ASSIGNMENT 1 – TEST PLAN

Overall Test Plan

Our testing strategy encompasses both individual components and complete user workflows, ensuring comprehensive coverage of the system's functionality. The tests are categorized into four key areas: database, API, tools, and CBT user flow.

- **Database Tests**: These validate the proper execution of all database interactions, including user creation, authentication, and message storage.
- **API Tests**: These ensure that all interactions with the OpenAI API return appropriate responses, successfully reaching the user.
- **Tool Tests**: These verify that user inputs triggering tool (function) calls produce correct responses. Additionally, some tests will confirm that expected modifications to backend data occur as intended.
- **CBT User Flow Tests**: These white-box tests assess the system's UI by guiding a user through a complete CBT session. The focus is on detecting UI inconsistencies, ensuring relevant and safe responses, and mitigating any hallucinations from the OpenAI API.

The database, API, and tool tests may involve multiple functions and are primarily implemented as white-box tests, though black-box testing through standard QA procedures may also be employed. Some tests in the API and tool category may be blackbox tests but will require developer validation of the response since these tests incorporate the OpenAI API and the expected responses may not be definitive. Each test is designed to run independently while collectively ensuring the stability and reliability of both the frontend and backend components. As our system I continuously built, more tests will surely be documented to cover additional tools, user flows, and database interactions.

Test Case Descriptions

DB1.1	**Database Test 1**
DB1.2	This test will ensure messages sent by the user will be saved to the database.
DB1.3	This test will run the same code that sends the users message from the UI to the backend server and records the message in the database. A query will be run to fetch all messages from the conversation database and the first entry's content will be tested against the users sent message content.
DB1.4	Inputs: Logged-in user token, test message
DB1.5	Outputs: Single database entry with the test message in the 'content' column
DB1.6	Normal
DB1.7	Whitebox
DB1.8	Functional
DB1.9	Integration

DB2.1	**Database Test 2**
DB2.2	This test will ensure new users' information will be saved to the database.
DB2.3	This test will run the same code that creates a new user in the database. A query will be run to attempt to fetch a database entry that contains the user's information.
DB2.4	Inputs: test name, test email, test password
DB2.5	Outputs: Single database entry with the test username, email, and/or password

DB2.6 Normal
DB2.7 Whitebox
DB2.8 Functional
DB2.9 Integration

DB3.1 Database Test 3
DB3.2 This test will ensure existing users can sign-in.
DB3.3 This test will run the code the allows a user to sign in and then attempt to find a token to prove sig. in was successful.
DB3.4 Inputs: test username, test password
DB3.5 Outputs: token to prove user has signed in successfully
DB3.6 Normal
DB3.7 Whitebox
DB3.8 Functional
DB3.9 Integration

DB4.1 Database Test 4
DB4.2 This test will ensure existing users can sign-out.
DB4.3 This test will run the code the allows a user to sign out and then attempt to find a token (which will not exist) to prove sign out was successful.
DB4.4 Inputs: signed in user
DB4.5 Outputs: Check that the token does not exist now that user is signed out
DB4.6 Normal
DB4.7 Whitebox
DB4.8 Functional
DB4.9 Integration

DB5.1 Database Test 5
DB5.2 This test will ensure that user authentication (2FA) is effective and correctly implemented.
DB5.3 This test will run the code the allows a user to sign in and check that the user authentication (sms, email, etc.) is set to the user.
DB5.4 Inputs: test username, test password
DB5.5 Outputs: email/sms sent to user for 2FA
DB5.6 Normal
DB5.7 Whitebox
DB5.8 Functional
DB5.9 Integration

DB6.1 Database Test 6
DB6.2 This test will ensure that a user can retrieve full previous conversations from the database.
DB6.3 This test will run the code the allows a user to view old conversation history and ensure the full conversation is retrieved.
DB6.4 Inputs: N/A
DB6.5 Outputs: check if conversation is retrieved
DB6.6 Normal
DB6.7 Blackbox

DB6.8	Functional
DB6.9	Integration

API1.1	**API Test 1**
API1.2	This test will ensure messages sent by the user will receive a response from the API.
API1.3	This test will run the same code that sends the users message from the UI to the backend server and check for a successful API response.
API1.4	Inputs: test message
API1.5	Outputs: API response with code 200, API response content displayed to user
API1.6	Normal
API1.7	Both
API1.8	Functional
API1.9	Unit

API2.1	**API Test 2**
API2.2	This test will ensure messages sent by the user that do not receive a response from the API are handled gracefully.
API2.3	This test will run the same code that sends the users message from the UI to the backend server but will receive a mocked failed response and check for a graceful error message to the user.
API2.4	Inputs: test message
API2.5	Outputs: API response with error code, error message displayed to user
API2.6	Normal
API2.7	Both
API2.8	Functional
API2.9	Unit

API3.1	**API Test 3**
API3.2	This test will test the performance of the API response when given the longest allowed message (number of tokens).
API3.3	This test will run the same code that sends the longest possible message from the UI to the backend server and test the time it takes to receive a response.
API3.4	Inputs: long test message
API3.5	Outputs: API response with response content, time to return response
API3.6	Boundary
API3.7	Whitebox
API3.8	Performance
API3.9	Integration

TOOL1.1	**Tool Test 1**
TOOL1.2	This test will trigger the self-harm tool.
TOOL1.3	This test will send a message indicating self-harm to the OpenAI API and expect a response with mental health and emergency resources.
TOOL1.4	Inputs: test message

TOOL1.5	Outputs: API message response with mental health and emergency resources
TOOL1.6	Normal
TOOL1.7	Whitebox
TOOL1.8	Functional
TOOL1.9	Unit
TOOL2.1	**Tool Test 2**
TOOL2.2	This test will trigger the tool that marks agenda items completed.
TOOL2.3	This test will send a message that indicates a response to an agenda item to the OpenAI API and expect the agenda item to be marked complete where it is stored.
TOOL2.4	Inputs: test message, current agenda item
TOOL2.5	Outputs: API response indicating the agenda item is complete, current agenda item in agenda dictionary being marked complete
TOOL2.6	Normal
TOOL2.7	Whitebox
TOOL2.8	Functional
TOOL2.9	Unit
TOOL3.1	**Tool Test 3**
TOOL3.2	This test will trigger the tool that marks a new agenda item as current.
TOOL3.3	This test will send a message that indicates a response to an agenda item to the OpenAI API and expect a new agenda item to be marked as current where it is stored.
TOOL3.4	Inputs: test message, current agenda item
TOOL3.5	Outputs: API response indicating the agenda item is complete, agenda item in agenda dictionary that is now marked current
TOOL3.6	Normal
TOOL3.7	Whitebox
TOOL3.8	Functional
TOOL3.9	Unit
CBT1.1	**CBT Test 1**
CBT1.2	This test will verify a full session of CBT can be completed error free.
CBT1.3	This test will have a user go through a full CBT session and identify if the chat flows as expected and all agenda items get completed effectively be the end of the session.
CBT1.4	Inputs: N/A
CBT1.5	Outputs: N/A
CBT1.6	Normal
CBT1.7	Blackbox
CBT1.8	Performance
CBT1.9	Integration

Test Case Matrix

Test Case ID	Normal/ Abnormal	Blackbox/ Whitebox	Functional/ Performance	Unit/ Integration
DB1	Normal	Whitebox	Functional	Integration
DB2	Normal	Whitebox	Functional	Integration
DB3	Normal	Whitebox	Functional	Integration
DB4	Normal	Whitebox	Functional	Integration
DB5	Normal	Whitebox	Functional	Integration
DB6	Normal	Blackbox	Functional	Integration
API1	Normal	Both	Functional	Unit
API2	Normal	Both	Functional	Unit
API3	Boundary	Whitebox	Performance	Integration
TOOL1	Normal	Whitebox	Functional	Unit
TOOL2	Normal	Whitebox	Functional	Unit
TOOL3	Normal	Whitebox	Functional	Unit
CBT1	Normal	Blackbox	Performance	Integration