Test Plan and Report

Product Name: LEMMA
Team Name: LEMMA
Revision Date: May 24, 2025

System Test Scenarios

US 1 - As a user, I want to create and edit markdown notes so that I can capture my thoughts with rich formatting.

Scenario 1: Create/edit markdown notes (Passed)

1. Start LEMMA app;
2. Click on Folder button in the buttons sidebar; Pop-up of File Explorer/Finder will appear
3. Select desired folder to open on;
4. Click on Add button in the buttons sidebar; New note will be created in the directory and editor tab on the new note will appear
5. To open an existing note, click on the existing note in the notes sidebar;
6. User should see the editor tab on the existing note appear
7. User can begin typing on the editor with the option above the editor to format styling of their text.
8. Click on another note's tab bar above the editor to switch the editor to that note.
9. When done editing, click on the "X" button in the tab bar corresponding to the note to close out the editor on the note.

US 2 - As a user, I want my notes to be automatically saved and stored in a local database so that I don't lose my work.

Scenario 2: Auto-save notes and store notes in database (Passed)

1. Click on an existing note in the notes sidebar to open an existing note;
2. Begin typing on the editor of an opened note;
3. Upon change of the note, application will write the latest change to the markdown file, ensuring the contents are preserved.
4. Additionally, application will insert/update the document containing the note's content into the vector database.

US 3 - As a user, I want to organize my notes with tags and categories so that I can easily find related content.

Scenario 1.3: Add tags (Passed)

1. Beginning typing in the editor of an opened note;
2. Press “#” to create a tag;
3. Upon typing a symbol after the “#”, a blue round box will appear around the text appended with the “#” to indicate that it’s a tag.
4. Continue typing symbols to edit the tag;
5. Press Space key to finish editing the tag;
6. To delete a tag, press Cmd-Backspace on macOS or Ctrl-Backspace on Windows;

US 4 - As a user, I want to visualize connections between my notes in a 3D knowledge graph so that I can see relationships between my ideas.

Scenario 4: View knowledge graph visualization of notes (Passed)

1. Click on an existing note in the notes sidebar to open an existing note or click on the button on the buttons sidebar to add a note;
2. User should see a knowledge graph tab to the right with a 3D display focused on the node corresponding to the currently opened note.
3. Upon opening/creating another note, knowledge graph camera will smoothly change its display to focus on the node of the opened note.
4. Left-click and drag on the display to change the camera angle of the knowledge graph;
5. Right-click and drag on the display to pan the camera of the knowledge graph;
6. Scroll up or down on the display to zoom in and out of the knowledge graph;
7. Left-click on a node to open the note corresponding to the note
8. Left-click and drag a node to move the node around in the knowledge graph.

US 5 - As a user, I want to manually create connections between notes so that I can build my knowledge graph.

Scenario 5: Create links between notes (Passed)

1. Beginning typing in the editor of an opened note;
2. Type “[[“ to suggest a link of another existing note;
3. User will see a pop-up of a list of file link suggestions.
4. Select a note in the pop-up to establish a link with another note.
5. User should see the text wrapped in a squared blue box that contains the name of the linked node.
6. User should also see an edge created between the node of the currently opened note, and the node of the linked note.

US 6 - As a user, I want to search across all my notes so that I can quickly find specific information.

Scenario 6: Full-text searching over all the notes in the folder (Passed)

1. Click on the Search button in the buttons sidebar;
2. User should see a search bar and an "X" button that replaces the notes sidebar.
3. Click on the search bar and begin typing a query;
4. Users should see search results below the search bar containing a list of partially matched text highlighted in blue. Search results are grouped by notes.
5. Click on a search result;
6. User should see the editor open on the note of the selected search result.
7. Click on the "X" button next to the search bar to close the search display and return to viewing the notes sidebar;

US 7 - As a user, I want the system to automatically suggest connections between notes based on content similarity and synthesize ideas across my notes so that I discover relationships I might have missed and generate new insights.

Scenario 7: Automatically synthesize and generate notes (Passed)

1. Start LEMMA app;
2. Click on Setting (or Gear) button in the buttons sidebar;
3. User should see a modal appear for configuring the settingss of the AI model.
4. Under the "Experimental Features," toggle on the "Enable Experimental AGI" settings; User should see "Enable Live Mode" settings appear right below.
5. Toggle on the "Enable Live Mode" settings; Click on "Save";
    a. If any notes have not been chunked yet (or synced to AGI), system will begin chunking, with a notification on the top right displaying "Syncing AGI"
    b. Once all notes have been chunked, user should see a notification displaying "AGI synced successfully"
6. Upon clicking "Save", user system should see a "Live AGI Consciousness" section on the bottom of the settings modal with an interface to start the AGI and view perception mode and total thoughts generated.
7. Within the "Live AGI Consciousness" interface, click on "Start" to begin generating new ideas and thoughts from existing notes;
8. Optionally, click on "View Thought History" to explore further information about the ideas generated;
    a. Users should see a list of generated thoughts with each section describing the perception mode employed, the notes selected for generation, and a preview of the generated content.

    b. Click on the “X” button to return to the settingss configuration within the modal.

9. To view update of knowledge graph during the generation,
    a. Click on the “Close” button to exit the modal;
    b. Click on the Grid button in the buttons sidebar; Users should see the list of generated notes by the inference model
    c. Select on one of the notes in the notes sidebar;
    d. Users should see the knowledge graph populating with nodes and edges as the notes are generating
10. To stop the generation,
    a. Click on the Settings (or Gear) button in the buttons sidebar to open the settings modal back;
    b. Under the “Live AGI Consciousness”, click on the “Pause” button;
    c. Generation of notes should be terminated

US 8 - As a user, I want to ask natural language questions about my notes so that I can retrieve information conversationally.

Scenario 8: Question-answering based on notes using RAG (Passed)

1. If any notes have not been chunked yet (or synced to AGI),
    a. Click on the Settings (or Gear) button in the buttons sidebar to open the settings modal;
    b. Under the “Experimental Features,” toggle on the “Enable Experimental AGI” settings; Click on “Save”;
    c. System will begin chunking, with a notification on the top right displaying “Syncing AGI”
    d. Once all notes have been chunked, user should see a notification displaying “AGI synced successfully”
2. Open the Dialogue button in the buttons sidebar;
3. User should see a Chat window pop up.
4. Type a question in the input field; Click on the Send button to the right;
    a. System will perform semantic search over the vector database, extracting all relevant chunks to be passed in as context to the inference model.
5. User should see a text message bubble of their question followed by a text message bubble of the response being generated token by token.
6. Click on the Trash Can button to clear the message history;
7. Click on the “X” button to exit the chat interface;

US 9 - As a user, I want a Chrome extension that can save the current webpage to my notes so that I can capture online content with less manual effort.

Scenario 9: Save current webpage to notes (Passed)

1. Open LEMMA app and open Google Chrome on a webpage;
2. Assuming LEMMA Extensions is already installed, click on the Puzzle button near the top right corner of Chrome; Select LEMMA Extensions from the list;
3. User should see the extension interface with a chat window.
4. On the top right corner of the extension, click on the Save Note button;
5. User will be prompted to enter a file name to save the web content on.
6. Select “Save” to save the content via the application; Select “Cancel” to abort;
7. After the inference model processes the web content into markdown, users should see the new file created on the notes sidebar of the app.

US 10 - As a user, I want to ask questions about the current webpage on the Chrome extension so that I can retrieve more digestible information.

Scenario 10: Question-answering based on current web content on Chrome Extension (Passed)

1. Open LEMMA app and open Google Chrome on a webpage;
2. Assuming LEMMA Extensions is already installed, click on the Puzzle button near the top right corner of Chrome; Select LEMMA Extensions from the list;
3. User should see the extension interface with a chat window.
4. Type a question in the input field; Click on the Send button to the right;
    a. Extension will send the question alongside the web content to the inference service in the app.
5. User should see a text message bubble of their question followed by a text message bubble of the response being generated token by token.

Unit Tests

Unit tests are designed using the Jest framework. They are executed in a separate environment apart from development and do not start the Electron application, as only the backend components utilized by the application are tested.

Setup

During setup, create fixtures and install test doubles to avoid dependency issues in the testing environment

- Install dummy object for the Electron dependency since the components under test (CUT) don't require the Electron app to start up
- The system's interface for accessing the settings/configuration data, which includes the user's model preferences, is a depended-on component that is only initialized when starting up the Electron application, so a test stub is installed for this DoC.
- Initialize a separate vector database (ChromaDB) for running unit test suites involving database operations.

Vector Database (`tests/spec/database.spec.ts`)

- Tests all CRUD operations on the vector database via the interface provided by `src/main/database.ts`
- Ensures the correctness of:
  - Inserting/updating/deleting one or more documents
  - Full-text and similarity searching
  - Querying based on a combination of different query parameters
- **All test methods passed**

Knowledge Graph backend (`tests/spec/knowledge-graph.spec.ts`)

- Tests all operations for manipulating the knowledge graph data structure
- Ensures the correctness of:
  - Creating a new node in the graph
  - Updating/deleting a node in a graph
  - Creating a new link between nodes
  - Updating/deleting a link between nodes
  - Syncing graph data structure based on the change in the state of the files, which includes identifying:
    - If files are created/deleted
    - If files contain new/updated/deleted links to other files

- **All test methods passed**

Question-answering system (`tests/spec/qa-system.spec.ts`)

- Ensures all files in the directory are successfully chunked and stored in the database
- Expected proper behavior of the retrieval-augmented generation (RAG) pipeline for question-answering on notes
    - A test spy is installed on the CUT, namely the inference service, to ensure semantic search is performed on the database interface
    - Test method retrieves the chunks returned by the database for simple RAG evaluation
- A simple evaluation of the responses is executed to ensure that each response is consistent with its corresponding context retrieved from the database
- **All test methods passed**

Teardown

During teardown, remove all fixture files, and shut down/delete testing vector database.