# Software Requirements Specification (SRS)

**for**

**AI-Powered Digital Adoption Plugin for Website [AdopTrix]**

[Version 1.0]

By:
Shreya Singh
Ast. SDE Intern

Supervisor:
Mr. Akhilesh Verma
Chief Executive Officer

**Akoode Technologies**

**6th June 2025**

## Table of Contents

## Revision History

| **Name** | **Date** | **Reason For Changes** |
|---|---|---|
| Shreya Singh | 6th June 2025 | First Copy |
| | | |

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to define the software requirements for the AI-Powered Digital Adoption Assistant called AdopTrix. It is a browser plugin designed to serve as a Digital Adoption Platform (DAP) tool. It will have an AI chatbot, UI element recognition feature, and guided walkthrough overlays to assist users in navigating complex web applications. The plugin will enhance user onboarding and support them by offering contextual, interactive, and adaptive guidance directly within the web interface.

It will provide step-by-step guidance by showing messages, tips, or pop-ups on the screen while the user is using the app. This will make it easier for people to understand what to do without following a video tutorial or a handbook manual by providing contextual help, interacting with elements on a page, and enhancing user experience and onboarding.

## 1.2 Document Conventions

This SRS follows standard conventions for software requirement specifications. Requirements are listed using hierarchical structure, with high level requirement providing content for more detailed requirements. Each requirement statement is assigned its own priority level to denote its importance. Additionally, key terms and concept may be highlighted or italicized for clarity and emphasis.

## 1.3 Intended Audience and Reading Suggestions

- **Developers**: To understand the system architecture, technical requirements, and implementation scope.
- **Project Managers**: For planning timelines, managing resources, and aligning deliverables with project goals.
- **Stakeholders**: To gain clarity on the platform's objectives, functionality, and strategic value.
- **Testers**: To design test cases and validate the system based on defined functional requirements.

## 1.4 Project Scope

The AI-Powered Digital Adoption Assistant, AdopTrix aims to simplify and enhance user interaction with complex web applications by offering context-aware support. It will be an AI-integrated Digital Adoption Platform (DAP) that overlays directly onto existing websites.

**AdopTrix** is an intelligent browser-based plugin designed to simplify digital adoption for users across complex web platforms. Its core purpose is to assist users in navigating unfamiliar digital interfaces by offering interactive, step-by-step walkthroughs created by platform administrators. By leveraging both simplified description and element detection, AdopTrix enhances onboarding, training, and real-time support for web users.

AdopTrix delivers the following key objectives:

- **Seamless User Guidance**: AdopTrix eliminates confusion for new users by offering clear, interactive navigation paths. These step-based visual cues help user's complete platform-specific tasks confidently, reducing the learning curve and increasing engagement.
- **Admin-Controlled Tour Creation**: Through the admin panel, platform managers can customize tours by interacting with the website while the plugin automatically captures each step. In capture mode the click actions are stored securely and organized sequentially, making it easy to deploy tailored help journeys.
- **Smart UI Detection**: AdopTrix uses UI element matching and AI-powered APIs for understanding queries to determine the user's current location within a website. It then dynamically highlights the next actionable element, ensuring accurate, context-aware guidance.
- **Task-Based Search and Execution**: Users can search for help with specific tasks (e.g., "Create a GitHub Repository"), and the assistant bot will walk them through the exact steps needed, one at a time, until completion.
- **Dual Guidance Modes**: The system supports two navigation engines—static tour playback using stored data, and by fetching the data from the web via chatbot API for more dynamic request—allowing flexibility for both static and dynamic websites.
- **Improved Productivity and Support Efficiency**: By automating user support and walkthrough creation, AdopTrix reduces dependency on human helpdesk teams. It empowers users to solve their issues independently and quickly.
- **MVP Focus on GitHub**: For its Minimum Viable Product (MVP), AdopTrix is being developed specifically for GitHub. This platform provides a controlled UI environment for demonstrating core functionality and testing features like task navigation, screen capture, and AI integration.

AdopTrix aligns with digital transformation goals by fostering platform adoption, enhancing user experience, and reducing onboarding time. It contributes to the broader vision of empowering users to confidently navigate digital tools, making web interfaces more accessible, intuitive, and efficient—no matter the complexity of the platform.

## 1.5 Minimum Viable Product

The initial Minimum Viable Product (MVP) will focus on Chrome browser compatibility, integration with open-source LLM-based APIs, and support for configurable documentation sources.

**The MVP (Minimum Viable Product)** will include:

- A chatbot UI embedded as a plugin on any website.
- Integration with LLM API for conversational processing.

- DOM interaction (scrolling, highlighting elements, prompting clicks).
- Dialog Box with Text Description for more clarity about the action.
- Ability to walk through web tasks with step-by-step guidance.

**Future versions** may include support for voice interactions, advanced analytics, and image recognition for locating UI elements. That can involve:

- Integration with application-specific documentation for domain training.
- Local LLM (Large Language Model Support).
- Mascot-guided interactive user navigation.

## 1.6 References

- WalkMe and Whatfix documentation
- LLM API documentation
- Chrome Extension Developer Guide
- References Karl E. Wiegers. (2002). "Software Requirements Specifications (SRS)." Retrieved from IEEE Software Requirements Specification Template.

# 2. Overall Description

## 2.1 Product Perspective

The Digital Adoption Platform is designed as a standalone chrome browser-compatible plugin that seamlessly integrates with existing websites. It functions as an interactive overlay, offering real-time, AI-driven assistance and contextual UI guidance without requiring any modifications to the underlying application.

Unlike conventional help documentation or external user manuals, this tool operates within the user's live workflow, delivering timely support through intelligent step-by-step walkthroughs, visual cues, and dynamic tooltips. It is purpose-built to enhance onboarding, navigation, and user engagement for both end-users and businesses — all while maintaining the native look and feel of the host platform.

## 2.2 Product Functions

### 2.2.1 End-User Assistance Module

This is the interface that end-users (i.e., general website visitors or platform users) interact with.

- AI Chatbot Overlay: Display a conversational AI assistant on the website interface to interpret user queries in natural language.
- Real-Time Query Handling: Provide immediate task-based help, such as “how to create a repo” or “find archived issues” by walking users through each step.
- Interactive Step-by-Step Guidance: Highlight relevant UI elements and offer instructions as users follow task-based paths.
- Smart Tooltip Support: Display contextual tooltips with brief instructions or hints near buttons and input fields to enhance the learning experience.
- Screen Matching & Navigation: Identify the user’s current screen using image recognition or AI element detection and trigger the correct next step accordingly.
- Progress Indicators: Show visual step counters and navigation buttons (Next/Back) to guide users through multi-step flows.

### 2.2.2 Admin Panel / Business Owner Services

This is the backend dashboard for platform administrators and business clients who install the plugin on their websites.

- Tour Creation Interface: Enable admins to manually create guided tours by navigating the platform while the system auto-records steps via screen capture and DOM tracking.
- Custom Workflow Builder: Allow the definition of specific task flows (e.g., onboarding tutorials, help tours) with defined start and end states.
- Styling and Branding Options: Let businesses customize the appearance of the assistant to align with their website’s branding (colours, icons, language, etc.).
- Role-Based Access: Differentiate between user types (Admin, User) for secure control over tour management and content updates.
- Storage access: Data provided by the admin in terms of UI element description and its function helps the plugin to guide the user accurately.

## 2.3 User Classes and Characteristics

The Digital Adoption Platform (DAP) is designed to support the following primary user groups:

**1. End Users (General Website Visitors)**

- Frequency of Use: End users interact with the platform occasionally, typically when they require assistance navigating or performing specific tasks on a website.
- Subset of Product Functions Used: These users leverage real-time AI assistance, guided tours, highlighted UI elements, tooltips, and navigation instructions to complete tasks more efficiently.
- Technical Expertise: Users may have varying levels of digital literacy. Hence, the interface must be highly intuitive, visually guided, and accessible to non-technical users.

- Security or Privilege Levels: Users have limited access to their current session and are only guided through predefined or AI-generated workflows.
- Educational Level/Experience: Designed to accommodate a broad range of users regardless of their educational background, with a focus on simplicity and clarity.

**2. Administrators (Website Owners / Managers)**

- Frequency of Use: Administrators use the platform regularly to manage and customize guided tours, chatbot behaviour, and user flows.
- Subset of Product Functions Used: They utilize the dashboard for creating walkthroughs, uploading documentation, training the chatbot, analysing usage data, and branding the interface.
- Technical Expertise: Admins may possess moderate technical knowledge. The system should support low-code/no-code tour creation tools to make adoption seamless.
- Security or Privilege Levels: Admins have elevated permissions, including content control, analytics access, and customization capabilities.
- Educational Level/Experience: Typically, tech-savvy or platform owners with enough experience to manage configuration, user assistance journeys, and system performance.

**3. Developers (System Integrators or Extension Developers)**

- Frequency of Use: Developers interact with the backend and APIs during integration, customization, and feature extension phases.
- Subset of Product Functions Used: Developers utilize APIs, plugin configuration settings, UI behaviour customizations, and access controls to align the plugin with platform-specific requirements.
- Technical Expertise: High technical proficiency is expected; the platform should provide adequate documentation and developer tools.
- Security or Privilege Levels: Developers may have full access to integration settings and backend services, requiring secure and controlled access mechanisms.
- Educational Level/Experience: Usually professionals with strong experience in web development, browser extension frameworks, and system design.

## 2.4 Operating Environment

The Digital Adoption Platform (DAP) is developed as a browser-based plugin and is designed to function seamlessly across various environments, ensuring compatibility and performance consistency:

a) **Supported Platforms: Web Browsers:** Compatible with modern browsers such as Google Chrome, Microsoft Edge, Mozilla Firefox, and Safari.
b) **Operating Systems:** Windows 10 and above, macOS and Linux.
c) **Hardware Requirements:** Devices capable of running supported browsers smoothly, including desktops and laptops with standard processing capabilities (minimum 4 GB RAM, dual-core processor).

d) **Software Components:** JavaScript runtime environment for plugin execution. Backend integration with cloud-hosted services for user authentication, tour data storage, and analytics. AI-based APIs for DOM analysis and response generation. Image processing or screen matching libraries (where applicable for advanced features)

The DAP is optimized for low latency and minimal resource consumption, ensuring non-intrusive performance across different systems while offering reliable real-time support.

## 2.5 Design and Implementation Constraints

Several constraints must be considered during the design and implementation of **AdopTrix:**

1. **Browser Compatibility:** AdopTrix will work on chrome browsers. This imposes constraints on the use of browser-specific APIs or experimental JavaScript features.
2. **DOM Accessibility:** AdopTrix relies on the ability to access and interpret the structure of web pages (DOM elements). Dynamic content, custom-rendered elements, or heavily obfuscated DOM structures may limit accurate interaction and highlight functionality.
3. **Security and Privacy:** The plugin must not collect or transmit sensitive user data unless explicitly permitted. AdopTrix must operate within browser sandboxing policies and comply with data protection regulations for safe usage across different regions.
4. **Performance Optimization:** The plugin must be lightweight and efficient, avoiding delays or memory issues on client machines. This necessitates careful code optimization, lazy loading of features, and minimal resource consumption.
5. **Cross-Platform Styling and Responsiveness:** Since AdopTrix overlays on various websites with different styles, it must use isolated CSS and resilient positioning to prevent conflicts with existing UI elements.
6. **AI and Image Recognition Constraints:** For functionality such as image-based navigation and screen matching, reliance on image recognition APIs introduces latency, accuracy variability, and dependency on third-party services.
7. **Admin-Level Access Limitations:** AdopTrix offers a visual tour creation feature for admins, but its ability to interact with certain page elements may be limited by security restrictions on the host website.
8. **Scalability and Analytics Handling:** As user volume grows, the backend infrastructure supporting storage, admin tours, and analytics must scale accordingly. This adds complexity to cloud storage, API throttling, and real-time update handling.
9. **Customizability Across Clients:** Businesses using AdopTrix will require branding and workflow customizations. Designing flexible configurations without sacrificing simplicity and performance is a key challenge.
10. **Limited Control over Third-Party Websites (for MVP):** For the MVP targeted at GitHub, AdopTrix will be constrained by GitHub's static and dynamic content structure. Tour creation and guidance must align with what can be reliably accessed or highlighted within GitHub's DOM.

## 2.6 User Documentation

For **AdopTrix**, the following user documentation components will be delivered along with the software:

1. **User Manual:** A detailed user manual explaining how to install, configure, and use AdopTrix's key features such as AI chatbot assistance, interactive walkthroughs, and tooltip guidance. The manual will include step-by-step instructions and practical examples to help users navigate and maximize the plugin's capabilities.
2. **FAQs (Frequently Asked Questions):** A curated list of common questions and answers related to installation, feature usage, compatibility, and troubleshooting. This section aims to provide users with immediate solutions and improve overall user satisfaction.
3. **Release Notes:** Documentation summarizing enhancements, new features, and bug fixes in each software release. Release notes will keep users informed about updates and improvements, helping them understand changes in functionality and performance.
4. **Glossary:** A glossary defining technical terms, acronyms, and concepts related to digital adoption platforms, browser plugins, and AI-based assistance used within AdopTrix. This will help users better understand the terminology and features throughout the documentation and software interface.

## 2.7 Assumptions and dependencies

1. **AI and UI Interaction Accuracy:** The effectiveness of AI chatbot and UI element recognition depends on the accuracy and reliability of third-party AI models and browser DOM analysis libraries. Limitations or inaccuracies in these technologies may affect the quality of real-time assistance and UI highlights.
2. **Integration with Target Websites:** It is assumed that websites where AdopTrix is deployed follow standard web technologies and provide stable DOM structures. Major changes or highly dynamic content could impact plugin performance. Dependencies exist on website compatibility and cooperation for seamless overlay.
3. **Browser Compatibility:** AdopTrix assumes support from chrome browser, any browser updates or limitations affecting extension APIs could impact plugin functionality.
4. **Regulatory and Privacy Compliance:** The development assumes full compliance with relevant data protection laws when handling user data. Proper encryption and privacy measures will be implemented to safeguard sensitive information.
5. **Availability of Skilled Development Team:** The project depends on the availability of developers with expertise in browser extensions, AI integration, and UI/UX design to deliver the required features within the planned timeline.
6. **Hosting and Backend Services:** AdopTrix relies on cloud infrastructure and third-party AI services for chatbot responses and analytics processing. The uptime and performance of these external services are critical for smooth operation.

7. **User Adoption and Engagement:** Successful deployment assumes that website owners and end-users will actively adopt and engage with AdopTrix, using its interactive guidance features as intended.
8. **Customization:** For effective chatbot performance, it is assumed that businesses will provide adequate documentation, FAQs, to customize the guided route. Lack of quality input data may limit the relevance of AI assistance.

# 3. External Interface Requirements

## 3.1 User Interfaces

AdopTrix is developed as a browser extension primarily targeting modern web browsers such as Google Chrome. The user interface is designed to seamlessly overlay on existing websites, providing intuitive, real-time AI assistance and UI guidance. The following characteristics define the user interface design for AdopTrix:

1. **Visual Design Standards:**

    - The UI is designed using Figma, following modern web and extension design principles for clarity and usability.
    - The design emphasizes minimal intrusion, using transparent overlays, tooltips, and highlight boxes that blend naturally with the underlying web content.
    - Consistent styling is maintained through CSS, ensuring the extension UI matches the look and feel of the host websites without overwhelming the user.

2. **Screen Layout and Interaction:**

    - The extension UI components such as chatbot overlays, tooltip boxes, and step-by-step walkthroughs are dynamically injected into the target website's DOM.
    - Layouts use flexible CSS positioning (e.g., fixed, absolute) to adapt to varying page structures and screen sizes.
    - Interactive elements respond fluidly to user actions, such as clicks and hovers, enabling smooth guidance and assistance.

3. **Resolution and Device Support:**

    - AdopTrix is optimized for desktop browsers, primarily supporting standard screen resolutions for laptops and desktops.
    - The UI components scale appropriately with zoom levels and different screen sizes to maintain legibility and usability.

4. **Navigation and Controls:**

    - Users interact with AdopTrix through intuitive controls including buttons, links, and chat input fields embedded in the overlay.

- Navigation within multi-step walkthroughs is facilitated via "Next," "Back," and "Close" controls, allowing users to progress or exit guidance easily.
- The AI chatbot supports text input and displays responses contextually within the overlay window.

5. **Error Handling and Feedback:**

   - Any errors in loading or processing user queries are communicated clearly through unobtrusive alert banners or popup messages within the extension UI.
   - Feedback messages provide actionable instructions or retry options to ensure users can resolve issues smoothly.

6. **Technology Stack Integration:**

   - The extension UI is implemented using JavaScript, HTML, and CSS, leveraging the Chrome Extensions API for injection and communication with background scripts.
   - Firebase is integrated to handle backend functions such as user authentication, real-time database for session and analytics tracking, and hosting cloud functions for AI interactions.
   - Figma serves as the design tool to prototype, iterate, and finalize UI/UX layouts before implementation.

## 3.2 System Features

### 3.2.1 Feature 1 – Interactive AI Chatbot

This feature provides users with a conversational AI assistant that understands natural language queries and offers context-based guidance to enhance user experience.

- **Natural Language Understanding:** The chatbot accepts free-form text queries from users in everyday language, interpreting intent and extracting relevant information.
- **Context Awareness:** It uses context from the current webpage or user session to provide precise and relevant responses.
- **Suggestion & Action Buttons:** Alongside text responses, the chatbot dynamically displays suggested actions and clickable buttons to allow users to quickly perform tasks or navigate options without typing.
- **Integration with Large Language Model (LLM) API:** The chatbot communicates with an external LLM API (e.g., OpenAI GPT models) to process user queries and generate intelligent, human-like responses.
- **Session Management:** Maintains conversation state to handle follow-up questions and multi-turn dialogues seamlessly.
- **Error Handling:** Provides clarifications or fallback options when queries cannot be understood or processed.

### 3.2.2 Feature 2 – Website Plugin

A browser extension that enhances user interaction with websites by detecting, highlighting, and overlaying instructions on UI elements.

- **Element Detection via CSS Selectors:** The plugin analyses the current webpage DOM and identifies relevant UI elements using CSS selectors, enabling precise targeting.
- **Dynamic Overlay Rendering:** Generates instructional overlays such as tooltips, highlights, or modal windows positioned relative to the detected elements to guide users visually.
- **Responsive to User Interaction:** Overlays adapt dynamically to user actions such as scrolling, resizing, or page navigation to maintain visibility and relevance.
- **Cross-Page Persistence:** Maintains state or progress of guidance when users navigate through multiple pages within the same website or application.
- **Performance Optimization:** Ensures minimal impact on page load times and browser responsiveness through efficient DOM manipulation and resource management.

### 3.2.3 Feature 3 – Guided UI Walkthroughs

Provides users with step-by-step walkthroughs to assist them in learning how to use specific software or web applications.

- **Intent-Based Triggering:** Walkthroughs are initiated automatically when the system detects relevant user queries or requests for help related to particular workflows.
- **Step Sequencing:** Breaks down complex tasks into sequential steps, guiding users progressively through each action required.
- **Visual Highlighting:** Each step visually highlights the relevant UI element(s) with emphasis effects like borders, shading, or arrows to draw user attention.
- **Scroll Control:** Automatically scrolls the webpage or application window to bring the target elements into view during each step.
- **User Control:** Allows users to pause, resume, skip, or exit the walkthrough at any time for better flexibility.
- **Progress Feedback:** Displays progress indicators (e.g., step numbers or completion percentage) to keep users informed of their status within the walkthrough.

### 3.2.4 Feature 4 – Description-Based Dialog Box

Displays concise dialog boxes near focused UI elements to provide contextual assistance and instructions.

- **Proximity Display:** Dialog boxes appear adjacent to the currently focused or highlighted element, ensuring guidance is contextually relevant and easy to associate.
- **Concise Messaging:** Presents brief, clear messages that inform users about the next action, feature description, or tips to enhance task completion.

- **Dynamic Content Updates:** Content of the dialog box updates dynamically based on the user's current step in a workflow or interaction, reflecting the latest relevant information.
- **Integration with Documentation and LLM Prompts:** Utilizes existing documentation and AI-generated responses to ensure the dialog content is accurate, helpful, and consistent.
- **User Controls:** Provides options for users to close, minimize, or reopen the dialog box as needed, avoiding obstruction of key UI components.
- **Accessibility Compliance:** Dialog boxes are designed to be accessible, supporting keyboard navigation and screen readers to accommodate all users.
- **Visual Design Consistency:** Follows design guidelines to maintain a clean, unobtrusive appearance that aligns with the overall user interface style.

## 3.3 Hardware Interfaces

AdopTrix operates as a browser extension and, as such, does not directly interface with any hardware components. The extension relies on the host device's standard hardware capabilities provided through the web browser environment.

Key hardware aspects utilized indirectly by AdopTrix include:

1. **Display:** The extension's user interface components (chatbot overlays, tooltips, highlights) are rendered on the device's screen via the web browser. AdopTrix adapts visually to various display resolutions and sizes typical of desktop and laptop monitors.
2. **Network Connectivity:** AdopTrix depends on the device's internet connection—whether wired, Wi-Fi, or cellular—for real-time communication with backend services such as Firebase and AI processing APIs. Reliable network connectivity is essential for delivering AI-driven assistance and analytics in real-time.
3. **Input Devices:** User interactions with AdopTrix occur through standard input devices such as keyboard and mouse (or touchpads), facilitated by the browser environment.
4. **Storage:** The extension may utilize the browser's local storage or Firebase to cache user preferences, session data, or temporary information to improve performance and user experience.

AdopTrix is designed to operate within the constraints and capabilities of the user's device and browser without requiring specialized hardware interfaces. Its performance and usability depend on the underlying hardware indirectly through the browser's interaction with these components.

## 3.4 Software Interfaces

AdopTrix interfaces with several software components and services to enable its core functionalities:

1. **Firebase Services:**

    - AdopTrix uses Firebase as its primary backend platform, leveraging services such as Authentication, Fire store (NoSQL database), and Cloud Functions.
    - Data items exchanged include user authentication tokens, user profile data, session information, and real-time application data.
    - This interface handles user management, data storage, synchronization, and serverless backend logic.

2. **Browser Environment (Chrome API):**

    - AdopTrix operates as a Chrome browser extension and uses the Chrome Extensions API to interact with browser features such as tabs, storage, messaging, and UI overlays.
    - Data items managed include user preferences, session state, UI event data, and communication between background scripts and content scripts.
    - This interface allows AdopTrix to embed AI-powered assistance and tooltips directly into web pages and manage extension lifecycle events.

3. **AI Processing APIs:**

    - AdopTrix integrates with third-party AI and machine learning APIs (for example, Google Cloud AI or custom AI inference services) to provide real-time contextual assistance, data analysis, and natural language processing.
    - Data exchanged includes text inputs, user interaction context, and AI-generated responses or recommendations.
    - This interface leverages advanced AI capabilities to enhance the extension's usability and intelligence.

4. **UI/UX Design Framework (Figma):**

    - Although not a runtime software interface, Figma is used extensively in the design phase for prototyping and UI/UX specifications.
    - Design artifacts and component specifications from Figma guide the development of the extension's user interface, ensuring consistency and adherence to design standards.

These software interfaces enable seamless communication, data management, and AI-driven functionalities across the extension's components. Detailed API references and integration protocols are documented separately to ensure smooth interoperability and maintainability.

## 3.5 Communications Interfaces

AdopTrix utilizes multiple communication interfaces to ensure smooth and secure interaction between the browser extension, backend services, and external APIs:

1. **HTTP/HTTPS Protocol:**

    - AdopTrix communicates with Firebase backend services and third-party AI APIs using HTTP/HTTPS protocols for all data exchanges.
    - All requests and responses use JSON (JavaScript Object Notation) format to ensure efficient and standardized data transfer.
    - Secure communication is ensured by enforcing HTTPS with TLS encryption to protect sensitive user data during transit.

2. **Firebase Real-time Communication (Realtime Database / Fire store):**

    - AdopTrix leverages Firebase's real-time database or Fire store for instantaneous data updates and synchronization across clients.
    - These Firebase services use WebSocket or similar persistent connection protocols behind the scenes to maintain real-time bidirectional communication.
    - This interface supports live data updates, notifications, and synchronization of user preferences or session information without explicit polling.

3. **Chrome Extension Messaging API:**

    - For internal communication between the background scripts, content scripts, and popup UI within the extension, AdopTrix uses Chrome's messaging API.
    - This messaging system facilitates the exchange of data such as user commands, AI interaction context, and UI state changes, enabling coordinated behaviour within the extension components.

4. **Third-party AI Processing APIs:**

    - AdopTrix interacts with external AI service APIs.
    - Entered data, user input text, or contextual information is sent for processing, and AI-generated responses are received to provide intelligent assistance.

These communication interfaces enable AdopTrix to provide responsive, secure, and intelligent assistance by integrating browser extension capabilities with backend and AI services. Implementation details follow best practices for web security and performance, ensuring a robust communication backbone for the extension.

# 4. System Use Cases

## 4.1 Use Case: Initiate AI Chatbot Query (UC-01)

1. **Objective:** Allow users to interact with the AI-powered chatbot within the browser extension to get context-based assistance on the web application they are using. This helps users quickly understand features and get guidance without leaving the page.

2. **Priority:** High
3. **Source:** End-user
4. **Actors:** User: Initiates queries through the chatbot interface in the browser.

5. **Flow of Events:**

   - Basic Flow: The user activates the AdopTrix extension in the browser. The chatbot interface appears as an overlay or popup. The user types or speaks a natural language query. The chatbot processes the query using the integrated Large Language Model (LLM) API. The chatbot displays suggestions, instructions, or actionable buttons. The user selects actions or follows guidance provided by the chatbot.
   - Alternative Flow(s): The user may close or minimize the chatbot at any time. The user may request help on a different topic or page.
   - Exception Flow(s): If network connectivity is lost, the chatbot shows an error message and suggests retrying later.

6. **Includes:** None
7. **Preconditions:** The user has installed and enabled the AdopTrix browser extension. The device has internet connectivity for API communication.

8. **Postconditions:** The user receives context-aware assistance to navigate or use the web application efficiently. User actions may be logged for analytics and improvement.

9. **Notes/Issues:** Ensure quick response times for a smooth user experience. Handle edge cases where chatbot cannot understand the query gracefully.

## 4.2 Use Case: Display Guided UI Walkthrough (UC-02)

1. **Objective:** Provide users with a step-by-step interactive walkthrough overlay that highlights and explains UI elements of a web application to assist learning and onboarding.
2. **Priority:** High
3. **Source:** End-user
4. **Actors:** User: Requests or triggers the guided walkthrough for a specific software feature or page.

5. **Flow of Events:**

- Basic Flow: The user navigates to a web application page. The AdopTrix extension detects the page and available guides. The user triggers the walkthrough manually or it is suggested based on queries. The extension highlights UI elements sequentially, displaying dialog boxes with brief instructions. The user follows the highlighted steps, scrolling as necessary. The walkthrough ends when all steps are completed or the user exits.

- Alternative Flow(s): The user can pause, skip, or exit the walkthrough at any time. The walkthrough content updates dynamically based on user progress.
- Exception Flow(s): If the page elements change or fail to load, the walkthrough pauses and notifies the user.

6. **Includes:** None
7. **Preconditions:** The user has the AdopTrix extension active on the current web page. The walkthrough guide for the page or feature is available.

8. **Postconditions:** The user gains improved understanding of the UI and application features. Walkthrough completion is logged for analytics.

9. **Notes/Issues:** Ensure guides are updated with web application changes to avoid broken walkthroughs. Support multi-step interactions smoothly.

## 4.3 Use Case: Manage Dialog Boxes During Walkthroughs (UC-03)

**1. Objective:** To enable users to effectively manage dialog boxes that appear during guided UI walkthroughs, allowing them to close, minimize, or navigate through these dialogs seamlessly. This enhances the user experience by giving control over on-screen assistance.

**2. Priority:** High

**3. Source:** End-user

**4. Actors:**

- **User:** Interacts with the dialog boxes during the walkthrough sessions.
- **System:** Displays and manages dialog boxes dynamically.

**5. Flow of Events:**

- Basic Flow: The system detects when a guided walkthrough step requires displaying a dialog box near a UI element. The dialog box appears with relevant instructions or information. The user can choose to:
    - Close the dialog box, ending the current step assistance.
    - Minimize the dialog box to keep it accessible but unobtrusive.
    - Navigate forward or backward through the walkthrough steps using dialog controls.
- Alternative Flow: If the user minimizes the dialog box, they can restore it at any time by clicking a floating icon or notification.
- Exception Flow: If the dialog box fails to render correctly, the system will retry rendering or display a fallback message.

**6. Includes:**

- Feature to detect UI elements dynamically (UC-02).
- Feature to display interactive chatbot responses (UC-01).

**7. Preconditions:** The user must be on a supported web page within the Chrome browser. The Digital Adoption Assistant extension must be active and loaded.

**8. Postconditions:** The walkthrough state is saved according to user interaction with the dialog boxes. The user has full control over the dialog box display during the session.

**9. Notes/Issues:** The system is currently optimized for compatibility with the Chrome browser to ensure reliable rendering and interaction of dialog boxes.

# 5. Other Nonfunctional Requirements

## 5.1 Performance Requirements

- **Fast Response Time:** All user interactions within the plugin, including loading the plugin interface, fetching tour data, and highlighting elements on the page, should complete within seconds to ensure a smooth and responsive user experience.
- **Efficient Screenshot Capture and Upload:** The process of capturing screenshots during the tour recording must happen almost instantly (in milliseconds) without noticeable delay to the admin user. Uploads to cloud storage should be asynchronous to prevent blocking the UI.
- **Minimal Plugin Size:** The entire plugin package, including all scripts, styles, and assets, must remain under **50MB** to ensure quick installation and updates, as well as to reduce browser memory consumption and performance impact.
- **Optimized Data Storage:** Image sizes of screenshots should be compressed appropriately before upload to Firebase Storage to balance between image quality and storage/bandwidth usage, ensuring efficient retrieval during user playback.
- **Resource Usage Constraints:** The plugin should minimize CPU and memory usage during both recording and playback, to avoid slowing down the user's browser or interfering with normal website operation.
- **Scalable Performance:** The backend services (Firebase Fire store and Storage) must be structured and queried efficiently to handle multiple concurrent users without degradation in response time.

## 5.2 Safety Requirements

- **Data Privacy:** All user data, including login credentials, screenshots, and tour details, must be securely stored and transmitted using encryption protocols (e.g., HTTPS, Firebase's built-in security). No sensitive information should be exposed or logged unintentionally.

- **User Consent:** The plugin must notify users when screen captures or data collection activities are performed and obtain explicit consent before recording any session or clicks.
- **Error Handling and Recovery:** The system should gracefully handle errors such as failed network connections, unauthorized access attempts, or corrupted data, ensuring no data loss or crash occurs during critical operations.
- **Safe Access Control:** Only authorized admin users should have access to the tour creation and editing features. User roles must be strictly enforced to prevent privilege escalation.
- **API Security:** All API tokens and keys used by the plugin, including those for AI services, should be stored securely and never exposed in client-side code. Communication with APIs must be authenticated and protected against unauthorized access.
- **Browser Compatibility & Stability:** The plugin should be tested across major browsers and versions to avoid any unsafe behaviour or crashes during usage.

## 5.3 Software Quality Attribute

1. **Adaptability**: The Digital Adoption Platform must be capable of adjusting to changes in user behaviour, software updates, and UI designs. This ensures it remains effective across evolving web applications and enterprise tools without major rework.
2. **Availability**: The platform should remain active and responsive during user sessions with minimal downtime. Continuous availability ensures real-time guidance is always accessible during software use.
3. **Correctness**: The system must detect and overlay instructions on the correct UI elements. Accurate responses from the AI chatbot and precise walkthrough steps are essential for delivering useful guidance.
4. **Flexibility**: The platform must adapt to various web applications and user scenarios. Users should be able to customize walkthrough paths, chatbot interactions, and plugin behaviour to suit specific workflows.
5. **Interoperability**: The DAP must integrate smoothly with diverse web applications. Support for standard web technologies ensures it can be embedded in or applied to many platforms without compatibility issues.
6. **Maintainability**: The system should be modular and well-documented to allow easy updates when application UIs change. Maintainable code enables faster bug fixes, feature enhancements, and onboarding of new developers.
7. **Portability**: The browser extension should support multiple environments, starting with Google Chrome and optionally extending to Firefox, Edge, etc. Portability increases reach across different user systems.

8. **Reliability**: The system must consistently provide the intended assistance without unexpected failures. Guided walkthroughs, overlay dialogs, and chatbot features should function reliably under normal usage conditions.
9. **Reusability**: UI components, walkthrough templates, and chatbot modules should be reusable across various applications and onboarding scenarios, reducing development time for future implementations.
10. **Robustness**: The system must gracefully handle missing elements, dynamic UI changes, or incorrect user inputs. It should prevent crashes and provide fallback options to maintain user flow.
11. **Testability**: Automated tests and debugging utilities should be built into the system to allow quick validation of walkthrough flows and UI element recognition accuracy after any update.
12. **Usability**: The interface — from overlay dialogs to chatbot UI — must be intuitive for users of all technical levels. Clear guidance, tooltips, and minimal disruption ensure a smooth and engaging user experience.

## 6. System Requirement Chart

| ID | Priority | Type | Contained in Use Case(s) | Description |
|---|---|---|---|---|
| R1 | High | Functional (F) | UC-01, UC-02, UC-03 | The system shall allow users to interact with the AI chatbot via natural language queries. |
| R2 | High | Functional (F) | UC-02 | The system shall provide guided UI walkthroughs with step-by-step highlighting and instructions. |
| R3 | High | Functional (F) | UC-01, UC-02 | The system shall dynamically detect and highlight web page elements using CSS selectors. |
| R4 | Medium | Functional (F) | UC-01 | The system shall display clickable suggestions and action buttons in chatbot responses. |
| R5 | High | Functional (F) | UC-03 | The system shall allow users to close, minimize, or navigate dialog boxes during walkthroughs. |
| R9 | High | Non-functional (NF) | UC-03 | The system shall be compatible for chrome browsers |

# 7. Appendix

## 7.1 Acronyms and Abbreviations

- **DAP**: Digital Adoption Platform
- **LLM**: Large Language Model
- **MVP**: Minimum Viable Product
- **DOM**: Document Object Model
- **CRM**: Customer Relationship Management
- **UI**: User Interface
- **API**: Application Programming Interface
- **WCAG:** Web Content Accessibility Guidelines
- **HTTP:** Hypertext Transfer Protocol
- **HTTPS**: Hypertext Transfer Protocol Secure
- **SRS**: Software Requirements Specification
- **UC**: Use Case
- **UX**: User Experience
- **NLP**: Natural Language Processing
- **SDK**: Software Development Kit
- **JS**: JavaScript
- **CSS**: Cascading Style Sheets
- **HTML**: Hypertext Markup Language
- **SQL / NoSQL**: Structured Query Language / Not Only
- **ID**: Identifier