# : RESULTS AND DISCUSSION

**4.1 INTRODUCTION**

## The previous chapter described the design approach by means of UML diagrams for system components and interactions. This chapter presents the outcomes of the project implementation, and it discusses the design, testing, and evaluation of the Mental Health Chatbot System. In addition to a description of the hardware and software environments, this chapter explains the test strategy and details the test cases executed during development. The chapter also provides an interpretation of the results, a summary of evaluation feedback from users, issues encountered during deployment, and recommendations for future improvements.

## 4.2 HARDWARE SPECIFICATIONS

*Tables 4-1 and 4-2* list the minimum hardware requirements necessary for running the Mental Health Chabot System in both desktop and mobile environments.

*Table 4‑1: Development Environment Hardware Requirements*

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **Minimum** | **Recommended** | **Purpose** |
| Processor | Intel Core i3 | Intel Core i5/i7 | Running development environment |
| RAM | 4 GB | 8 GB or higher | Smooth multitasking |
| Storage | 20 GB free disk space | 50 GB free disk space | Stores project files, virtual environment, libraries, and logs |
| Network | Stable broadband internet connection | High-speed internet | Quick dependency installations, version control updates |

*Table 4‑2: Production Environment Hardware Requirements*

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **Minimum** | **Recommended** | **Purpose** |
| Processor | Intel Core i5 | Intel Core i7/i9 | Handles concurrent user requests |
| RAM | 8 GB | 8 GB or higher | Smooth performance |
| Storage | 20 GB free disk space | 50 GB or more | Accommodates static files, user data, logs, and Chabot content |
| Network | Reliable broadband | High-speed connection | Reduces latency, stable communication |

## 4.3 SOFTWARE SPECIFICATION

*Table 4-3* lists the minimum software requirements necessary for running the Mental Health Chabot System in both desktop and mobile environments.

*Table 4‑3: Development and Production Software Requirements*

|  |  |  |
| --- | --- | --- |
| **Software** | **Version/Requirement** | **Purpose** |
| Operating System | Linux, Windows 10/11 | Provides the environment on which Django and related services run |
| Python | 3.10 | Primary language for Django; logic and backend development |
| Django | 5.1.4 | A high-level Python web framework that encourages rapid development and clean, pragmatic design. |
| Database | PostgreSQL | Stores user data, conversation logs, and chatbot content |
| Web/Application Server | Django dev server (development), Gunicorn/uWSGI + Nginx | Serves the Django application; manages static files and handles production traffic efficiently |
| Frontend | HTML5, CSS3, JavaScript | Provides the user interface: chat window, styling, basic interactivity |
| Browser (Client) | Chrome v89+, Firefox v86+, Edge v44+, Safari 12.1+ | Allows end users to access the chatbot interface |
| Version Control | Git and GitHub | Manages source code revisions, collaboration, and deployment workflows |
| Virtual Environment | Python venv | Isolates project dependencies and prevents conflicts with other system-installed packages |
| SSL/TLS Certificates | Required for HTTPS in production | Secures data in transit, especially critical for mental health-related user data |
| Django REST Framework | For REST API | The backend logic is exposed to the frontend or client via RESTFUL APIs |

## 4.4 TEST PLAN

It includes the testing strategies and methodologies applied to test if the system works properly, and in the most efficient manner. This document summarizes the testing goals, types of testing carried out, and the tools used.

### 4.4.1 Quality Objectives

The objective of testing is to check that the chatbot system functions and performs as expected, is secure, and has a good user experience. Specific objectives include:

1. To make sure that the chatbot answers with correct responses based on user queries.
2. Verifying user authentication and authorization mechanisms.
3. Integrating the chatbot with external services like LLM APIs and databases
4. Ensuring that mental health data protection regulations have been evaluated with respect to data security and privacy.
5. Preventing the chatbot from going down in high user traffic.

### 4.4.2 Types of Testing

#### 4.4.2.1 Unit Testing

Unit tests check the individual parts of the system through these components:

* User authentication module.
* Message processing and response generation.
* Data storage and retrieval mechanisms.
* API calls to external services.

The chosen testing tool was UnitTest from the Python built-in framework.

#### 4.4.2.2 Integration Testing

The integration test confirms that different software components communicate properly with one another to achieve expected functionality. This involves:

* The REST APIs that connect frontend (HTML/CSS/JS) elements with backend Django components function properly
* The system allows users to move between different pages starting from login to reach the chat interface.
* The test checks that API communication functions properly between the chatbot and external service systems.
* The process includes examination of data consistency and evaluation of database transactions.

Approach:

* Bottom-up integration: Start with testing individual modules before combining them.
* Top-down integration: Simulate real-world interactions from user inputs to chatbot responses.

Testing tools: Postman (API Testing), Django Test Client

#### 4.4.2.3 System Testing

The chatbot is tested in a system testing environment that is similar to the production environment. It checks:

* User login to chatbot workflow from end to end.
* Performance under load (e.g., concurrent users chatting simultaneously).
* Error handling and system recovery in case of failures.
* All browsers supported: Chrome, Firefox, Edge, and Safari.

#### 4.4.2.4 User Acceptance Testing (UAT)

Real users are placed to test the chatbot, to ensure they get the expectation. Areas of focus:

Natural conversation flow, but very easy to use.

* Accuracy of responses.
* Privacy concerns, especially handling of sensitive data pertaining to sensitive user.
* Feedback collection for further improvements.

Participants: End users

### 4.4.3 Test Cases

*Table 4‑4: Registration/Signup Module Test Case*

This table outlines the test case for user registration.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | Registration | |  | | | |
| **Test Case ID** | | AA 001 | |
| **Test Case Description** | | To verify registration functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | | **Actual Results** | **Status** | **Comments** |
| 1. Enter Username | Bright M | | Registration successful!  An activation link sent through email. | | Registration successful!  Email notification | Pass | Works well |
| 2. Enter email | bm@gmail.com | |
| 3. Enter Password | a123456789T! | |
| 4. Agree Privacy Policy and Terms & Conditions, Click on sign up | | |
| 5. Open email and click activation link | | |  | |  |  |  |

*Table 4‑5: Login Module Test Case*

This table details the test cases executed for the login functionality.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | Login | |  | | | |
| **Test Case ID** | | AA 002 | |
| **Test Case Description** | | To verify login functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | | **Actual Results** | **Status** | **Comments** |
| 1. Enter email | bm.@gmail.com | | Login successful!  Redirected to chat page | | Login successful!  Redirected to chat page | Pass | Works well |
| 2. Enter Password | a123456789T! | |
| 3. Click on login |  | |

*Table 4‑6: Forgot Password Module Test Case*

This table describes the test cases for the password recovery process.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | Forgot Password | |  | | | |
| **Test Case ID** | | AA 003 | |
| **Test Case Description** | | To verify forgot password functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | | **Actual Results** | **Status** | **Comment** |
| 1. Click [Forgot Password?](http://127.0.0.1:8000/forgotpassword/)  On Login Page |  | | Password reset link sent.  Sends a password reset email with a secure link. | | Password reset link sent.  Sends a password reset email with a secure link. | Pass | Works well |
| 2. Enter Email on Forgot Password Page | bm@gmail.com | |
| 3. Click Send Reset Link |  | |

*Table 4‑7: Reset Password Module Test Case*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | Reset Password | |  | | | |
| **Test Case ID** | | AA 004 | |
| **Test Case Description** | | To verify reset password functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | | **Actual Results** | **Status** | **Comment** |
| 1. Click the reset link send  to your inbox |  | | Password reset successful! You can log in with new password. | | Password reset successful! You can log in with new password. | Pass | Works well |
| 2. Enter new password | q123456789T! | |
| 3. Click Reset Password |  | |

*Table 4‑8: Generate Response Module Test Case*

This table provides test cases for validating chatbot responses during mental health queries.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | Generate Response | |  | | | |
| **Test Case ID** | | AA 005 | |
| **Test Case Description** | | To verify chatbot generate response functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | | **Actual Results** | **Status** | **Comment** |
| 1. Login to the system | Valid login credentials | | Chatbot returns empathetic, supportive message with advice | | Chatbot returns empathetic, supportive message with advice | Pass |  |
| 2. Enter a message into the input box | “I’m feeling anxious” | |
| 3. Click on Send and observe the response |  | |

*Table 4‑9: Conversation Manager Module Test Case*

This table provides test cases for validating the saving of encrypted message exchanges (user-bot messages) and retrieval and decryption of messages

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | Conversation Manager | |  | | | |
| **Test Case ID** | | AA 006 | |
| **Test Case Description** | | To verify chatbot conversation manager functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | | **Actual Results** | **Status** | **Comment** |
| 1. Login to the system | Valid login credentials | | User and chatbot message exchanges are saved to the conversation in the database. The conversation is retrieved and when user clicks on the conversation on the sidebar, the conversation is loaded on the chat area | | User and chatbot message exchanges are saved to the conversation in the database. The conversation is retrieved and when user clicks on the conversation on the sidebar, the conversation is loaded on the chat area | Pass | In the database, the message exchanges are saved to a conversation in an encrypted format |
| 2. Chat with the chatbot | User messages | |
| 3. Reload the page and click the hamburger menu to toggle the left sidebar if not open |  | |
| 4. Observe and click on the conversation shown on the sidebar |  | |

*Table 4‑10: MFA Verification Module*

Table 4-10 provides a test case for MFA Verification Module to validate its functionality

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | MFA Verification | |  | | | |
| **Test Case ID** | | AA 007 | |
| **Test Case Description** | | To verify MFA verification functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | | **Actual Results** | **Status** | **Comments** |
| 1.User logs at first and enables MFA by clicking on user menu icon, click on Account, and choose Enable MFA, or enabled by admin |  | | MFA Verification Modal Appears after submitting email and password.  After entering an MFA code, it is successfully verified and user logs in | | MFA Verification Modal Appears after submitting email and password.  After entering an MFA code, it is successfully verified and user logs in | Pass | The MFA Verification process allows user to retry enhancing user experience |
| 2. Logout by click on user menu icon, click on logout, and Signout |
| 3.Navigate to login and follow the subsequent steps |
| 4. Enter Email | bm@gmail.com | |
| 5. Enter Password | a123456789T! | |
| 6. Check MFA verification code sent to your email |  | |
| 7. Enter Code sent through email | Valid MFA Code | |

*Table 4‑11 ADMINISTRATION DASHBOARD*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | ADMIN | |  | | | |
| **Test Case ID** | | AA 008 | |
| **Test Case Description** | | To verify admin dashboard functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | | **Actual Results** | **Status** | **Comments** |
| 1. Navigate to admin login page |  | | The dashboard shows:  6 Key Metrics, that is total number of: users, active users, admins, resources, conversations, and MFA adoption  4 Analytical charts: conversation trends, user engagement, user distribution, and resources by category.  2 data tables; latest users and System admin logs | | The dashboard shows:  6 Key Metrics, that is total number of: users, active users, admins, resources, conversations, and MFA adoption  4 Analytical charts: conversation trends, user engagement, user distribution, and resources by category.  2 data tables; latest users and System admin logs | Pass |  |
| 2. Enter Email | bm@gmail.com | |
| 3. Enter Password | a123456789T! | |
| 4. Click login, if MFA is enabled do step |  | |
| 4. Enter MFA verification Code sent through email | Valid MFA Code | |
| 5. Confirm the dashboard shows:   * 6 Key Metrics, that is total number of: users, active users, admins, resources, conversations, and MFA adoption * 4 Analytical charts: conversation trends, user engagement, user distribution, and resources by category. * 2 data tables; latest users and System admin logs | | |

*Table 4‑12 ADMIN USER MANAGEMENT PANEL*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | ADMIN USER MANAGEMENT | | |  | | |
| **Test Case ID** | | AA 009 | | |
| **Test Case Description** | | To verify admin user management functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | **Actual Results** | | **Status** | **Comments** |
| 1. Navigate to admin login page |  | | User management panel allows to:  Search for users using either username or email.  Filter by role or MFA status.  Enable/Disable MFA.  Activate or Deactivate user account.  Delete user account.  Create an admin user. | User management panel allows to:  Search for users using either username or email.  Filter by role or MFA status.  Enable/Disable MFA.  Activate or Deactivate user account.  Delete user account.  Create an admin user. | | Pass |  |
| 2. Enter Email | bm@gmail.com | |
| 3. Enter Password | a123456789T! | |
| 4. Click login, if MFA is enabled do step |  | |
| 4. Enter MFA verification Code sent through email | Valid MFA Code | |
| 5. Click on USERS on the navigation on the admin dashboard | | |
| 6. Confirm you can:  See a table with list of users.   * Search for users using either username or email. * Filter by role or MFA status. * Enable/Disable MFA. * Activate or Deactivate user account. * Delete user account. * Create an admin user. | | |

*Table 4‑13 ADMIN RESOURCE MANAGEMENT*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Name** | | ADMIN RESOURCE MANAGEMENT | | |  | | |
| **Test Case ID** | | AA 009 | | |
| **Test Case Description** | | To verify admin user management functionality | | | | | |
| **Prerequisites** | | 1. Stable internet connection  2. Browser (Chrome, Edge) | | | | | |
| **Environmental Information** | | 1. OS: Windows  2. Device: Laptop | | | | | |
|  | | | | | | | |
| **Test Steps** | **Test Input** | | **Expected Results** | **Actual Results** | | **Status** | **Comments** |
| 1. Navigate to admin login page |  | | Resource management panel allows to:  See a table with list of resources.  Filter by category  Edit a resource.  Delete a resource  See a form :  With 4 fields; Title, Category, URL, and Description.  Allows adding a new resource. | Resource management panel allows to:  See a table with list of resources.  Filter by category  Edit a resource.  Delete a resource  See a form :  With 4 fields; Title, Category, URL, and Description.  Allows adding a new resource. | | Pass |  |
| 2. Enter Email | bm@gmail.com | |
| 3. Enter Password | a123456789T! | |
| 4. Click login, if MFA is enabled do step |  | |
| 4. Enter MFA verification Code sent through email | Valid MFA Code | |
| 5. Click on RESOURCES on the navigation on the admin dashboard | | |
| 6. Confirm you can:  See a table with list of resources.   * Filter by category * Edit a resource. * Delete a resource   See a form :   * With 4 fields; Title, Category, URL, and Description. * Allows adding a new resource. | | |

## 4.5 IMPLEMENTATION PLAN

The deployment strategy includes detailed description of the Mental Health Chatbot System's delivery through activities and assigned personnel and timeline. The developer team created this design to make the launch process between development phases and operational deployment transparent and seamless.

### 4.5.1 System Overview

The Mental Health Chatbot System functions as a web-based application accessible through mobile devices to help students manage their mental health by interactive chat. The system delivers appropriate mental health information by using AI algorithms which also provide step-by-step self-care guidance. Through natural language processing (NLP) technology the chatbot boosts both user interaction quality and overall experience.

### 4.5.2 System Description

The system is a web based mental health chatbot application, which offers discreet, confidential, conversational support to people affected by stress, anxiety, depression or other emotional challenges. It is a modular, scalable architecture that is developed using Django, Django REST Framework (DRF) for the backend, and HTML, CSS, and JavaScript for the frontend.

There are six core modules:

1. User Management: This module manages every part of accounts like secure sign up, activation, deletion and providing multi factor authentication (MFA) for more security.
2. Authentication Service: This module handles secure logins and ensures strong session protection, and supports Multi Factor Authentication.
3. Response Generator: The module uses natural language processing to generate thoughtful and empathetic replies to students’ mental health needs.
4. Resource Repository: Here is a rich library of helpful tools, articles, and self care exercises, and referral to professionals. The chatbot uses this collection to recommend things to students.
5. Conversation Manager: This module is in charge of maintaining the natural dialogue between users and the chatbot, remembering context, managing back and forth, and saving conversations for continuity.
6. Administrator: Built for the people managing the platform, this module gives insights through analytics, tools to manage users and roles, and makes it easy to update resources.

There are two user roles:

1. Students: These are the people who will use the system. Students can interact with the chatbot for mental health support and get contextually appropriate guidance, as well as access self-help resources.
2. Administrators: The staff members are the administrators. Users in charge of system level operations such as managing platform settings, managing user accounts, and maintaining response logic and resource base of the chatbot.

*Table 4‑14 Implementation Plan*

|  |  |  |  |
| --- | --- | --- | --- |
| **Strategy** | **Activity** | **Persons Involved** | **Duration** |
| Deployment | Uploading the system to the web server. | Developer | 1 day |
| Implementation | Conducting orientation and hands-on training for intended users | Staff or Administrators | 2 days |
| Monitoring | Continuously testing and observing system behaviour to identify and fix potential bugs or performance issues post-deployment. | Developer and Administrators | 2 days |

The implementation plan of the system is presented in Table 4-14. It describes what to do, who is involved, and how long it will take for each phase. The system is supposed to be deployed by the developer and then to be monitored to ensure that it's stable post deployment. In the implementation stage, a staff or administrator training session will be designed in a structured manner to make sure that they know how to operate the system.

## 4.6 EVALUATION

The researcher carried out an evaluation of the developed system with 71 respondents altogether. These respondents were a cross-section of the intended users of the system: 40 students, 1 system administrator, 10 staff members (mental health professionals or academic support staff), and 20 faculty members that are indirectly involved in user support and oversight.

Four (4) sets of survey questionnaires were prepared by the researcher to determine the effectiveness and usability of the system, which were designed for the different roles of the participants – administrator, students, staff members, and faculty members. The questionnaires were created to measure the current issues in the tradition of mental health support systems and the usability and functionality of the new platform.

The administrator was requested to assess the system in its overall acceptability in terms of system manageability, configuration options, and analytics features. At the same time, students, staff, and faculty members were also requested to appraise the system as to the ease of its use, accessibility and operative performance with an emphasis on the platform capacity to provide timely and pertinently mental health support.

Participants were also encouraged to leave comments and suggestions, which are beneficial in the future system improvement and iterations.