**صورة تحتوي على شعار, رمز, علامة تجارية, دائرة

تم إنشاء الوصف تلقائياً**

**Faculty of Computer and Information Technology**

**Department of Computer Engineering**

**CPE 591: Graduation Project II Report**

**[AI-Based interactive digital content application: search, summarize, save, translate, integrate, chat and present]**

**Noor Saleh Sharkawi 143673**

**Aya Amjad Mahmoud 145227**

**Supervised By**

**Dr.Lo'ai Tawalbah**

**Winter 2024-2025**

**2. Acknowledgment**

**//Acknowledgment (Those who truly helped you achieve your goals and made contributions)**

We would like to express our deepest gratitude to Lo'ai Tawalbah our supervisor , for invaluable guidance , continuous support, all the feedback that give us throughout this project .

We extend our appreciation to JUST,CPE for providing us with the resources and knowledge that enabled us to complete this project successfully.

A special thanks to our families and friends for their unwavering encouragement and patience throughout this journey. Their support has been a constant source of motivation.

Lastly, we are grateful to everyone who contributed, directly or indirectly, to the success of this project. Your help and guidance have played a significant role in bringing our vision to reality.

**3. Abstract**

In the digital era, efficient content management and accessibility have become essential for students, researchers, and professionals. This project presents an AI-Based Interactive Digital Content Application, designed primarily for file management and document interaction rather than traditional e-book reading. The application integrates Flutter for the frontend and Python with Flask for the backend, ensuring a seamless user experience.

The system incorporates AI-powered features, including automated summarization and a chatbot, both utilizing the Google Gemini API. Additionally, the application supports searchable PDFs, allowing users to locate key information with highlighted results. Translation functionality is provided through a simple widget similar to Google Translate. To enhance productivity, users can manage notes and to-do lists, all synchronized using Firebase for real-time data management.

Security and user authentication are handled via Firebase Authentication, while Firestore serves as the primary database for storing user-generated content. The implementation of bookmarks and a ‘Continue Reading’ feature ensures an enhanced user experience tailored to content engagement.

This project aims to provide a smart and efficient solution for digital content organization, catering to students and professionals alike. By integrating AI and cloud-based services, the system offers a powerful and intuitive platform that simplifies document management, enhances productivity, and improves information retrieval.

**4.1 Table of content**

// This will be generated automatically once your report is complete, based on the headings and subheadings.

**4.2 List of Figures**

//This section lists all the figures (diagrams, charts, UI screenshots, etc.) included in your report, with their corresponding page numbers.

**4.3 List of Tables**

//Similar to the list of figures but for tables (e.g., database schemas, performance comparisons).

**5. Introduction**

**5.1 Statement of the Problem**

**/\*** **This section defines the problem your project aims to solve \*/**

Draft:  
In today’s digital world, managing large volumes of documents and extracting relevant information efficiently remains a significant challenge. Traditional e-book readers focus primarily on displaying content rather than enhancing productivity. Users often struggle with finding key information quickly, summarizing long documents, and managing notes effectively. Additionally, current solutions lack seamless AI-powered assistance for searching, summarizing, and organizing content in an intuitive way.

This project aims to bridge this gap by developing an AI-Based Interactive Digital Content Application that integrates automated summarization, intelligent search, note-taking, and cloud synchronization, providing users with an enhanced content management experience.

**5.2 Significance of the Project**

**/\*This section explains why your project is important.\*/**

Draft:  
The significance of this project lies in its ability to improve digital content interaction and accessibility. By leveraging AI-powered summarization and intelligent search capabilities, users can quickly extract key insights from lengthy documents. Additionally, the integration of cloud-based storage (Firebase) ensures data availability across multiple devices, making it an ideal tool for students, researchers, and professionals.

Unlike traditional e-book readers, this application provides personalized productivity tools, such as to-do lists, bookmarks, and a ‘Continue Reading’ feature, to enhance user experience. The project contributes to the advancement of AI in digital education by introducing smart functionalities that simplify document interaction.

**5.3 Goals**

**This section outlines the objectives of your project.**

Draft:  
The primary goal of this project is to develop an AI-Based Interactive Digital Content Application that enhances document management and content accessibility. The specific objectives include:

1. Implement AI-powered summarization to generate concise summaries of uploaded documents using the Google Gemini API.
2. Develop an AI chatbot to assist users with general inquiries about the content.
3. Enable searchable PDFs with keyword highlighting for faster information retrieval.
4. Provide a built-in translation widget for multi-language support.
5. Integrate cloud storage via Firebase for seamless access to notes, bookmarks, and to-do lists across multiple devices.
6. Ensure secure authentication and data management using Firebase Authentication and Firestore.
7. Optimize the user experience by adding features like ‘Continue Reading’ and bookmarking.
8. Maintain a user-friendly UI/UX with an intuitive interface developed in Flutter.

**5.4 Contemporary Issues**

**This section discusses current challenges related to digital content management and how your project addresses them.**

Draft:  
Several contemporary issues affect the efficiency of digital document management, including:

1. Information Overload – With the vast amount of digital content available, users struggle to extract key insights quickly.
   * Solution: AI-driven summarization provides concise, relevant information.
2. Inefficient Search Mechanisms – Many existing e-book readers lack advanced search features, making it difficult to locate specific content.
   * Solution: Our app supports keyword-based PDF searching with highlighted results for better accessibility.
3. Limited Productivity Features – Most document readers do not include built-in note-taking, bookmarks, or to-do lists, forcing users to rely on external apps.
   * Solution: Integrated notes and task management enhance productivity.
4. Lack of AI Assistance – Few document management systems incorporate intelligent AI chatbots for user support.
   * Solution: A chatbot powered by Google Gemini API will assist users with queries.

By addressing these contemporary challenges, our project enhances digital learning and professional workflows.

**5.5 Impact on Society**

**This section explains how your project benefits users and society.**

Draft:  
The proposed application will have a significant impact on students, researchers, and professionals by:

1. Improving Learning and Research Efficiency – AI summarization and intelligent search reduce time spent reviewing lengthy documents.
2. Enhancing Accessibility – Features like translation and keyword search make content more user-friendly for diverse audiences.
3. Increasing Productivity – Built-in notes, to-do lists, and bookmarks provide a seamless workflow.
4. Encouraging Digital Transformation – The integration of AI and cloud services supports modern digital education and workplace productivity.
5. Reducing Cognitive Load – Users can focus on essential information without getting overwhelmed by unnecessary details.

This project empowers users with smart digital tools, making information management more efficient, accessible, and intuitive.

**5.6 Initial Constraints**

**This section lists the limitations faced during development.**

Draft:  
The project faced several initial constraints that influenced the design and implementation:

1. AI API Limitations – The Google Gemini API may have rate limits or restricted access, affecting performance.
2. Cloud Storage Costs – Firebase storage and Firestore database have cost implications as data usage increases.
3. Computational Complexity – AI summarization and chatbot responses require efficient processing to maintain real-time performance.
4. Platform Compatibility – Ensuring seamless integration between Flutter (frontend) and Flask (backend) required careful optimization.
5. User Interface Design Challenges – Balancing feature richness with UI simplicity was a key consideration.

Despite these constraints, strategic planning and optimization techniques help mitigate these challenges, ensuring a smooth and functional user experience.

**6.Project Planning and Task Definition.**

**6.1 Task Identification**

**This section will involve identifying the core tasks required to complete your graduation project. Here’s a possible breakdown of tasks for your E-book reader app:**

Task Breakdown:

1. Research and Design Phase:
   * Define Requirements: Clarify the app’s functionalities (file management, summarization, chatbot, etc.).
   * UI/UX Design: Sketch wireframes for the app, focusing on the user interface and user experience.
   * System Architecture Design: Define the technical architecture (how Python and Flask will interact with Flutter, Firebase usage, etc.).
2. Development Phase:
   * Setup Development Environment: Install and configure all necessary tools (Flutter, Firebase, Python, Flask, etc.).
   * Create Core App Structure: Build the basic structure for navigation, layout, and integration between Flutter and Python.
   * Backend Development (Flask & Firebase):
     + Implement API endpoints for file uploads, user authentication, and productivity tools (notes, to-do lists).
     + Integrate Firebase Authentication and Firestore.
   * Frontend Development (Flutter):
     + Develop the user interface based on the UI/UX design.
     + Integrate PDF handling using Flutter packages (highlighting, searchable PDFs).
     + Implement the "Continue Reading" feature, bookmarks, and PDF uploads.
   * AI Features:
     + Integrate Gemini API for summarization and chatbot functionality.
     + Test and refine AI responses for the chatbot.
   * Localization (Translation): Implement the translation widget.
3. Testing Phase:
   * Unit Testing: Test individual components of both the backend and frontend.
   * Integration Testing: Ensure that the app works seamlessly across all features (Firebase, AI, PDF handling).
   * User Testing: Conduct testing with a small group of users to gather feedback on the app’s usability.
4. Deployment Phase:
   * Prepare for Launch: Finalize the app for deployment on relevant platforms.
   * Deploy the App: Deploy the app and ensure that Firebase services are connected.
5. Documentation:
   * Technical Documentation: Document the code and explain the app's features, structure, and APIs.
   * Project Report: Begin writing the final report, including chapters on planning, design, implementation, and results.

**6.2 Timeline**

Week 1-2: Research & Design

* Define app requirements and core functionalities.
* Start UI/UX design (wireframes, flow).
* Plan system architecture and tech stack.
* Research Firebase and AI integration.

Week 3-5: Backend & Frontend Setup

* Setup development environment (Flutter, Python, Firebase).
* Design and implement basic structure of the app (Flutter navigation, initial screens).
* Set up Firebase Authentication and Firestore.
* Develop initial backend APIs in Flask.

Week 6-8: Core Functionality Development

* Implement file upload, note-taking, and to-do list features.
* Start integrating AI features (summarization, chatbot with Gemini API).
* Develop PDF handling features (searchable PDFs, highlighting, etc.).

Week 9-10: Testing

* Conduct unit and integration tests.
* Refine AI chatbot responses.
* User testing for feedback and usability.

Week 11-12: Finalizing & Documentation

* Finalize all features, ensuring smooth integration.
* Prepare the app for deployment (test on actual devices).
* Complete technical documentation.
* Begin writing the final project report.

**7. Literature Review**

**/\*The Literature Review chapter will provide an overview of existing research, applications, and technologies related to your app. The purpose is to justify your project by identifying gaps in current solutions and explaining how your app improves upon them.\*/**

**7.1 Introduction**

* Explain the purpose of the literature review.
* Describe how the review will cover related research on AI-based content applications, PDF management apps, chatbots, and summarization tools.

**7.2 AI in E-File Readers and Content Management**

* Review existing E-File reader applications (e.g., Kindle, Adobe Acrobat, Google Play Books).
* Discuss how AI has been applied in content summarization, text recognition, and PDF management.
* Analyze limitations in these applications (e.g., lack of personalized summaries or AI-driven content recommendations).

**7.3 Summarization Techniques Using AI**

* Explain different AI summarization methods (e.g., extractive and abstractive summarization).
* Discuss research on models like Gemini, GPT, or BERT for text summarization.
* Provide comparisons on accuracy, efficiency, and adaptability of AI models.

**7.4 AI Chatbots for Content Assistance**

* Analyze how AI chatbots assist users in understanding content.
* Review studies or applications where chatbots provide educational support.
* Highlight challenges in chatbot response accuracy and user engagement.

**7.5 PDF Management and Searchable PDFs**

* Discuss existing PDF management tools and applications (e.g., Foxit PDF Editor, PDFelement).
* Explore advancements in PDF text recognition (using OCR) and search functionalities.
* Identify how your app improves this aspect using Flutter packages.

**7.6 Firebase in Mobile App Development**

* Provide an overview of Firebase Authentication and Firestore.
* Discuss why Firebase is widely used for real-time data management.
* Explain the security measures Firebase offers.

**7.7 Gaps and Justification for Your Project**

* Summarize the identified gaps in existing solutions.
* Explain how your app addresses these gaps through its features.
* Justify the importance of integrating AI for better user experience and productivity.

**Chapter 8: Preliminary Design**

**8.1 Concept**

The preliminary design concept of the App is centered around providing users with an intuitive and efficient platform for managing and interacting with their digital documents. The app incorporates features such as AI-powered summarization ,AI-powered Chatbot, translation, note-taking, and to-do lists. Users can upload, read, and search PDF files, with AI-generated summaries providing quick insights. Additionally, Firebase integration ensures real-time synchronization and secure user authentication.

**8.2 Concept Evaluation and Selection**

Multiple concepts were evaluated to ensure the most effective and user-friendly design. Factors considered included:

* **User Experience:** Ensuring a seamless, engaging interface.
* **Performance:** Optimizing AI response times and PDF loading speed.
* **Scalability:** Supporting large document uploads and efficient data management.
* **Security:** Implementing Firebase Authentication for robust user management.
* **Accessibility:** Providing features such as translation for multilingual users.

After evaluation, the chosen concept integrated Google Gemini API for AI features and Firebase for backend services. This combination offered the best balance between functionality, reliability, and cost-efficiency.

**8.3 Design Constraints**

The preliminary design faced the following constraints:

* **Platform Compatibility:** The app must function seamlessly on both Android and iOS devices.
* **Real-time Performance:** AI features must provide results with minimal latency.
* **Storage Management:** Efficiently manage PDF files without excessive memory consumption.
* **Network Dependency:** Some features, like AI-powered summarization, rely on active internet connectivity.
* **Data Security:** Ensuring user data privacy through Firebase authentication and Firestore database rules.

**8.4 Applicable Codes and Standards Used in the Design**

The design adheres to the following standards and codes:

* **IEEE 829-2008:** For software test documentation.
* **ISO/IEC 9126:** Ensuring software quality and usability.
* **OAuth 2.0:** For secure authentication using Firebase.
* **WCAG 2.1:** Maintaining accessibility standards for all users.
* **JSON API Standards:** For efficient data exchange between Flutter and the backend.

**8.5 Preliminary Analysis**

A preliminary analysis was conducted to estimate the app's performance and ensure it meets user requirements. Key analysis points include:

* **AI Summarization Accuracy:** Benchmark tests were performed to measure the accuracy and response time of the Google Gemini API.
* **Firebase Load Testing:** Evaluated to ensure reliable data synchronization under peak load.
* **User Interface Responsiveness:** Monitored to maintain smooth performance across various device specifications.
* **Error Handling and Logging:** Implemented detailed logs for error tracking and debugging.

The results from the preliminary analysis indicate that App is well-positioned to meet its goals, providing a responsive and user-friendly experience with accurate AI-driven assistance.

**Chapter 9: Detailed System Design**

**9.1 Engineering Analysis and Simulation**

The detailed system design of the App underwent extensive engineering analysis to ensure performance, reliability, and user satisfaction. Various simulations and performance tests were conducted, focusing on: /\*تغيير كل كلام القسم هاد\*/

* **AI Model Accuracy:** Evaluating the Google Gemini API's summarization and question-answering capabilities using diverse datasets.
* **System Load Simulation:** Stress testing the backend using Firebase to assess database response times under high user activity.
* **User Interface Performance:** Monitoring Flutter's rendering capabilities to ensure smooth interactions and minimize latency.
* **Error Recovery and Logging:** Implementing robust error-handling mechanisms with comprehensive log generation for debugging.

**9.2 Layouts, Drawings, Equipment Specifications**

**System Architecture Diagram:**

* A high-level architecture consisting of the Flutter frontend, Python Flask backend, Firebase Authentication, and Firestore database.
* APIs are used to communicate with the Google Gemini API for AI-powered functionalities.

**Component Overview:**

* **Frontend:** Developed using Flutter, featuring widgets for document viewing, note-taking, tranlation,and AI-based interactions.
* **Backend:** Built using Python with Flask for managing API requests and responses.
* **Database:** Firebase Firestore for real-time data storage and synchronization.
* **AI Integration:** Google Gemini API for summarization and chatbot assistance.

**Layout Design: /\*تعديل\*/**

* **Home Screen:** Displays uploaded files, recent activities, and quick access options.
* **Drawer to access all the tools**
* **Document Viewer:** Supports PDF viewing, searching, and annotating.
* **AI Interaction Screen:** Provides summarized content and answers to user queries.
* **Notes and To-Do Screen:** Allows users to manage productivity tasks.

**9.3 Applicable Codes and Standards Used in the Design**

The app adheres to the following standards and best practices:

* **ISO/IEC 25010:** Ensuring software quality and maintainability.
* **IEEE 830:** Standard for software requirements specifications.
* **ISO 27001:** Implementing information security best practices.
* **OAuth 2.0:** For secure authentication and authorization.
* **REST API Standards:** Ensuring seamless communication between the backend and external services.

**9.4 Economic Analysis**

The economic analysis of the App evaluates development, deployment, and maintenance costs. Key factors include:

* **Development Costs:**
  + Flutter and Python development resources.
  + Google Gemini API integration fees.
  + Firebase database and authentication costs.
* **Hosting and Cloud Costs:**
  + Firebase Firestore storage fees.
  + API usage fees based on the number of requests.
* **Maintenance and Updates:**
  + Regular monitoring, bug fixes, and feature updates.
* **Revenue Model:**
  + Potential monetization through premium subscriptions or in-app advertisements.

The analysis concludes that the App is economically viable, especially with scalable Firebase solutions and effective cost management. Further refinements can be applied based on user feedback and market performance.

**Chapter 10: Implementation**

**10.1 Construction**

The construction phase of the App involved assembling the various components and ensuring seamless integration. The primary components constructed include:

* **Frontend:** Developed using Flutter for a cross-platform experience on Android and iOS. Components such as PDF viewers, AI interaction screens, and note-taking interfaces were built using customizable widgets.
* **Backend:** Implemented using Python and Flask to handle API requests, data processing, and interactions with external AI services.
* **Database:** Firebase Firestore was set up for real-time data storage and synchronization.
* **AI Integration:** Integrated Google Gemini API for summarization and chatbot features.

The construction phase followed an agile methodology, allowing iterative testing and development.

**10.2 Programming**

The programming phase consisted of the following key steps:

* **Frontend Development:**
  + Designed with a clean and responsive UI using Flutter's state management techniques.
  + Implemented modular components for easy maintenance and updates.
* **Backend Development:**
  + Developed API endpoints using Flask for communication between the frontend and backend.
  + Applied error handling and logging mechanisms.
* **AI Integration:**
  + Incorporated Google Gemini API for accurate text summarization and chatbot interactions.
  + Managed API requests and responses effectively.
* **Firebase Integration:**
  + Configured Firestore for real-time data management.
  + Implemented Firebase Authentication for secure user access.
* **Testing:**
  + Conducted unit testing for individual components.
  + Performed integration testing to ensure seamless communication between the frontend, backend, and AI APIs.

**10.3 Validation**

The validation phase ensured the application met its functional and non-functional requirements. The key validation activities included:

* **Functional Testing:**
  + Verified core features such as PDF upload, AI summarization, and note-taking.
  + Ensured accurate AI-generated summaries using sample documents.
* **Performance Testing:**
  + Assessed app responsiveness across various devices and network conditions.
  + Validated API response times from Google Gemini.
* **Security Testing:**
  + Confirmed proper authentication and user data protection using Firebase.
* **User Acceptance Testing (UAT):**
  + Conducted testing sessions with users to gather feedback.
  + Addressed usability issues and made iterative improvements.

The validation results confirmed that the app performed as expected, meeting the design specifications and providing a seamless user experience.

**Chapter 11: Results and Discussion**

**11.1 Summary of Goals Met by the Design and Justification for Any Shortcomings**

The App successfully achieved most of its design goals, providing a user-friendly and efficient platform for managing digital documents. Key accomplishments include:

* **AI-Powered Summarization and Chatbot:**
  + Integrated Google Gemini API to provide accurate and concise summaries.
  + The chatbot feature delivers relevant answers to user queries.
* **Document Management:**
  + Users can upload, read, and search PDF files with smooth performance.
  + Searchable PDF functionality allows keyword identification and highlighting.
* **Productivity Features:**
  + Notes and to-do list management with Firebase ensure data persistence and synchronization.
  + Continue Reading and Bookmark features enhance user experience.
* **Cross-Platform Compatibility:**
  + Developed using Flutter, the app provides a seamless experience on both Android and iOS devices.
* **Security and Data Management:**
  + Firebase Authentication ensures secure user login.
  + Firestore Database offers reliable real-time data management.

**Shortcomings and Justifications:**

* **AI Latency:** Some users may experience minor delays in AI-generated summaries during high API traffic. This was mitigated by providing loading indicators and optimizing API request handling.
* **Offline Functionality:** Due to API and Firebase dependencies, the app lacks complete offline support. Future iterations could implement limited offline mode for document viewing and note-taking.
* **Limited Customization:** Users have restricted customization options for AI summaries and text preferences. This was a trade-off to simplify the interface and ensure usability.

**11.2 Summary of Constraints and Codes Met by the Design**

The design adhered to multiple constraints and standards to ensure quality, security, and usability.

**Constraints Met:**

* **Platform Compatibility:** The app was successfully developed for both Android and iOS using Flutter.
* **Performance:** Achieved low-latency AI responses and smooth PDF rendering.
* **Security:** Firebase Authentication ensures user data security.
* **Storage Management:** Implemented efficient storage and management of user data using Firebase Firestore.
* **Network Dependency:** API requests were optimized to minimize network load and improve response time.

**Codes and Standards Met:**

* **ISO/IEC 25010:** Ensured software quality, usability, and maintainability.
* **IEEE 830:** Followed requirements documentation standards.
* **ISO 27001:** Applied security best practices to protect user data.
* **OAuth 2.0:** Used for secure authentication and authorization.
* **REST API Standards:** Ensured consistent communication between the backend and APIs.

The successful implementation of these standards has contributed to the reliability and efficiency of the App, ensuring user satisfaction and compliance with industry best practices.

12. Conclusion

13. Recommendations

List of references

Appendices

Meeting minutes