	CST Retrieval Page
CST_C_1	The web application shall connect to the database when first loaded in the user's web browser.	CST Web Service
CST_S_1	The application shall possess user authentication capabilities to maintain the confidentiality of the question templates written to the interface.	CST User Authentication Service
CST_S_2	The application shall transmit user credentials between the client and the server in a manner that preserves confidentiality thereof.	CST User Authentication Service
CST_S_3	Availability of the application shall be reserved to professors and TAs employed at the receiving university.	CST User Authentication Service
CST_S_4	The application shall employ protection of text input fields against injection attacks, including, but not limited to, SQL injection, command injection, and cross-site scripting.	CST Web Service
CST_S_5	The application shall possess protection of file upload fields against remote code execution.	CST Web Service
CST_E_1	The application shall operate continuously, notwithstanding attempted denials of service.	CST Web Service
CST_E_2	The application shall continue to accept requests for question template retrieval in the event of an internal error.	CST Retrieval Page

CST_E_3	The application shall continue to allow question template uploads in the event of an internal error.	CST Storage Page
CST_E_4	The application shall dispose of unused memory in a secure manner that prevents leakage.	CST Web Service
CST_E_5	The application shall raise exceptions in the event of errors that compromise confidentiality, integrity, or availability.	CST Web Service