iPaper Project Proposal

Manoj Bhatt (Student Number: 8306746)

Omkar Shrinivas Gandhi (Student Number: 7570880)

Saugat Timilsina (Student Number: 8209546)

Siddharth B. Singha (Student Number: 7694118)

Surbhi Singh (Student Number: 7355038)

Taha Bombaywala (Student Number: 8387035)

September 26, 2024

Abstract

The iPaper project aims to develop an advanced system designed to assist researchers in efficiently managing and analyzing large collections of research papers. By automating the generation of concise summaries for each uploaded paper, the system provides quick insights into their content. Additionally, iPaper employs topic clustering, allowing users to explore and identify related research more easily within specific fields. This innovation addresses the current challenge faced by researchers, who often manually read and summarize numerous papers, which is both time-consuming and labor-intensive. The system enhances the organization of research materials, facilitates literature reviews, and promotes efficient collaboration. By streamlining research management, iPaper significantly improves productivity and accelerates academic discoveries, offering substantial benefits to researchers, academic professionals, and students alike.

1 Introduction

In today's fast-paced academic and research environments, managing and analyzing large collections of research papers is a challenging and time-consuming task. Researchers often find themselves overwhelmed with the volume of information and struggle to effectively organize, summarize, and extract meaningful insights from vast amounts of literature. The iPaper project addresses these challenges by introducing a smart system designed to automate the summarization and clustering of research papers. This system allows users to upload papers, receive concise summaries, and explore related topics, streamlining the process of literature review and discovery.

1.1 Aims

1.2 Related Work

Several platforms and systems exist to assist researchers in managing and exploring vast collections of academic papers. These systems employ machine learning, natural language processing (NLP), and recommendation algorithms to provide functionalities that align with the goals of the iPaper project. Below, we examine three prominent systems that are closely related to iPaper's objectives, while highlighting how iPaper aims to improve upon these existing approaches.

1.2.1 Review Sematic Scholar

Semantic Scholar, developed by the Allen Institute for AI, is an AI-driven research tool that aids researchers in discovering and summarizing scientific papers. It uses advanced NLP and topic modeling techniques to generate paper summaries and provide recommendations based on thematic similarities. While Semantic Scholar offers robust recommendations, it does not provide users the flexibility to upload their own papers for real-time summarization and analysis. iPaper aims to fill this gap by allowing users to upload research papers, from which it will automatically generate concise summaries and recommend related work, streamlining the literature review process.

1.2.2 Research Gate "Related Search"

ResearchGate is an academic social network where researchers can upload their work and discover related research through its "Related Research" feature. This functionality utilizes machine learning algorithms to cluster papers based on thematic content and offers suggestions for similar papers within a researcher's field of interest. Although ResearchGate offers a degree of interactivity, its recommendation system is based more on user activity (such as publication and interaction history) rather than direct content analysis. In contrast, iPaper emphasizes content-based recommendations. When a user uploads a research paper, iPaper will analyze the document's content—using keyword extraction and clustering methods like LDA—and provide personalized recommendations without relying on user history.

1.2.3 Mendeley Suggest

Mendeley is a popular reference management tool that features Mendeley Suggest, which recommends papers based on a user's library. It applies NLP to understand the thematic content of a user's stored papers and suggests related documents. While Mendeley allows users to manage papers and track references, it does not offer real-time document summarization or personalized recommendations upon upload. iPaper introduces this capability, allowing users to upload research papers and receive automated summaries along with topically related recommendations, ensuring a quick and efficient way for users to digest new information and discover relevant studies.

1.2.4 Gaps Addressed by iPaper

Although systems like Semantic Scholar, ResearchGate, and Mendeley offer valuable tools for literature discovery and organization, they often lack the ability to provide real-time summarization and recommendations based on user-uploaded documents. iPaper addresses these limitations by enabling users to upload their research papers directly, generating concise summaries and offering customized recommendations based on advanced keyword extraction and topic modeling. This functionality enhances the efficiency of literature review processes and helps researchers explore related work more effectively. Additionally, iPaper provides greater flexibility in clustering and analysis, giving users more control over how they explore research fields.

This version highlights the unique upload feature of iPaper, differentiating it from other systems by emphasizing the real-time summarization and recommendation capabilities based on content analysis of user-uploaded research papers.

2 Methodology

The development of the iPaper system follows a structured approach, incorporating various phases to ensure the successful implementation of its key functionalities such as automatic summarization and topic clustering of research papers. The methodology integrates both software development practices and machine learning techniques to build a robust, scalable, and user-friendly platform.

Comprehensive Workflow Overview for iPaper Project:

The iPaper project workflow seamlessly integrates three core components: the User Interface, Machine Learning Models, and Backend Systems, working together to form a highly efficient and scalable platform for analyzing research papers. The User Interface serves as the primary interaction point for users, allowing them to upload and select research papers for analysis in a user-friendly and intuitive manner. Once the paper is selected, the system engages two powerful machine learning models to process the document. The first model, the Data Extraction ML Model**,** extracts the Abstract, Introduction, and Methodology from the research paper and performs clustering. This process ensures that the system accurately separates and categorizes the various sections of the paper. The second model, the Data Analysis ML Model, takes the segmented data and performs in-depth analysis, including training and testing, to derive meaningful insights from the paper.

On the backend, the system handles critical tasks such as file storage, where each research paper is securely stored in a database for future use. The backend also supports efficient search operations, allowing users to quickly retrieve relevant documents for analysis. Additionally, the results of the machine learning analysis are stored for future reference, providing users with a comprehensive and accessible database of analyzed papers. The scalability of the backend system ensures that the platform can handle a growing volume of research papers without compromising performance, making it ideal for academic institutions, researchers, and students dealing with large datasets.

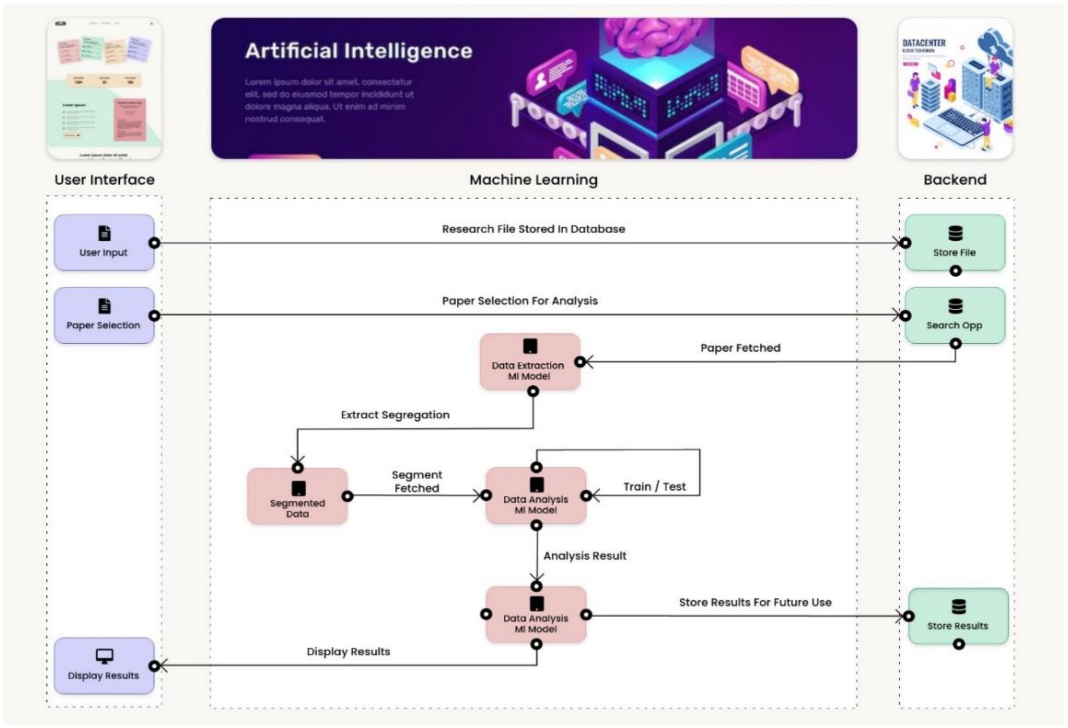


Figure 2: Image showcasing how the application would interact with ML model and get response back to app and store it in database.

The seamless integration of the user interface, machine learning models, and backend systems enables the iPaper platform to deliver accurate, efficient, and scalable research paper analysis. This enhances the user's ability to effectively manage, explore, and gain valuable insights from large collections of academic papers.

2.1 Technologies Used

Front-end:

Technology	**Description**	**Important Feature**
React	JavaScript library for building user interfaces.	Component-based architecture allows for reusable and modular components, improving scalability and maintainability.
Redux (or React Context API)	State management for React applications.	Centralized state management simplifies complex data flows, making the application more predictable and scalable
React Router	Routing library for React applications.	Dynamic routing ensures smooth transitions between different

		views in single-page applications.
Jest	JavaScript testing framework.	Snapshot testing helps ensure UI consistency by comparing rendered components to saved snapshots.
React Testing Library	Library for testing React components.	Testing through user interaction allows you to simulate user behaviour, ensuring components behave as expected.
Tailwind CSS	Utility-first CSS framework.	Responsive design utilities enable quick and efficient styling for different screen sizes, improving the app's adaptability.
Axios	HTTP client for making API requests.	Promise-based HTTP requests allow for easier and cleaner asynchronous API interactions with built-in error handling.
Styled Components	CSS-in-JS library for styling React components.	Scoped styling ensures that styles are applied only to the specific components, avoiding global CSS conflicts.

Back-end:

Technology	**Description**	**Important Feature**
FastAPI	A modern, fast web framework for building APIs in Python. It supports asynchronous programming, making it efficient for handling real-time data and integrations.	Provides high performance for API creation and supports async and await features for handling concurrent tasks.
Pydantic	For data validation and serialization in FastAPI.	Provides built-in validation for API request/response structures, ensuring data consistency.
Requests/HTTP	For API integration, especially for communicating with machine learning services	Simplifies communication with external services (e.g., ML models) through easy-to-use HTTP clients.
Pytest	For running unit and integration tests.	Facilitates comprehensive testing of API endpoints, database operations, and other components. .
Docker	An open-source technology called Docker was created to automate the deployment of programs inside small, portable containers.	Docker provides containerization, allowing applications and their dependencies to run consistently across various environments.
MongoDB	MongoDB is a NoSQL, document-oriented database that stores data in JSON-like documents, offering flexibility in	Provides horizontal scalability, schema-less design, and seamless integration with FastAPI for managing large volumes of dynamic data efficiently.

	handling unstructured or semi-structured data.	

Machine Learning Model:

Technology	**Description**	**Important Feature**
Pymupdf4llm	For extracting text from PDF documents and converting it to markdown format.	Efficient PDF text extraction to markdown conversion.
Regular Expressions (Regex)	To identify and extract specific section like Abstract, Introduction, conclusion, etc.	Powerful pattern matching for section-specific text extraction
Hugging Face Transformers (Pegasus-XSum)	For summarizing the extracted text using transformer models.	High-quality text summarization using transformer-based models.
Python	Primary programming language for text extraction, processing, and summarization.	Flexible and widely used programming language.
NumPy & Pandas	For data manipulation and pre-processing of the extracted text.	Efficient data handling for pre-processing tasks.
scikit-learn (Optional)	For topic modelling and clustering papers based on content similarity.	Optional tool for analysing content similarity among documents.
Latent Dirichlet Allocation (LDA)	For topic clustering and organizing papers into relevant groups.	Effective topic clustering based on probabilistic models.
Jupyter Notebooks	For documentation, experimentation, and code testing.	Interactive environment for code testing and documenting experiments.

Version Control:

Technology	**Description**	**Important Feature**
GitHub	A cloud-based platform for version control and collaboration, enabling teams to manage code changes.	Supports distributed version control, branch management, and collaboration on code.
OneDrive	A cloud storage service that allows file versioning, sharing, and collaboration.	Provides easy file sharing, automatic versioning, and seamless integration with Microsoft Office tools.

Project Management:

Technology	Description	Important Feature
Project Libre	An open-source project management software for scheduling, resource allocation, and task tracking.	Provides Gantt Chart Visualization, task scheduling and resource management
Jira	A popular project management tool for tracking issues, managing tasks, and facilitating agile workflows.	Supports agile methodologies, customizable workflows, and efficient task tracking.

2.2 Project Challenges

User Interface Challenges:

In UI design, one of the primary technical challenges is ensuring consistency across platforms while maintaining visual appeal. The goal is to create an interface that feels intuitive and responsive for different devices, whether it's a desktop, tablet, or mobile phone. Achieving this level of adaptability requires the use of responsive design techniques, media queries, and flexible layouts. At the same time, the aesthetic choices must align with usability principles, ensuring that users can easily understand and interact with the interface without cognitive overload.

Component	Description	Approach
Cross-platform Compatibility	Designing an interface that works seamlessly across multiple devices.	Use responsive frameworks like Bootstrap or Flexbox, optimize for touch.
Aesthetic vs Usability	Balancing visual appeal with ease of use for the end-user.	Conduct regular usability tests to identify potential areas of confusion.
Accessibility	Making the interface accessible to all, including people with disabilities.	Follow WCAG guidelines, ensure keyboard navigation, and use alt texts.

The key challenge is creating a design that is both functional and accessible to all users while ensuring that it retains a consistent look and feel across different screen sizes and devices.

User Experience Challenges:

User experience (UX) challenges are often deeper and less visible than UI issues but are equally significant. The focus of UX is ensuring that users can navigate through the interface with ease and achieve their objectives without unnecessary friction. One of the biggest hurdles in UX design is minimizing cognitive load, meaning that users should not feel overwhelmed or confused by the interface. Additionally, UX designers must ensure that the design can scale to accommodate future changes or new features without becoming cluttered or confusing.

Component	Description	Approach

Cognitive Load	Reducing the mental effort required to use the application.	Use minimalistic designs, group similar actions together, simplify paths.
User Journey Optimization	Ensuring that users can achieve their goals without unnecessary steps.	Map out user journeys and conduct usability testing to identify friction.
Scalability	Designing a flexible UX that can evolve with future updates or features.	Adopt a modular design, focus on scalable navigation and layouts.

Minimizing user frustration through thoughtful UX design is essential. This involves creating streamlined user flows, making frequent iterations based on user feedback, and anticipating future updates to the application.

Coding Challenges:

While UI/UX design focuses on the user's interaction with the interface, the back-end code that brings the design to life presents its own challenges. One significant issue is ensuring that complex features such as animations, transitions, or dynamic content don't hinder performance. As designers push the limits with interactive designs, developers must ensure that the underlying code can handle these demands efficiently, especially across different browsers and devices.

Component	**Description**	**Approach**
Code Implementation	Translating complex design features into efficient, maintainable code.	Use optimized CSS/JS libraries, implement reusable components.
Cross-browser Compatibility	Ensuring the design functions properly across various web browsers.	Test on multiple browsers, apply polyfills or vendor-specific code where needed.
Performance Optimization	Maintaining a fast and responsive user experience, particularly for mobile.	Optimize images, reduce HTTP requests, and use lazy loading techniques.

Developers must often strike a balance between the creative aspirations of the design and the practical limitations of what can be implemented efficiently through code. Additionally, keeping the site responsive and performant is critical for a seamless user experience.

Front-end Integration Challenges:

Front-end integration is the process of bringing a design to life by converting static wireframes and mockups into an interactive interface. One of the primary challenges in this area is ensuring that the design remains true to its original vision while interacting with dynamic data from back-end systems. Front-end developers often encounter issues when integrating third-party APIs, managing state, or rendering real-time data without compromising the visual consistency or performance of the site. Additionally, ensuring that the design works across various devices and screen sizes without breaking the user experience requires thorough testing and a solid responsive design strategy.

Component	**Description**	**Approach**

Data Integration	Ensuring that dynamic content from APIs or back-end systems does not disrupt the UI design.	Plan integration early in the design process, use state management solutions like Redux or Vuex.
Responsive Design Testing	Ensuring that the interface remains consistent and functional across devices and screen sizes.	Implement mobile-first design principles, use CSS media queries, and conduct device testing.
Front-end and Back-end Sync	Aligning the front-end design with back-end functionalities without losing performance.	Collaborate closely with back-end teams, mock API data for testing, and use tools like Swagger for API documentation.

A key challenge in front-end integration is maintaining a balance between design and dynamic content. Front-end developers must work closely with both designers and back-end developers to ensure that the user interface looks and functions as intended, even when integrated with complex systems that may introduce changes in data or structure.

Front-end Integration Challenges:

Incorporating unit testing into front-end development presents its own set of challenges. These challenges can include writing meaningful test cases for dynamic UI elements, ensuring that the tests are comprehensive, and managing dependencies without over-complicating the test setup. Additionally, balancing the time spent on writing tests with the actual development work can be a significant challenge, especially under tight deadlines.

Component	**Description**	**Approach**
Testing UI Components	Ensuring individual UI components render and behave as expected.	Use tools like Jest, Mocha, or Enzyme to test isolated UI components.
Testing State Management	Verifying that the state is managed correctly within the application.	Use tools like Redux Testing Library to ensure proper state management.
Mocking API Calls	Simulating API responses for testing UI elements without using live data.	Use libraries like Axios-mock or Sinon for API mocking during tests.
Managing Test Dependencies	Keeping tests isolated from external dependencies to avoid test failures.	Use mock data and services to isolate components from third-party APIs.

By integrating unit testing into the front-end development process, we will improve the quality and reliability of the project while reducing time spent on debugging later in the development lifecycle.

Machine Learning Challenges:

Implementing machine learning models presents several challenges that impact the efficiency and accuracy of the system. These challenges include handling the complex structure of research papers, ensuring effective data extraction from unstructured content, and managing the computational resources required for real-time processing, summarization, and clustering. Addressing these issues is critical to delivering reliable and scalable machine learning solutions within the platform.

Component	**Description**	**Approach**
Formatting and Structure of PDF Documents	The complex structure of PDFs, with lines, columns, Images and non-linear layouts, makes it difficult to extract clean, structured text. PDFs often lack clear section markers, complicating the extraction of specific sections which contain images, tables, etc.	For the purposes of analysis, summarization, and clustering, we focus exclusively on the Abstract, Introduction, and Conclusion sections of the research papers.
Processing Large Amounts of Text and Lack of Model Training Resources	Handling large datasets or long documents requires significant computational resources. Text summarization models like transformers are resource intensive and can become a bottleneck without sufficient computational infrastructure.	Optimize model processing through task batching, memory management, and the use of scalable cloud computing resources to efficiently process large datasets.
Text Extraction: Full Text vs. Abstracts	Choosing between extracting full text or abstracts introduces trade-offs. Full text provides more information but risks noise from irrelevant sections, while abstracts are concise but may lose important context, impacting model accuracy.	Balance extraction between full text and abstracts by testing both approaches. Implement noise filtering for full text or detailed analysis for abstracts to ensure the best balance between data richness and system performance.

To address the machine learning challenges in iPaper, we employ advanced pre-processing techniques to handle the unstructured format of research papers and ensure accurate data extraction. Additionally, we optimize the machine learning models by utilizing asynchronous processing and scalable cloud resources to manage computational demands, ensuring efficient real-time analysis, summarization, and clustering

Backend Challenges:

The iPaper project include optimizing real-time API performance using asynchronous operations, ensuring efficient data validation with Pydantic, and managing resilient API integrations with external services. Additionally, consistent application deployment and scalability are maintained using Docker and container orchestration, while MongoDB handles large datasets efficiently through indexing and caching, ensuring scalability with MongoDB Atlas

Component	**Description**	**Approach**
Optimizing Real-Time API Performance and Asynchronous Task Management	Handling high traffic loads while managing real-time API responses with asynchronous operations.	Implement async functionality and optimize with middleware and caching strategies.
Efficient Validation and Data Handling with Pydantic:	Validating large datasets and unstructured data efficiently without compromising performance.	Use lazy validation and bulk operations to minimize bottlenecks during validation.

API integration and external service communication:	Handling API communication with external services (e.g., ML models) under heavy load.	Implement connection pooling, retries, and error handling for resilient API communication.
Ensuring Scalability and Consistency with Docker	Ensuring consistent application deployment and scaling across multiple environments.	Utilize container orchestration (e.g., Docker, Kubernetes) and CI/CD pipelines for consistent deployments.
Managing large Datasets with MongoDB	Storing, retrieving, and managing large volumes of unstructured data in a scalable manner.	Optimize data with indexing, and caching. Ensure scalability with MongoDB Atlas.

To address backend challenges in the iPaper project, FastAPI utilizes asynchronous functionality, middleware, and caching to optimize real-time performance under high traffic. Pydantic ensures efficient data validation with lazy validation and bulk operations, while Docker, Kubernetes, and CI/CD pipelines ensure consistent deployment and scaling. MongoDB is optimized with indexing and caching, ensuring efficient data management and scalability through MongoDB Atlas.

2.3 Project Team

The iPaper project is driven by a team of talented professionals, each bringing a unique set of skills and expertise to create an innovative solution for managing and analyzing research papers. The team consists of experts in machine learning, UI/UX, backend, software development, and project management, ensuring a comprehensive approach to delivering a robust system. This project requires every group member to acquire and apply new skills while also utilizing their existing ones.

Manoj Bhatt

Manoj Bhatt serves as the Machine Learning Engineer and Documentation Assistant for the project. With over 5 years of experience as a software engineer, his expertise primarily lies in backend development, working extensively with technologies such as Django, FastAPI, and databases like PostGIS and MongoDB. He has also gained hands-on experience with CI/CD tools like Jenkins and Docker. In addition, Manoj has worked on data intelligence and visualization projects using Python making him a very proficient in Python. As AI continues to evolve, Manoj has shifted his focus to machine learning, acquiring the skills necessary to excel as a machine learning engineer. Through various projects in his Master's program in Computer Science, he has developed proficiency in building machine learning models using techniques such as supervised learning and deep learning, as well as natural language processing (NLP) methods like transformer models and topic modeling for text abstraction. In this project, Manoj is responsible for managing the entire machine learning pipeline, which includes text extraction from research papers, data collection and formatting, and model development and fine-tuning for optimal performance. He will also assist with project documentation whenever required.

Omkar Shrinivas Gandhi

Omkar is a Front-end Lead of the iPaper project, primarily focused on front-end development for the Web application. He has strong expertise in React and Redux for building scalable, modular user interfaces. His skills in React Router enable smooth navigation within single-page applications, while his experience with Jest and React Testing Library ensures UI consistency and accurate component behavior through testing. Additionally, Omkar's proficiency in Tailwind CSS and Styled Components ensures responsive design and scoped styling, preventing global CSS conflicts. His ability to handle Axios for API requests ensures reliable communication between the frontend and backend, making him the perfect candidate to lead the frontend development. In addition to leading the front-end development, Omkar plays a crucial role in the final proofreading and editing of the project's documentation. His attention to detail ensures that all written materials are clear, concise, and free of errors. Omkar's involvement in the documentation process guarantees that the technical aspects are accurately communicated, while also enhancing the overall readability and professionalism of the project's reports and deliverables.

Taha Bombaywala

Taha Bombaywala is the Frontend Developer for the iPaper project which focuses on consumer website designing and frontend Development. With a strong understanding of React Fluent in Js, Taha specializes in dynamic and responsive web application development aiming to provide users with an effortless experience. His skills in React and other modern front-end technologies guarantee the user interface for our project is intuitive and delightful. Taha incorporates the use of React Router for managing navigation around the application, providing seamless interactions and a quality experience as users move in and out of different features or views. In addition to being a UI developer, he is capable of making API calls to the server using Axios for Frontend-backend interaction. He also uses MongoDB and RESTful APIs for real-time data fetching/POSTing, updating the frontend UI with the most current database information, which allows Taha to deliver transactional applications to the data-driven enterprises involving very high performance and reliability. Taha's commitment to clean, well-tested code and eye for detail have made him highly valuable at iPaper, where his contributions led to a robust, responsive web app that caters to the often-frustrated user base of research professionals, all while learning new ways of doing things and keeping the project on the cutting edge of modern web development.

Saugat Timilsina

Saugat serves as the ML Developer and Tech Lead for the iPaper project, leveraging his extensive background in software engineering, machine learning, and mobile app development. Currently pursuing a master's degree in computer science with a specialization in Software Engineering, Big Data, and Machine Learning, Saugat brings nearly five years of experience as a Full Stack and Mobile Application Developer. He has expertise in platforms like Android, iOS, Flutter, and frameworks such as Django, FastAPI, and Flask for building RESTful APIs. With growing proficiency in machine learning frameworks like Scikit-Learn, PyTorch, and TensorFlow, Saugat is well-equipped to lead the team in implementing ML models and managing data workflows. As Tech Lead, he oversees the technical direction of the project, addressing challenges, ensuring smooth deployment processes, and integrating machine learning solutions effectively.

Siddharth B. Singha

Siddharth Singha is a key contributor to the iPaper project, specializing in back-end development with a focus on creating scalable and reliable APIs using technologies like Flask, FastAPI, and Python. His expertise ensures the platform can efficiently handle high traffic and maintain seamless communication between the front-end, machine learning models, and databases. In addition to his back-end role, Siddharth demonstrates versatility by contributing to front-end development and machine learning integration, ensuring that all components of the system work cohesively. His collaborative approach and problem-solving skills help maintain project momentum, aligning with goals and facilitating smooth progress. Siddharth's dedication to building efficient solutions and driving innovation makes him an invaluable asset to the iPaper team, contributing significantly to the project's immediate and long-term success. His adaptability across various domains reflects his strong commitment to the project's vision.

Surbhi Singh

Surbhi plays a key role in the iPaper project, serving as Project Manager, Backend Engineer, and Documentation Lead. As Project Manager, she ensures seamless coordination between the front-end, back-end, and machine learning teams, maintaining project alignment with goals in an agile environment. As Backend Engineer, Surbhi utilizes a modern technology stack, including FastAPI for building high-performance APIs, Pydantic for data validation, and MongoDB for scalable data management. She also implements Requests/HTTP for integrating machine learning models and Pytest for ensuring system reliability. In her role as Documentation Lead, Surbhi plays a vital role in ensuring the highest quality of documentation for the iPaper project. Her strong emphasis on documentation quality guarantees that the iPaper project's technical aspects are communicated effectively and comprehensibly. By balancing her technical expertise with her commitment to clear and comprehensive documentation, Surbhi ensures that the project not only functions smoothly but is also well-documented for future reference and scalability. With over three years of experience as a business analyst and full-stack developer in .Net framework, she combines analytical and technical skills to provide valuable insights into both the functional and technical aspects of the project.

2.3.1 Roles and Responsibilities

In this section we will explain the roles of the project, the attached responsibilities, and who will be stepping up for each of these positions.

Project Manager (Surbhi Singh):

- Assigns roles and tasks to group members.
- Decides the final timeline of the project.
- Responsible for finalizing the project plan.
- Ensures all group members are performing to a high standard.
- Coordinates meetings and collaboration between group members.

UI/UX Design Team (Omkar (Lead)):

- Design the prototypes for each of the pages and the features expected within iPaper.
- Visualize the user journey through the application using these prototypes.
- Enhance the prototypes to their full potential and communicate them to the front and back-end teams.

Front-End Development Team (Omkar (Lead), Taha):

- Develop the designs provided by the UI/UX team.
- Develop the functions of each design as explained in the user journeys.
- Coordinate with the Back-End development team to connect APIs and microservices where needed
- View the developed features in web to ensure functionality

Back-End Development Team (Siddharth (Lead), Surbhi Singh):

- Develop the microservices for the application.
- Develop required databases for the microservices.
- Coordinate with the Front-End development team to connect APIs and microservices where needed.

Machine Learning Team (Saugat (Lead), Manoj):

-

Documentation Lead (Surbhi Singh):

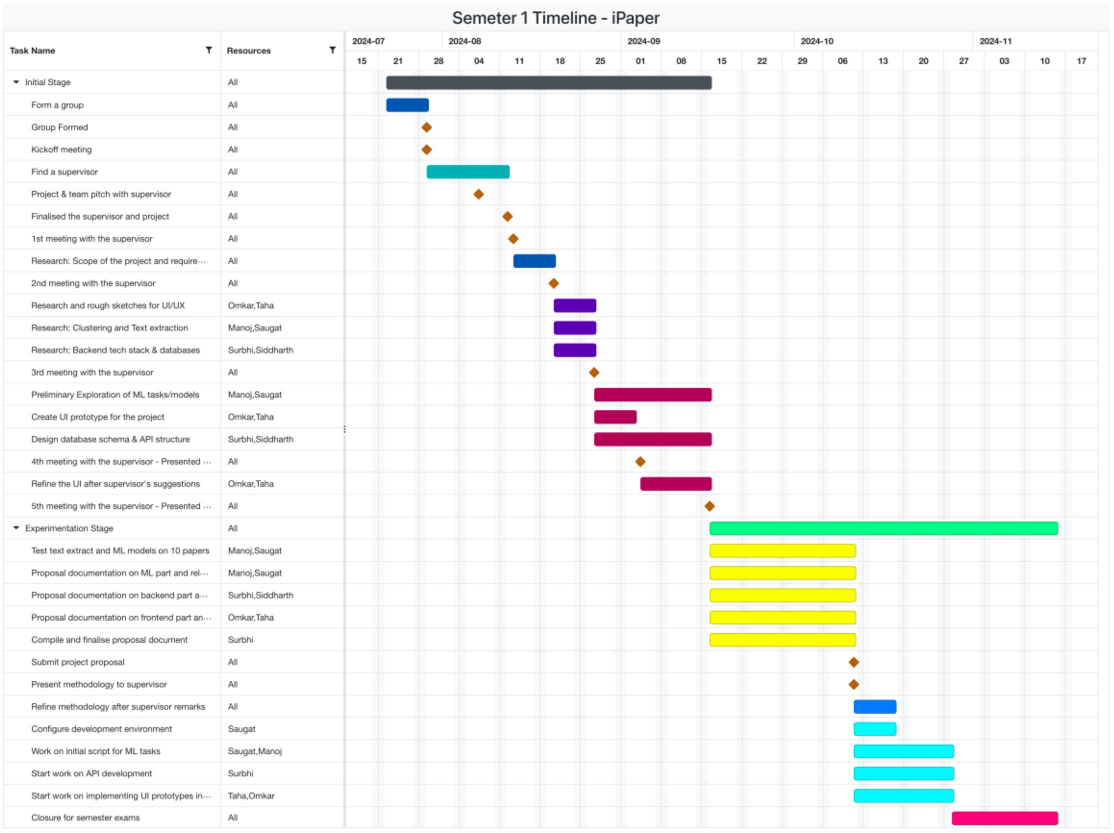
- Determines the documentation necessary for the success of the project
- Aids the project manager in determining the roles for documentation and in project planning

- Compiles the necessary documents
- Documents Meeting Notes

2.4 Plan and Timeline

2.4.1 Gantt Chart

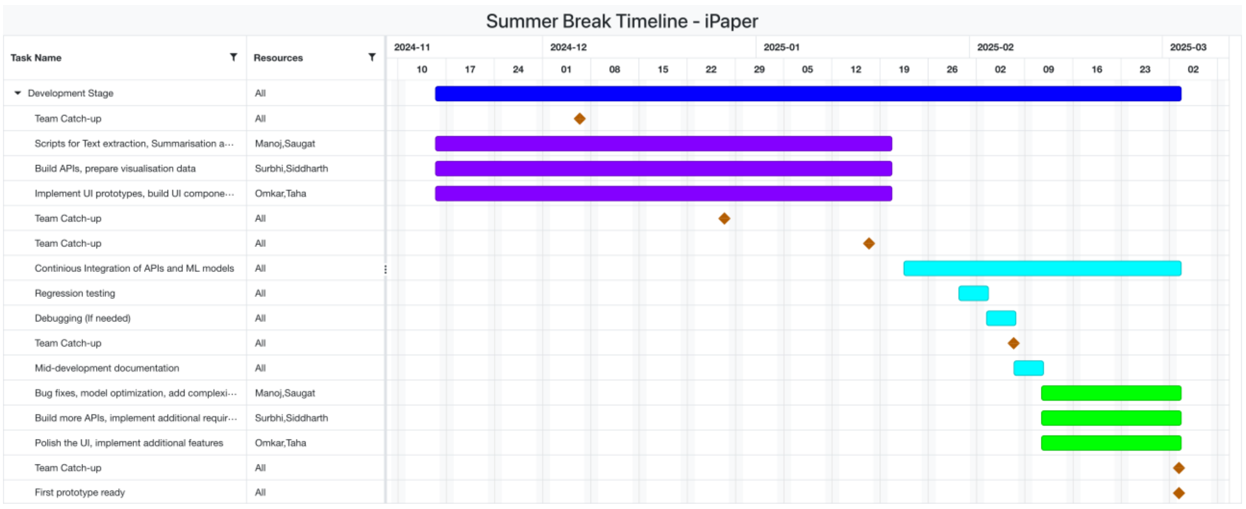
1. Semester 1 Gantt Chart (July 2024 - November 2024)
The first semester focuses on project initiation and experimentation. It includes the group formation, supervisor meetings, and research on the project's scope. Tasks such as creating the project proposal, designing UI prototypes, and conducting preliminary machine learning explorations are key components. The experimentation stage involves testing text extraction and ML models on sample papers, along with the preparation and submission of the project proposal by mid-October. It ends with exams for spring semester and work only resumes during summer break.



Task Name	Resources
Initial Stage	All
Form a group	All
Group Formed	All
Kickoff meeting	All
Find a supervisor	All
Project & team pitch with supervisor	All
Finalised the supervisor and project	All
1st meeting with the supervisor	All
Research: Scope of the project and require…	All
2nd meeting with the supervisor	All
Research and rough sketches for UI/UX	Omkar,Taha
Research: Clustering and Text extraction	Manoj,Saugat
Research: Backend tech stack & databases	Surbhi,Siddharth
3rd meeting with the supervisor	All
Preliminary Exploration of ML tasks/models	Manoj,Saugat
Create UI prototype for the project	Omkar,Taha
Design database schema & API structure	Surbhi,Siddharth
4th meeting with the supervisor - Presented …	All
Refine the UI after supervisor's suggestions	Omkar,Taha
5th meeting with the supervisor - Presented …	All
Experimentation Stage	All
Test text extract and ML models on 10 papers	Manoj,Saugat
Proposal documentation on ML part and rel…	Manoj,Saugat
Proposal documentation on backend part a…	Surbhi,Siddharth
Proposal documentation on frontend part an…	Omkar,Taha
Compile and finalise proposal document	Surbhi
Submit project proposal	All
Present methodology to supervisor	All
Refine methodology after supervisor remarks	All
Configure development environment	Saugat
Work on initial script for ML tasks	Saugat,Manoj
Start work on API development	Surbhi
Start work on implementing UI prototypes in…	Taha,Omkar
Closure for semester exams	All

Fig 1: Semester 1 Gantt Chart

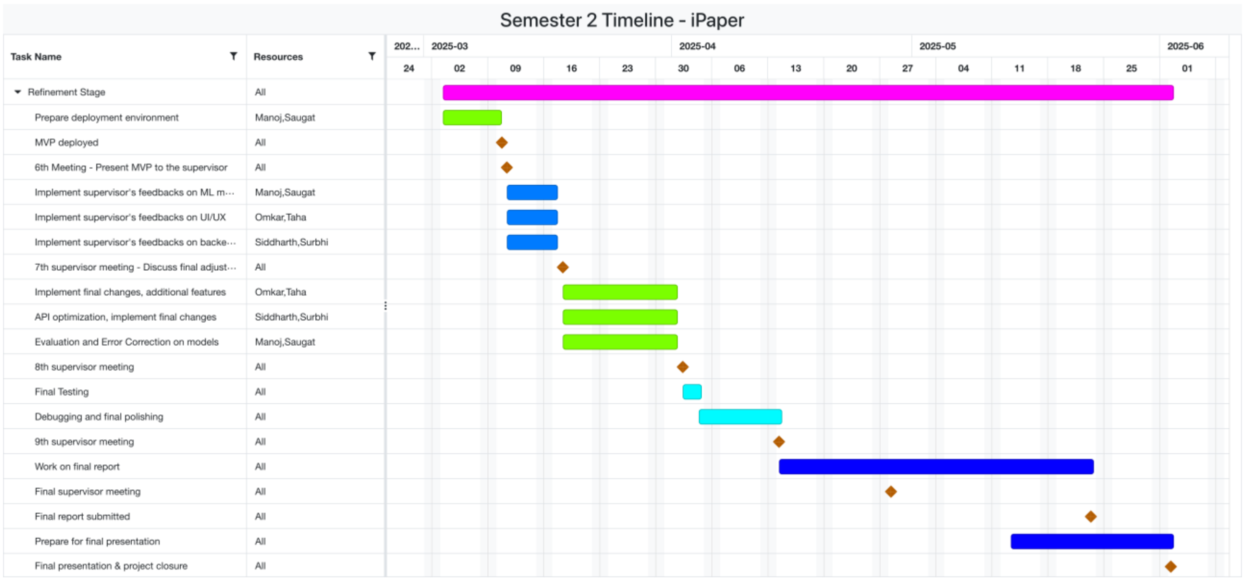
2. Summer Break Gantt Chart (November 2024 - March 2025):
The summer break is primarily dedicated to development. The team focuses on implementing the full machine learning pipeline, including text extraction, data collection, model building, and API integration. Concurrently, the backend and frontend teams work on implementing the backend architecture and UI features. Regular reviews and refinements of the models and code are scheduled to ensure smooth progress when transitioning into the next semester.



Task Name	Resources
Development Stage	All
Team Catch-up	All
Scripts for Text extraction, Summarisation a···	Manoj,Saugat
Build APIs, prepare visualisation data	Surbhi,Siddharth
Implement UI prototypes, build UI compone···	Omkar,Taha
Team Catch-up	All
Team Catch-up	All
Continious Integration of APIs and ML models	All
Regression testing	All
Debugging (If needed)	All
Team Catch-up	All
Mid-development documentation	All
Bug fixes, model optimization, add complexi···	Manoj,Saugat
Build more APIs, implement additional requir···	Surbhi,Siddharth
Polish the UI, implement additional features	Omkar,Taha
Team Catch-up	All
First prototype ready	All

Fig 2: Summer break Gantt Chart

3. Semester 2 Gantt Chart (March 2025 - June 2025:
In the second semester, the refinement stage begins. The team works on improving the machine learning models' accuracy, optimizing the codebase, and integrating all project components. Final touches are added to the frontend, backend, and machine learning systems. This phase also focuses on testing, bug fixing, and preparing the final report and presentation. The project concludes with the final report submission and a presentation at the end of May 2025.



Task Name	Resources
Refinement Stage	All
Prepare deployment environment	Manoj,Saugat
MVP deployed	All
6th Meeting - Present MVP to the supervisor	All
Implement supervisor's feedbacks on ML m···	Manoj,Saugat
Implement supervisor's feedbacks on UI/UX	Omkar,Taha
Implement supervisor's feedbacks on backe···	Siddharth,Surbhi
7th supervisor meeting - Discuss final adjust···	All
Implement final changes, additional features	Omkar,Taha
API optimization, implement final changes	Siddharth,Surbhi
Evaluation and Error Correction on models	Manoj,Saugat
8th supervisor meeting	All
Final Testing	All
Debugging and final polishing	All
9th supervisor meeting	All
Work on final report	All
Final supervisor meeting	All
Final report submitted	All
Prepare for final presentation	All
Final presentation & project closure	All

Fig 3: Gantt chart for semester 2

The iPaper project is scheduled to commence on October 11, 2024, and conclude by May 8, 2025. The project is divided into key areas: UI/UX, front-end, machine learning, back-end, and documentation. These divisions allow for parallel progress across different components, which will be clearly illustrated in the accompanying Gantt charts. Each segment is broken down into tasks and sub-tasks, with specific assignments made to the respective project members, to be completed according to the timelines outlined in the Gantt charts.

Effective collaboration tools are essential to the success of this project. We plan to utilize Zoom for voice and video conferencing due to its high-quality screen sharing capabilities, Jira for project management, task tracking, and assigning tasks during sprints, and OneDrive for document management. WhatsApp will be used for general messaging when necessary, and OneDrive will also serve as a platform for documenting project status and progress to the subject coordinators. Finally, Overleaf/Latex will be used for creating essential documents, such as the project proposal. These tools will ensure smooth communication, efficient task management, and thorough documentation throughout the project.

2.4.1 In Scope:

Table 1: This table details the implementation items and deliverables that are within the scope for our project and describes these items.

2.4.2 Out of Scope:

Table 2: This table details the items and deliverables that are out of scope for our project and describes these items.

3 Outcomes

3.1 Product Overview

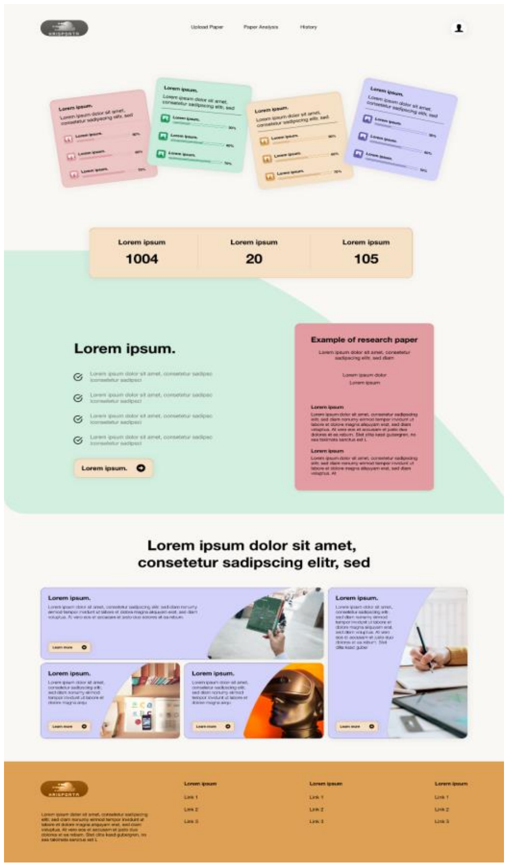
A. Home Page

- 1. Feature Display Cards: The homepage highlights the key features of the application using colorful cards that provide a quick visual overview, helping users understand the app's main functionalities at a glance.

- 2. Key Metrics Section: This section presents important statistics such as the number of papers uploaded, analyzed, and pending, giving users a clear picture of the app's current activity and usage.

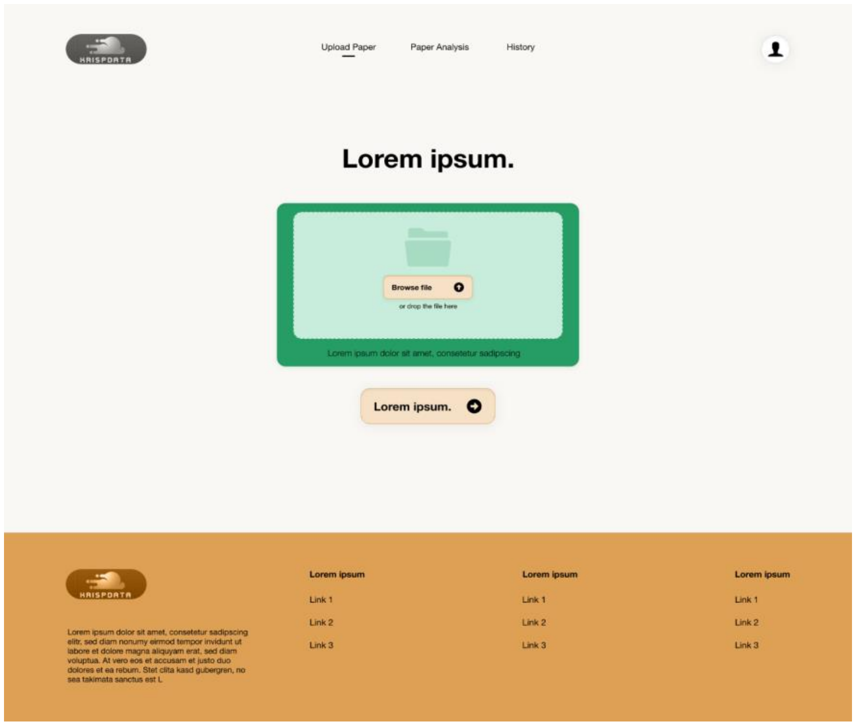
- 3. Research Paper Example: A dedicated section offers an example research paper layout, serving as a reference or guide for users to understand the structure and format expected during paper submission.

- 4. Resource Links: The page includes multiple resources with "Learn more" buttons, guiding users to additional support or information related to paper analysis, usage instructions, or feature explanations.



B. Upload Paper Page

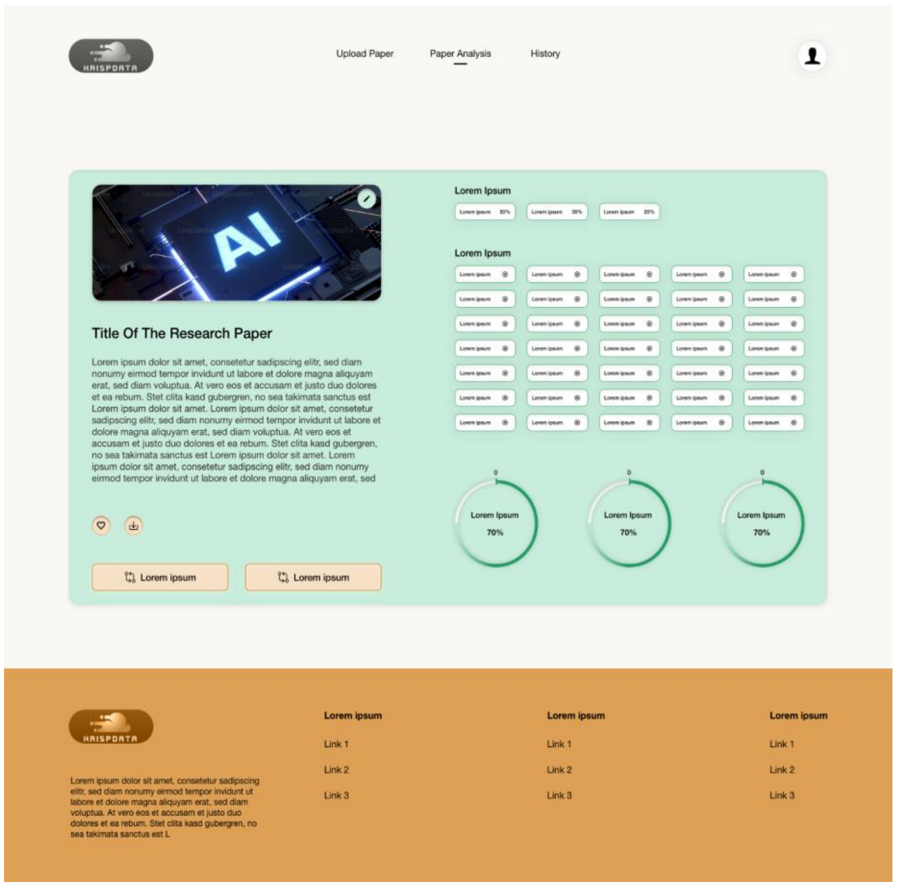
- 1. File Upload Box: The page features a central file upload box where users can either drag and drop files or click "Browse File" to select a file from their device.
- 2. User-Friendly Interface: The page has a clean and intuitive design, ensuring that users can easily understand how to upload their files with minimal effort.
- 3. Visual Feedback: The file upload box provides a clear visual cue, using a folder icon and descriptive text to guide the user through the upload process.
- 4. Primary Action Button: A large button below the upload box directs the user to continue with the file submission, reinforcing the simplicity of the upload process.



Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est L

C. Paper Analysis Page

- 1. Analysis Results: The page displays the results of the paper analysis, including extracted keywords, a summary of the paper, and statistical data, offering a comprehensive overview of the research.
- 2. Downloadable Results: Users can download the results of the analysis, allowing them to keep a record or further process the data offline.
- 3. Favorites Feature: The page includes a feature that allows users to mark the paper as a favorite, making it easier to access frequently analyzed or important papers.
- 4. Interactive Stats: The page presents visually appealing statistics (like circular progress charts) to show the percentage of various analysis metrics, offering a clear, easy-to-understand display of the paper's performance or relevance.



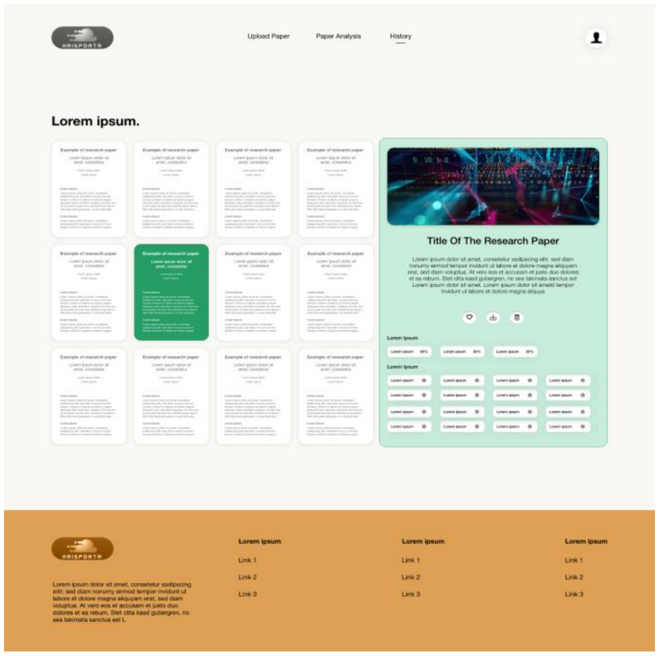
Title Of The Research Paper

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est L

D. Related Paper Page

- Here are four key points updated to reflect the selected paper being from the paper analysis page:
- 1. Related Paper List: The page displays a list of research papers related to the one currently being analyzed on the paper analysis page, helping users discover similar studies or works.
- 2. Paper Overview: Each related paper includes a brief description or keywords, allowing users to quickly assess its relevance without opening each paper individually.
- 3. Seamless Navigation: Users can effortlessly switch between the selected paper and its related papers, maintaining focus on their current research while exploring other relevant documents.
- 4. Selected Paper Details: The right side of the page continues to display the detailed view of the paper being analyzed, providing a constant reference while users explore related materials. This ensures that the user stays connected with their main research focus.



Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit ametd tempor invidunt ut labore et dolore magna aliquya.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est L.

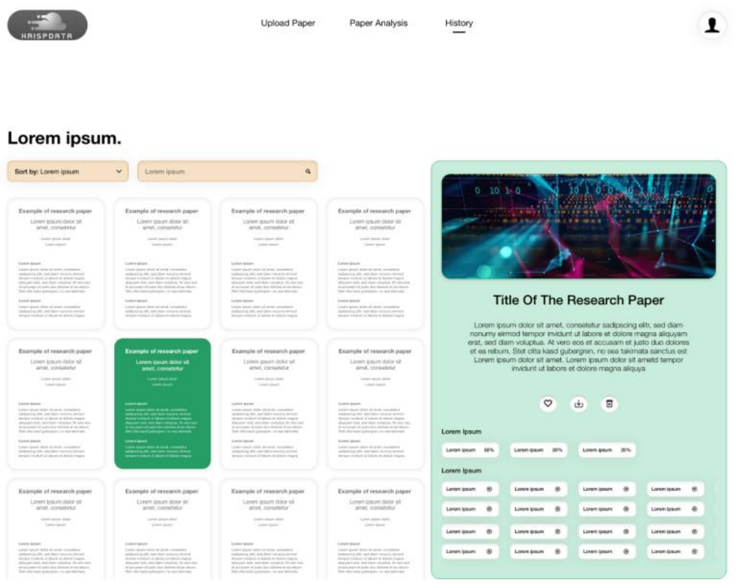
E. History Page

1. Paper List Display: The page shows a list of all the research papers available, allowing users to browse through them and select specific papers for analysis.

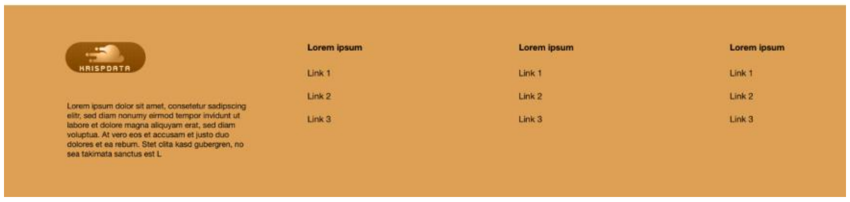
2. Filtering Options: A filter or sorting feature is available, enabling users to sort papers by specific criteria, making it easier to find relevant or previously analyzed papers.

3. Paper Selection for Analysis: Users can click on a paper from the list to view more detailed information and initiate a fresh analysis if needed.

4. Detailed Paper View: Once a paper is selected, a detailed view on the right side of the page displays key information, including the title, abstract, and other analysis options.



Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit ametd tempor invidunt ut labore et dolore magna aliquya



Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est L

3.2 Key Features

The goal of iPaper is to improve productivity for researchers and academics by automating the time-consuming tasks of summarizing, organizing, and retrieving research papers, making the research process more efficient and effective.

- **Automated Summarization:** The machine learning models will be used to generate concise summaries of uploaded research papers, allowing users to quickly grasp the key points of a document.
- **Topic Clustering:** The system clusters research papers based on their topics, helping users explore related works and organize their literature collections according to thematic areas.
- **Data Extraction:** iPaper efficiently extracts core sections (Abstract, Introduction, Methodology, etc.) from research papers, saving user's time on manual reading and organization.
- **Search and Retrieval:** Users can search through their stored research papers, and iPaper retrieves relevant documents based on keywords, topics, or specific metadata, offering a powerful search experience.
- **Scalable Data Management:** Built on a robust backend powered by MongoDB, iPaper can handle large datasets efficiently, ensuring that even extensive collections of research papers are organized and retrieved quickly.
- **User-Friendly Interface:** The platform is designed with an intuitive UI, making

it easy for users to upload papers, view results, and manage their collections effortlessly.

3.3 How It Works

The iPaper application allows users to upload research papers, which are then processed through machine learning models for data extraction, summarization, and topic clustering. It provides users with efficient tools for searching, retrieving, and organizing their research collections based on key sections and topics.

- **Upload Paper:** Users can upload research papers via the platform's user-friendly interface. The system processes the paper and prepares it for analysis.
- **Data Extraction:** Using machine learning models, the system extracts essential sections like the Abstract, Introduction, and Methodology from the research paper.
- **Summarization:** The system automatically generates concise summaries of the uploaded research papers, helping users quickly understand the key points of each document.
- **Topic Clustering:** iPaper groups related research papers based on their topics, allowing users to explore and organize works thematically.
- **Search and Retrieval:** The platform enables users to search and retrieve relevant research papers using keywords, topics, or metadata, offering an efficient way to manage extensive research collections.

3.4 Benefits

The iPaper project offers numerous benefits to users, including time-saving automated features, enhanced research organization, and powerful search capabilities, all aimed at improving the overall research and data management experience.

- **Time Efficiency:** iPaper automatically extracts key sections like the Abstract, Introduction, and Methodology, saving user's time from manually skimming through lengthy research papers.
- **Automated Summarization:** Users can quickly understand the main points of a paper through concise, machine-generated summaries.
- **Improved Organization:** The platform clusters papers by topics, helping users manage and categorize large research collections efficiently.
- **Enhanced Search Capabilities:** With powerful search functionality, users can retrieve relevant research papers based on keywords, topics, or metadata.
- **Seamless Data Management:** iPaper provides an intuitive interface for storing, accessing, and organizing research papers, improving overall research workflow.
- **Accessibility:** The platform allows users to access their research papers anytime, ensuring they have critical information at their fingertips when needed.

3.5 Expected Outcomes

The iPaper project is anticipated to achieve significant outcomes, including precise summarization, effective data extraction, topic-based clustering, and an intuitive user interface, all aimed at improving the research and data management experience for users.

- **Accurate Summarization:** The system will generate precise summaries of research papers, allowing users to quickly grasp the key insights of each document.
- **Efficient Data Extraction:** Key sections such as the Abstract, Introduction, and Conclusion will be extracted automatically from research papers.
- **Topic Clustering:** Research papers will be grouped based on similar topics, facilitating better organization and exploration of related content.
- **Improved Research Workflow:** Users will experience a more streamlined research process with faster access to relevant papers, enhanced search functions, and seamless data management.
- **Scalable Data Management:** iPaper will be able to handle large volumes of research data, ensuring scalability as the user's collection grows.
- **User-Friendly Interface:** The platform will offer an intuitive interface, making it easy for users to upload, analyse, and manage research papers.