An application for augmenting technical documentation using AR and AI

CS310: Third Year Project

Ani Bitri

Supervisors: Dr. Paris Giampouras, John McNamara

Department of Computer Science

University of Warwick and IBM

January 2025

1   Introduction

The discipline of software engineering is fundamentally abstract. As software systems have evolved from monolithic mainframes to distributed, microservice-based architectures, developers heavily rely on technical documentation to understand, maintain and build upon existing tools, systems and frameworks. However, traditional documentation formats, such as PDFs, static web pages and printed manuals, often fall short in conveying complicated concepts, especially when it comes to technical diagrams and schematics. These visuals are essentials for understanding the systems they represent, yet they often make it more difficult and confusing for users and other developers to grasp the underlying ideas.

The core objective of this project is to offer a solution that enhances the user experience when interacting whith technical documentation while still preserving the integrity and accuracy of the original content. By combining Augmented Reality (AR) and Aritificial Intelligence (AI), the project aims to create an application that transforms static documentation and diagrams into interactive, immersive and context-aware experiences.

Insert paragraph here for AI assistant

2   Research

2.1   Literature Review and Existing Solutions

2.1.1   DGT-AR

DGT-AR (Dependency Graph Tool in Augmented Reality) is an academic project which utilizes the Microsoft HoloLens and its core premise is similar, in that it aims to visualize software architecture to aid developers in understanding complex systems. This solution renders a software's dependency graph as a 3D node-link structure in the user's physical environment. The researchers on this project found that the virtual space of AR addressed the limitation that a 2D screen has when visualizing large and complex graphs. By using the HoloLens, developers could walk around the graph, using motion parallax to better percieve what the graph represented.

2.1.2   "Retargeting Technical Documentation to Augmented Reality" Paper by P. Mohr et al.

This paper presents a system which automatically

2.1.3   Differentiation from Existing Solutions

This project differentiates itself from existing solutions, such as DGT-AR and the work by P. Mohr et al., in several key aspects:

-
-
-
-

2.2   Usability & Accessibility

Since the solution involves creating an application, it is cruical to ensure that the app is user-friendly and accessible to any potential user. While not all features an

2.3   Development

3   Requirements

3.1   Functional Requirements

1. **Augmented Reality System:**

   - FR-1: The system shall allow users to open the camera interface within the mobile application to scan technical documents.

   - FR-2: The system shall allow the user to upload technical documents from their device storage.

   - FR-3: The system shall recognize diagrams, images, flowcharts, and schematics within the scanned or uploaded documents using image recognition techniques.

- **FR-4:** The backend shall extract diagram elements and textual information from the recognized diagrams using IBM Granite Vision.

- **FR-5:** The system shall generate interactive AR overlays that provide explanations of components, relationships, and interactions within the diagrams.

- **FR-6:** The system shall allow users to interact with the AR overlays to explore additional information about the diagram components and their relationships.

- **FR-7:** The system shall provide visual cues to confirm successful recognition and overlay placement.

2. **AI Assistant:**

- **FR-8:** The system shall allow users to input natural language queries related to the scanned or uploaded technical documents.

- **FR-9:** The backend shall use the Granite AI model to process user queries and generate relevant responses based on the document content.

- **FR-10:** The AI shall combine diagram data from Granite Vision with textual information from the documents to provide comprehensive answers.

- **FR-11:** The system shall display the AI-generated responses within the mobile application

- **FR-12:** The AI assistant shall retain context from previous interactions to provide coherent and relevant responses.

3. **Data Management and Communication:**

- **FR-13:** The app shall communicate with the backend over HTTPS with token-based authentication to ensure secure data transmission.

- **FR-14:** The backend shall handle user requests, forward images to IBM Granite Vision, combine results with IBM Granite AI, and return structured JSON responses to the mobile app.

- **FR-15:** The system shall implement caching mechanisms to store previously processed diagrams and AI responses for improved performance.

- **FR-16:** The system shall log user interactions and queries for analytics and improvement purposes.

- **FR-17:** The system shall provide error handling mechanisms to manage failures in image recognition, AI processing, or network connectivity, and inform users appropriately.

4. **User Interface and Experience:**

- **FR-18:** The mobile application shall present a home screen with navigation options to access camera view, document upload, AI assistant, and settings.

- **FR-19:** The system shall provide an interactive chat interface for AI interactions, showing both user queries and AI responses in a conversational format.

- **FR-20:** Users shall be able to switch between text input, speech input and AR interaction modes seamlessly.

4 System Architecture

Provide a high-level overview of the system architecture. Include diagrams where appropriate to illustrate the system components and their interactions.

4.1 Design Pattern

The design pattern which will be adopted for this project is the Model-View-ViewModel (MVVM) pattern. This pattern is well-suited for applications that require a clear separation of concerns between the user interface and logic, making it easier to manage and test the application. Furthermore, the MVVM pattern facilitates a more modular and maintainable codebase, which is essential for the iterative development process planned for this project. The image below illustrates the MVVM architecture which will be used in the project.

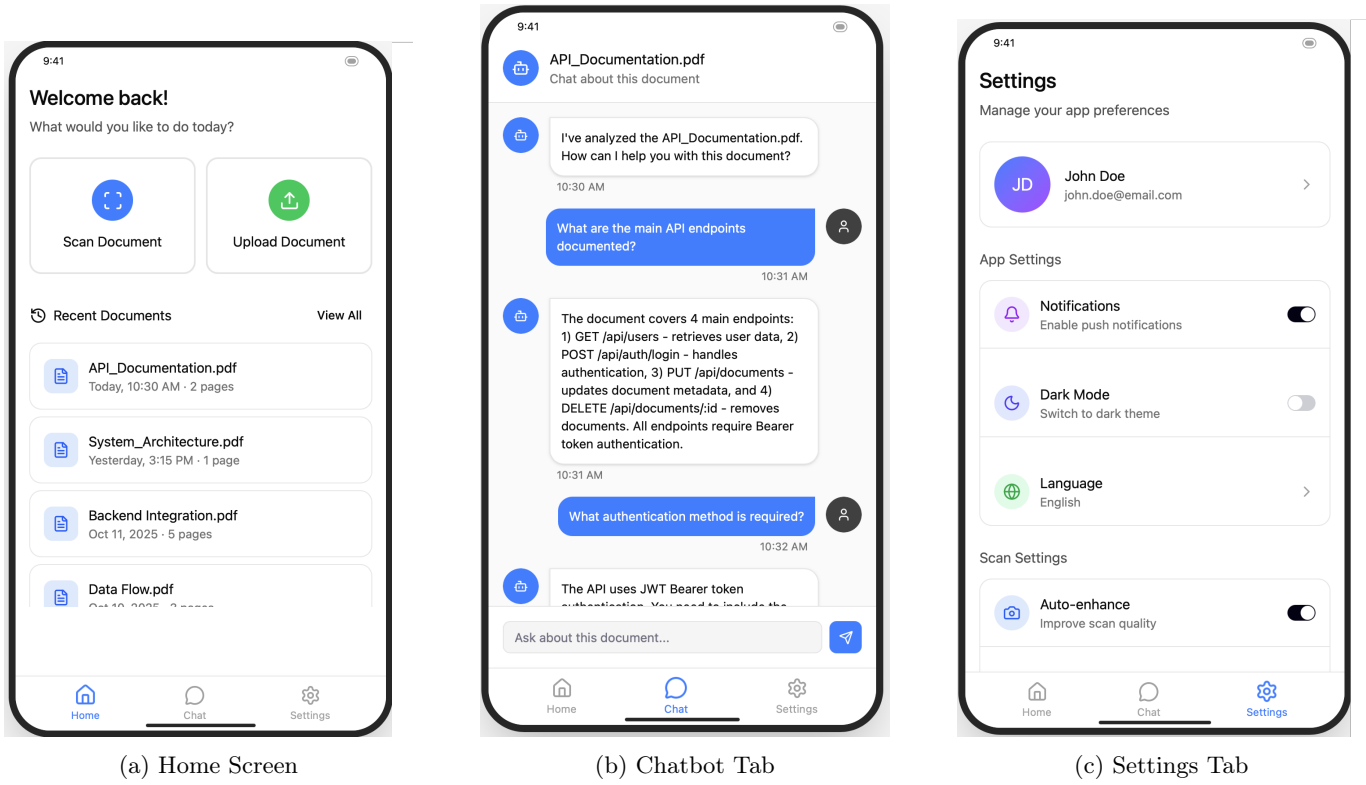| (a) Home Screen | (b) Chatbot Tab | (c) Settings Tab |

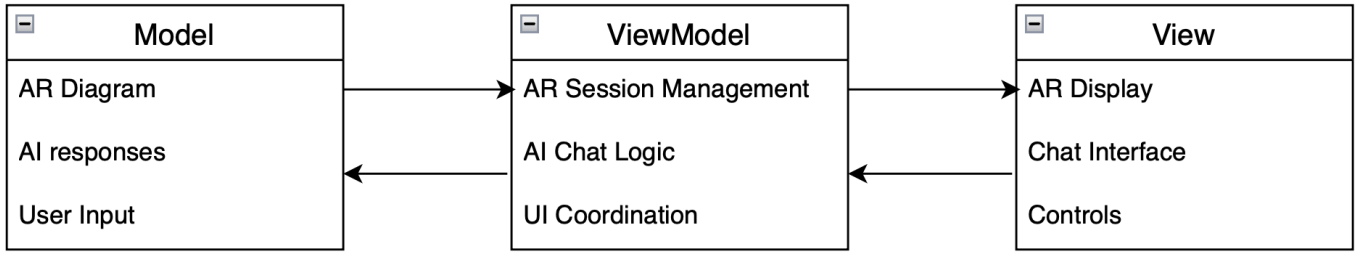Figure 3: Proposed UI Design for the Application

Figure 2: MVVM Architecture Pattern

4.2 Frontend Design

4.2.1 User Interface and User Experience

The application will feature a user-friendly interface that allows users to easily navigate through the app and access its features. The UI will be designed to be intuitive and responsive, ensuring a seamless user experience. When the user open the app, they will be presented with a simple home screen with options to scan a document, upload a document or access the history of previously scanned documents. Besides the home screen, the app will also include a tab for the chatbot, where users can interact with the AI assistant, and a settings tab for configuring app preferences. The image below illustrates the proposed UI design for the application:

Upon selecting the scan option, the user will be directed to the camera interface, where they can capture pictures of the technical documentation. Alternatively, the user can choose to upload the document from their device's storage and access the same features. After scanning or uploading a document, the app will process the text, images and diagrams, and give the user the option to view the AR overlays or interact with the AI assistant. The AR overlays will provide interactive elements that explain components, relationships and interactions within the diagrams based on the user's input and the information extracted from the document. The image below illustrates a flow chart of the app's user interface and navigation:

4.2.2 Error Handling and Feedback

To ensure a smooth user experience, the application will include robust error handling mechanisms. Internal errors will be handled gracefully and automatically, with appropriate messages displayed to the user upon interruptions. For example, if the app fails to recognize a diagram or if the AI assistant cannot process a query, the user will be informed of the issue and provided with suggestions for resolution. External errors will be handled by providing informative error

messages to guide users in case of issues such as failed scans or uploads, network connectivity problems or AR tracking errors. Furthermore, error prevention strategies will be implemented to minimize their occurrence and impact on the user experience. Such strategies include input validation, network status checks and AR tracking optimizations. This approach will ensure that users can effectively utilize the app's features without being hindered by technical issues.

4.2.3 Integration with Backend and Data Flow

The frontend of the application will communicate with the backend services to process the scanned or uploaded documents and retrieve relevant information. The data flows as follows:

1. The user scans or uploads a document using the app's interface.

2. The frontend validates the input and sends the document data to the backend for processing.

3. The validated data is sent to the backend services for text recognition, diagram analysis and AI processing.

4. The backend processes the data and generates AR overlays and AI responses based on the document content.

5. The generated AR overlays and AI responses are sent back to the frontend for display to the user.

6. The user interacts with the AR overlays and AI assistant, and any further queries or actions are sent back to the backend for processing.

4.3 Backend Design

4.3.1 Overview

The backend of the application will act as the intelligence and coordination layer that connects the React Native frontend to IBM's AI services. Its primary function is to process the scanned or uploaded documents and other user inputs, understand their content, and generate appropriate AR overlays and AI responses. The backend ensures that heavy computational tasks are offloaded from the mobile device, providing a seamless user experience.

4.3.2 Workflow

The backend workflow consists of several key steps:

1. **Request Reception:** The backend receives the scanned images or uploaded from the frontend via RESTful API calls.

2. **Image and Data Processing:**
   - If an image is recieved, it is processed using IBM Granite Vision to extract text and identify diagrams, returning structured data.
   - If text data is recieved, it is analyzed to identify key components, relationships and interactions.

3. **AI Reasoning and Response Generation:** The backend composes of a prompt that combines:
   - Extracted diagram information from IBM Granite Vision.
   - Relevant documentation snippets.
   - The user's query or interaction context.

   The composite prompt is sent to IBM Granite, which then generates a natural language response.

4. **Response Delivery:** The backend merges the AI-generated response with structured metadata and send a single JSON payload back to the mobile app. The fronted then displays the AR overlays and AI responses to the user.

4.3.3 Data Management and Caching

To improve efficiency, the backend implements:

- **Caching:** Previously processed diagrams are stored to reduce redundant processing using Redis or local storage.
- **Error Handling:** AI or network failures return structured JSON error messages to the frontend for user notification.
- **User session logs:** Stored in a database for analytics and improvement (PostgreSQL or Firebase).

4.4 Technology Stack

The project will utilize the following technologies:

- **Frontend:** React Native for cross-platform mobile development, ARCore/ARKit/ViroReact for augmented reality functionalities.

- **Backend:** Node.js with Express for server-side logic, IBM Granite 4.0 for AI capabilities, IBM Granite Vision for image recognition, and MongoDB for data storage.

- **Development Tools:** Visual Studio Code for code editing, Git for version control, REST APIs for communication between frontend and backend.

4.5 Testing

5 Current Progress

5.1

6 Project Management

7 Outlining Future Steps

7.1 Updated Timeline

7.2 Web App

7.3 Additional Features

8 Project Evaluation

9 Legal, Social, Ethical and Professional Issues & Considerations