1.1 Use Case Specification

1.1.1. Use Case 1

Use case ID	**UC01**
Use Case	Sign up
Brief Description	The user creates a new account to use the system. This process involves selecting a user tier and making a payment.
Actor	User (Individual, Business), Payment Service
Pre-Condition	The user is not logged in and is on the homepage.
Result	The account is created, and the service tier is activated.
Main Scenario	1. The user selects the "Sign up" function. 2. The user enters personal information (Email, password, name...). 3. The system requests the user to choose a service tier. 4. The user selects a tier. 5. The system proceeds to the payment step. 6. The user selects a payment method (Banking or E-wallet). 7. The system sends a request to the Payment Service. 8. The Payment Service confirms the payment is successful. 9. The system creates the account and notifies success.

Alternative Scenarios	*At step 8: If the payment fails:* *8a. The system notifies a transaction error.* *8b. The system requests the user to choose another method or try again.*
Non-Functional Constraints	- Passwords must be encrypted before storage. - Payment transactions must comply with security standards (e.g., secure API connection).

1.1.2. Use Case 2

Use case ID	**UC02**
Use Case	Sign in
Brief Description	The user accesses the system to use Bot management and Helpdesk functions.
Actor	User (Individual, Business)
Pre-Condition	The user is not logged in (Guest).
Result	The user receives an authentication token and accesses the system.
Main Scenario	1. The user clicks the "Sign in" button. 2. The system displays the login form. 3. The user chooses a login method (Enter Email/Password OR select Google SSO). 4. The system authenticates the user information. 5. If the information is correct, the system redirects the user to the Dashboard page.

Alternative Scenarios	*At step 4: If the username or password is incorrect:* 4a. The system displays an "Incorrect login information" notification. 4b. The system requests the user to re-enter the information.
Non-Functional Constraints	- Authentication tokens (e.g., JWT) must be managed securely. - The client must never connect directly to cloud provider services using raw credentials.

1.1.3. Use Case 3

Use case ID	**UC03**
Use Case	Sign out
Brief Description	The user ends the current session to secure their account.
Actor	User (Individual, Business)
Pre-Condition	The user is currently logged in.
Result	The session is terminated, and the user returns to the Guest state.

Main Scenario	1. The user clicks on the Avatar or account menu. 2. The user selects "Sign out". 3. The system requests confirmation (displays a confirmation popup). 4. The user confirms. 5. The system deletes the login session information (Session/Token) on the browser. 6. The system redirects to the Landing Page or Login Page.
Alternative Scenarios	At step 4: if the user declines. 4.a: User returns to bot management page.
Non-Functional Constraints	- Session invalidation must happen immediately on the client side. - Navigation to protected pages after logout must be blocked.

1.1.4. Use Case 4

Use case ID	**UC04**
Use Case	View & Change Subscription
Brief Description	The user upgrades or downgrades their current service package.
Actor	User (Individual, Business), Payment Service
Pre-Condition	The user is logged in and is viewing subscription information (View subscription).
Result	The account is updated with the new service package.

Main Scenario	1. From the subscription view interface, the user selects "Change Subscription". 2. The user chooses to change the user tier (Change user tier). 3. The user selects a new package. 4. The system calculates the cost difference and requests payment. 5. The user performs the payment via Banking or E-wallet. 6. The system updates the new service package for the account.
Alternative Scenarios	*At step 5: If the user cancels the payment:* 5a. The system retains the current package. 5b. Return to the subscription information screen.
Non-Functional Constraints	- Pricing calculations (e.g., 200,000 VND for Individuals, 500,000 VND and 100 VND/token for Business) must be accurate.

1.1.5. Use Case 5

Use case ID	**UC05**
Use Case	Create Bot
Brief Description	The process of creating a new chatbot, including loading knowledge data, setting instructions, and testing before completion.
Actor	User (Individual, Business)
Pre-Condition	The user is logged in.

Result	The Bot is successfully created, data is vectorized, and the bot is ready for operation.
Main Scenario	1. The user selects the "Create bot" function on the Dashboard. 2. The system displays the Bot creation interface. 3. The user enters basic information (Bot Name, Description). 4. **Add Data to Knowledge Base:** The user selects a data source, performs **Upload file** (supports: .pdf, .docx, .txt, images .jpg/.png...) or enters text directly. 5. The system processes the file and vectorizes the data. 6. **Setting Instruction Bot:** The user enters instructions (System Prompt) into the editor or selects **Upload file instruction** (if a behavioral rule description file is available). 7. **Test Chat Bot:** Right on the Create Bot page, the system displays a **Preview** (Chat Demo) window. The user sends a test message to the Bot; the Bot responds based on the newly entered data and instructions. 8. The user clicks "Save/Finish" to complete. 9. The system saves the Bot configuration and notifies success.

Alternative Scenarios	At step 4: If the file format is not supported: 4a. The system displays a file format error notification. 4b. The system requests the user to upload a valid file. At step 5: If the data vectorization process fails: **5a. The system reports a data processing error.** **5b. The user checks the content and tries again.**
Non-Functional Constraints	- Strict data privacy: Third-party AI APIs must not use internal data for training. - Supported file formats must include PDF, DOCX. - File processing and embedding creation must handle "Cold Starts" from serverless functions gracefully.

1.1.6. Use Case 6

Use case ID	**UC06**
Use Case	Create Chat Widget
Brief Description	Generate embed code or a chat interface to integrate into a website.
Actor	User (Individual, Business)
Pre-Condition	At least one Bot has been created.
Result	The user receives the script code to embed into their website.

Main Scenario	1. The user selects the Bot needing a Widget. 2. The user selects the "Create chat widget" function. 3. The system generates the embed code. 4. The user copies the code for use.
Alternative Scenarios	
Non-Functional Constraints	- The generated script must be optimized to not affect the loading speed of the target website. - The widget must be responsive on both desktop and mobile devices.

1.1.7. Use Case 7

Use case ID	**UC07**
Use Case	View Helpdesk
Brief Description	Access the customer support dashboard to view tickets and reports.
Actor	User
Pre-Condition	Logged in.
Result	The user sees an overview of the customer support status.
Main Scenario	1. The user selects the "Helpdesk" menu. 2. The system displays the list of support tickets (View ticket). 3. The system displays report charts and statistics (View reports + statistics). 4. In statistics view the user can export the

	chart to file.
Alternative Scenarios	At step 4: If the system failed to export to file. 4.a: An error message pops up to indicate failure to export.
Non-Functional Constraints	- Reports must accurately reflect interaction metrics and ticket resolution times. - Dashboard data loading should be efficient.

1.1.8. Use Case 8

Use case ID	**UC08**
Use Case	Handle Tickets
Brief Description	Staff/User responds to and resolves requests from customers.
Actor	User
Pre-Condition	The user is viewing the ticket list (View ticket).
Result	The ticket status is updated, and the response is sent.

Main Scenario	1. From the ticket view interface, the user selects a specific ticket to handle. 2. The system displays detailed ticket content. 3. The user enters a response or changes the ticket status (Open -> Resolved). 4. The system saves the update.
Alternative Scenarios	At step 3: if changes can't be made to the ticket status at the moment due to some error: 3.a: an alert shows up. 3.b: user retries to change the ticket status.
Non-Functional Constraints	- Updates to ticket status must be recorded in the NoSQL database immediately. - The system must manage customer and employee data securely,

1.1.9. Use Case 9

Use case ID	**UC09**
Use Case	Create Group
Brief Description	Create a working group for multiple people to manage bots together.
Actor	Business (Business account only)
Pre-Condition	The account type is Business.
Result	A new group is created with the invited members.

Main Scenario	1. The Business User selects the group management function. 2. The user selects "Create group". 3. The user enters the group name and adds members (enter email). 4. The system sends invitations and creates the group.
Alternative Scenarios	*At step 3: If the email does not exist in the system:* 3a. The system notifies that the email is not registered. 3b. The system sends an email inviting them to join the platform (optional).
Non-Functional Constraints	- Access control must ensure only Business accounts can create groups

1.1.10. Use Case 10

Use case ID	**UC10**
Use Case	Share Bot
Brief Description	Share Bot access rights with a group or another user.
Actor	Business
Pre-Condition	A Bot exists, and the account is Business.
Result	The shared user has access rights to the Bot.

Main Scenario	1. The Business User selects the Bot to share. 2. The user selects the "Share bot" function. 3. The user selects a Group. 4. The user sets permissions (View/Edit). 5. The system grants access rights.
Alternative Scenarios	
Non-Functional Constraints	- Sharing settings must be updated in real-time for collaboration. - Security constraints regarding data access must be maintained for shared users.

1.1.11. Use Case 11

Use case ID	**UC11**
Use Case	Save Data to Database
Brief Description	System-level use case to ensure data persistence.
Actor	System
Pre-Condition	A data change event occurs (create bot, chat, edit config).
Result	Data is securely stored in the database.

Main Scenario	1. This use case is triggered automatically. 2. When there is a new conversation -> Trigger **Save conversations**. 3. When a user creates/edits a bot -> Trigger **Save bot's configuration**. 4. The system performs data writing to the Database. 5. The system returns a success status.
Alternative Scenarios	At step 5: If the system fails to write data to database. 5.a: The system returns failure status. 5.b: The system retry from step 4 until success.
Non-Functional Constraints	- Data is stored in a NoSQL database (Firebase) . - Security by Layer: Sensitive data handling must be restricted to Server Components.