**CHAPTER 7**

**TESTING**

Software testing is an essential process in the development life cycle, helping ensure that the final product is reliable, secure, and performs as intended. It identifies bugs, verifies compliance with requirements, and improves overall software quality. Various types of testing are employed at different phases to cover the system thoroughly. The testing process includes unit testing, validation testing, functional testing, integration testing, and user acceptance testing. Each type has a distinct purpose and is performed by different stakeholders such as developers, testers, or end-users. This chapter discusses the testing strategies adopted during the project to ensure a robust and efficient system.

**7.1 METHODS OF TESTING**

The testing strategy for the system involved a combination of automated and manual testing methods, applied at different stages of development. Each method was carefully selected to ensure comprehensive coverage of the application’s behavior and performance. The key objective was to validate individual modules, verify correct integration, confirm functional correctness, and ultimately ensure the software met user expectations. Below are the key testing methods used:

**7.1.1 Unit Testing**

Unit testing is the process of testing individual components or functions of the software in isolation. The purpose is to ensure that each unit performs as expected. This form of testing is typically done by developers using testing frameworks such as PyTest, JUnit, or unittest. In this project, unit testing was applied to critical backend logic such as user input processing, data validation, and database operations. By catching bugs early in development, unit testing reduces the complexity of later-stage testing and simplifies debugging. Automated unit tests were written for functions like login authentication, data parsing, and error handling to ensure consistent and repeatable results. This contributed significantly to the stability of the application.

**7.1.2 Validation Testing**

Validation testing ensures that the software meets the user requirements and behaves as intended under real-world conditions. It answers the question: "Are we building the right product?" This is different from verification, which asks: "Are we building the product right?" In this project, validation testing was conducted by comparing the system’s outputs with expected results based on given inputs. For instance, legal sections suggested by the system were compared to official law documents for accuracy. Any deviation was carefully analyzed and fixed. Validation testing helped guarantee that the final product was not only technically correct but also usable and beneficial to its target audience.

**7.1.3 Functional Testing**

Functional testing evaluates whether the system's features perform according to specified requirements. It focuses on user interactions and the outcomes of different functions. Each feature—like login, language selection, crime input, and legal section suggestion—was tested with a variety of input combinations to ensure it behaved as expected. This testing was done using test cases that described the feature, expected results, actual results, and pass/fail status. Tools such as Selenium and Postman (for APIs) can be used for automated or manual functional testing. Ensuring that all functionalities work correctly helped increase the reliability and usability of the application.

**7.1.4 Integration Testing**

Integration testing verifies that different modules or services of the application work together as intended. Even if individual components pass unit tests, they may fail to work properly when combined. In this project, integration testing focused on ensuring smooth communication between the front-end (HTML/CSS), back-end (Python), the Rasa NLP engine, and the MySQL database. Test cases simulated real user workflows—like entering a crime description, getting follow-up questions, and receiving legal outputs. Integration issues like data misinterpretation or delayed responses were identified and resolved. This phase was critical to achieving a seamless and consistent user experience across all layers of the system.

**7.1.5 User Acceptance Testing (UAT)**

User Acceptance Testing (UAT) is the final phase of testing, performed by actual users to determine if the software is ready for deployment. It evaluates whether the system meets user needs in a real-world scenario. For this project, UAT involved law enforcement professionals and legal interns using the chatbot to input real or simulated crime cases. Feedback was collected on system accuracy, clarity of responses, and ease of use. Minor adjustments were made based on this feedback, such as improving the clarity of legal explanations or refining the follow-up questions. UAT confirmed that the solution was practical, efficient, and valuable for its intended users.

**7.2 Test Cases**

**7.1.1 Unit Testing**

**Test Cases:**

**Table 7.1: Unit Testing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Module** | **Test Scenario** | **Expected Output** | **Status** |
| UT01 | Login Module | User enters correct admin credentials | Redirect to admin dashboard | Pass |
| UT02 | Login Module | User enters incorrect credentials | Show "Invalid credentials" error message | Pass |
| UT03 | Crime Input | User submits a well-formed crime description | System generates appropriate follow-up questions | Pass |
| UT04 | Law Suggestion | NLP engine processes description and identifies legal section | Correct IPC section and crime category returned | Pass |

**7.1.2 Validation Testing**

**Test Cases:**

**Table 7.2: Validation Testing**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Expected Output** | **Status** |
| VT01 | User enters a valid crime description | System accurately interprets the text and identifies intent | Pass |
| VT02 | System analyzes the entered crime details | Correctly extracts relevant entities (person, place, act) | Pass |
| VT03 | Based on extracted data, suggest legal sections | Returns accurate IPC section and crime category | Pass |
| VT04 | User enters crime in a supported second language (e.g., Hindi/Portuguese) | System detects language and responds accordingly | Pass |

**7.1.3 Functional Testing**

**Test Cases:**

**Table 7.3: Functional Testing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Feature** | **Expected Output** | **Status** |
| FT01 | User logs in with valid credentials | Login Functionality | Redirect to dashboard/home page | Pass |
| FT02 | User selects a different language | Language Selection | Interface updates to the selected language | Pass |
| FT03 | User submits crime description | Crime Description Handling | System generates follow-up questions | Pass |
| FT04 | System processes follow-up answers and suggests IPC | Legal Section Assignment | Displays correct IPC section and crime category | Pass |

**7.1.4 Integration Testing**

**Test Cases:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Modules Tested** | **Test Scenario** | **Expected Output** | **Status** |
| IT01 | Frontend & Backend | User logs in through the UI, data passed to backend | User successfully authenticated and redirected | Pass |
| IT02 | Input Module & NLP Engine | User enters crime description, processed by Rasa NLP | Extracted entities and intent passed to next module | Pass |
| IT03 | NLP Engine & Legal Section Mapper | NLP identifies crime details, system suggests legal sections | Correct IPC section based on extracted intent | Pass |
| IT04 | Backend & MySQL Database | User information submitted and stored in database | Data successfully inserted into MySQL tables | Pass |

**Table 7.4: Integration Testing**

**7.1.5 User Acceptance Testing (UAT)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Expected Output** | **Status** |
| UAT01 | Law enforcement user inputs a real-world crime case description | System asks relevant follow-up questions and suggests appropriate IPC sections | Pass |
| UAT02 | Legal intern tests system with edge-case crime scenarios | System handles ambiguities gracefully and returns meaningful legal advice | Pass |
| UAT03 | User tests the chatbot in a supported regional language (e.g., Hindi or Marathi) | Chatbot correctly interprets and responds in the selected language | Pass |
| UAT04 | User provides feedback on clarity of legal explanations | System incorporates feedback; improved explanations deployed in next iteration | Pass |

**Test Cases:**

**Table 7.5: User Acceptance Testing (UAT)**