International Journal of Science and Research Archive
eISSN: 2582-8185
Cross Ref DOI: 10.30574/ijsra
Journal homepage: https://ijsra.net/

(RESEARCH ARTICLE)

Check for updates

SPARK: A Smart System for Public Aid, Recommendation, and Knowledge Management

Gayatri B. Dhumal *, Sumit A. Hirve, Atharva M. Gaikwad, Rasika P. Mahakalkar, Aditya M. Midgule and Nilesh Thorat

Department of Computer Science, MIT School of Computing, MIT-ADT University, Loni Kalbhor, Pune, Maharashtra, India

International Journal of Science and Research Archive, 2025, 17(02), 591–599

Publication history: Received on 03 October 2025; revised on 13 November 2025; accepted on 15 November 2025

Article DOI: https://doi.org/10.30574/ijsra.2025.17.2.3019

Abstract

The increasing number of government welfare schemes in India has created a challenge for citizens to access accurate information and identify eligibility criteria. Many people remain unaware of the benefits they are entitled to, leading to underutilization of public welfare resources. To address this problem, we propose SPARK (System for Public Aid, Recommendation, and Knowledge), a web-based platform designed to provide citizens with easy access to government schemes and personalized recommendations.

By allowing users to input basic demographic and socio-economic details, SPARK automatically evaluates their eligibility and recommends relevant schemes. The platform employs data filtering, rule-based eligibility checks, and recommendation logic to ensure accuracy and relevance. The system ensures reliability through rule-based eligibility checks and helps users navigate government welfare opportunities more efficiently.

Keywords*:* Public Aid; Recommendation System; Knowledge Management; Data Filtering; Eligibility Verification; Government Schemes.

1. Introduction

Government schemes play a vital role in improving the social and economic conditions of citizens by providing financial aid, subsidies, and welfare opportunities. However, due to the large number of schemes introduced at central, state, and local levels, it becomes difficult for common people to identify which schemes they are eligible for. The lack of awareness, scattered information, and complex eligibility criteria often result in the underutilization of these welfare benefits. A centralized and intelligent system is therefore necessary to bridge the gap between the government and the public.

The proposed project, SPARK (System for Public Aid, Recommendation, and Knowledge), addresses this challenge by providing a digital platform that simplifies the process of discovering and applying for government schemes. By collecting basic user details such as age, gender, income, occupation, and social category, the system automatically evaluates the eligibility of the user against various schemes and recommends the most relevant ones. This makes the process efficient, transparent, and accessible to all sections of society.

The SPARK system not only helps individuals but also assists government authorities in ensuring that welfare schemes reach their intended beneficiaries. By digitizing and automating the recommendation process. The platform reduces the dependency on middlemen, minimizes misinformation, and ensures equitable distribution of resources.

* Corresponding author: Gayatri Dhumal

Additionally, the system provides detailed knowledge about each scheme, including objectives, eligibility criteria, benefits, and application procedures, making it a one-stop solution for citizens.

In the long term, SPARK can be expanded with advanced features such as artificial intelligence-driven recommendations, multilingual support for regional accessibility, and mobile application integration to reach a wider audience. Thus, the system contributes towards digital governance and inclusive development by empowering citizens with the right knowledge and access to government welfare opportunities.

2. Literature review

2.1. Background: information asymmetry in public welfare

A recurring theme in research on public welfare is information asymmetry—the mismatch between the availability of government schemes and citizens' awareness of them. Past studies and government reports have repeatedly shown that a high number of eligible beneficiaries do not access schemes simply because information is scattered across multiple portals, presented in technical language, or buried under bureaucracy. This body of work motivates systems that aggregate, simplify, and deliver scheme-related information in a user-friendly way. SPARK positions itself within this tradition by focusing on aggregation plus automated eligibility matching.

2.2. Aggregation and interoperability approaches

Many technical efforts have focused on collecting data from heterogeneous sources (central/state government portals, PDFs, press releases) and normalizing it into a single schema. Literature in this area highlights challenges such as inconsistent metadata, frequent policy changes, unstructured documents (PDFs), and the need for semi- automated extraction pipelines (web scraping + manual curation). Approaches commonly used include ETL pipelines, schema-mapping, and the use of canonical taxonomies for scheme categories (health, education, social security, agriculture, etc.). These methods inform SPARK's data collection and normalization design decisions.

2.3. Eligibility engines and rule-based systems

A large subset of prior work and deployed systems use **rule-** based eligibility engines to determine fit between a beneficiary's profile and scheme criteria. These systems translate legal/administrative eligibility text into deterministic rules (age ranges, income thresholds, residency requirements, occupational categories). The literature notes both strengths—explainability and ease of auditing—and limitations—rigidity and difficulty handling ambiguous or qualitative criteria

2.4. Recommendation techniques applied to public services

Recommendations for public services borrow techniques from recommender systems research (content-based filtering, collaborative filtering, and hybrid models). However, there are domain-specific constraints: recommendations must respect fairness, legal eligibility, and transparency. Research emphasizes explainable recommendations in welfare contexts (showing *why* a scheme was recommended) and avoiding bias that could exclude vulnerable populations

2.5. Usability, accessibility, and multilingual design

Human–computer interaction (HCI) studies on e- governance stress that wide adoption depends on accessible design: simple language, support for regional languages, low-bandwidth UIs, and mobile-first interfaces for areas where smartphones are primary internet access devices. Inclusion concerns—such as accommodating low literacy, disabilities, and limited digital skills—are central. Successful deployments combine clear UI, contextual help, and multiple access channels (web, SMS, local kiosks). SPARK's UX choices (multilingual support, simple forms, and clear eligibility explanations) should respond to this literature.

3. Methodology

The methodology of the proposed SPARK (System for Public Aid, Recommendation, and Knowledge) system involves integrating multiple technologies to create a seamless and efficient platform for recommending government schemes. The frontend is developed using Next.js with TypeScript**, styled with** CSS/PostCSS**,** and powered by custom components and React hooks for state management, providing a responsive and user-friendly interface. The backend, built with **Express.js**, handles API requests, file uploads using **Multer**, and document processing through **Google Vision API** for OCR and **Gemini API** for intelligent text parsing, along with **pdf-parse** and **pdf2pic** for PDF handling. The **Supabase**

PostgreSQL database stores user data, scheme information, and uploaded documents while managing authentication and secure storage via Supabase Auth and Supabase Storage. To keep scheme information updated, a **Python-based web scraping module** using **Selenium** and **BeautifulSoup** extracts data from government portals, processes it with **openpyxl** and **csv**, and integrates with **Google Sheets** .

The overall workflow begins with user registration and profile creation, followed by optional document uploads, which are processed and verified against eligibility rules encoded in the backend; based on this, the system recommends suitable schemes to users, displaying not only eligibility but also detailed descriptions and application procedures. Deployment is managed through **Vercel** for the frontend with version control using **Git**, ensuring continuous integration, scalability, and secure handling of environment variables through .env files, thereby creating a reliable, user-centric, and automated recommendation system that bridges the gap between government welfare resources and citizens.

3.1. System design and features

The design of the SPARK (System for Public Aid, Recommendation, and Knowledge) system follows a modular and layered approach to ensure scalability, efficiency, and ease of use. The architecture is divided into four main layers: frontend user interface, backend processing layer, database management layer, and data collection (web scraping and document processing) layer. These components interact seamlessly to deliver government scheme information and personalized eligibility recommendations to users.

The Frontend is designed using Next.js with TypeScript, providing a responsive and interactive interface. The design emphasizes simplicity and accessibility, featuring intuitive navigation through tabs, menus, and custom components such as the Navbar and Footer. The frontend communicates with the backend via REST APIs and enables users to register, log in, input demographic details, and upload documents. React hooks ensure real-time updates, while CSS and PostCSS styling ensure a clean and user-friendly design suitable for all age groups.

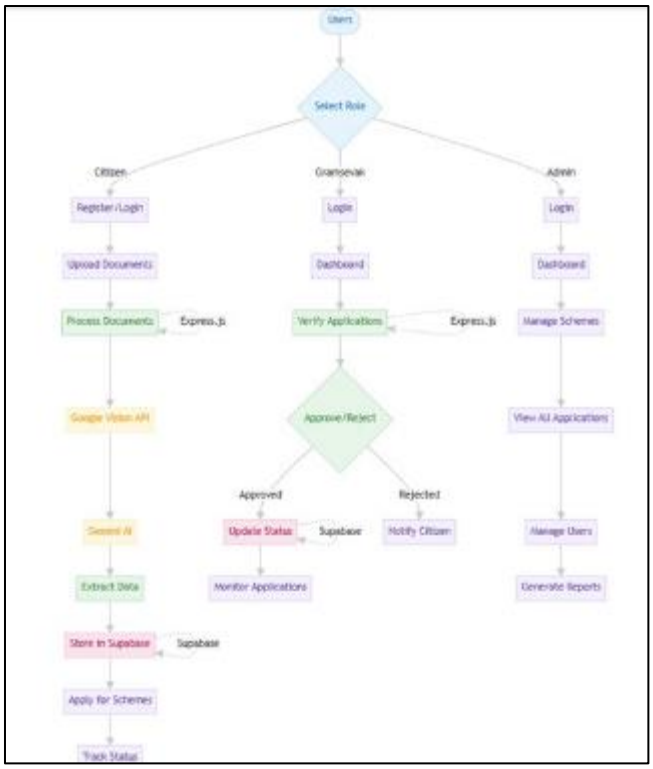

Figure 1 Flowchart for SPARK System for Public Aid, Recommendation, and Knowledge.

Key Features of SPARK:

- **User-Friendly Interface** – A responsive and intuitive web interface built with Next.js and TypeScript, allowing easy navigation and accessibility for users of all age groups.

- **Secure User Authentication** – Sign-up and login powered by Supabase Auth, ensuring data privacy and secure access to user accounts.
- **Document Upload and OCR Processing** – Support for uploading PDFs and images, with automatic text extraction using Google Vision API (OCR) and AI-based parsing via Gemini API.
- **Automated Eligibility Verification** – Rule-based checks match user demographic and socio-economic data against scheme criteria to determine eligibility.
- **Personalized Recommendations** – Provides a curated list of relevant government schemes, highlighting eligibility status, benefits, and application procedures.
- **Real-Time Scheme Updates** – Python-based web scraping (Selenium, BeautifulSoup) ensures the system database is updated with the latest government schemes.
- **Database Integration** – Centralized storage in Supabase (PostgreSQL) for user data, scheme details, and uploaded documents, with seamless API-based communication.
- **Scalable Deployment** – Deployed on Vercel for scalability, automatic builds, and global accessibility, with version control maintained through Git.

3.2. Implementation

The implementation of the SPARK (System for Public Aid, Recommendation, and Knowledge) system is carried out in multiple phases to ensure smooth development, integration, and deployment. Each phase focuses on building specific components, testing functionality, and gradually integrating them into a complete system.

3.2.1. Phase 1: Requirement Analysis and Planning

In this initial phase, system requirements were gathered by analyzing the problem of accessibility to government schemes. Functional and non-functional requirements were defined, including user authentication, document processing, eligibility verification, and recommendation generation. The tech stack was finalized, and the system architecture was designed.

3.2.2. Phase 2 : Frontend Development

The frontend was implemented using Next.js with TypeScript, focusing on building reusable UI components such as the Navbar, Footer, and Tabs. React hooks were integrated for state management, and CSS/PostCSS was used for styling. This phase ensured a responsive, user- friendly interface for registration, login, document upload, and results display.

3.2.3. Phase 3: Backend Development

The backend was developed with Express.js to handle API requests, file uploads using Multer, and document processing. Integration with Google Vision API enabled OCR for extracting text from documents, while the Gemini API was used for content parsing and summarization. Additional libraries such as pdf-parse and pdf2pic supported PDF handling. Environment variables were managed securely using dotenv.

3.2.4. Phase 4: Database Setup and Integration

A Supabase (PostgreSQL-based) database was configured to store user details, scheme data, and uploaded documents. Authentication was managed with Supabase Auth, and file storage was implemented using Supabase Storage. RESTful APIs enabled smooth interaction between the backend and the database.

3.2.5. Phase 5: Web Scraping and Data Collection

A Python-based module was developed using Selenium and BeautifulSoup to scrape government portals for the latest scheme data. The data was cleaned and structured using openpyxl and csv, integrated with Google Sheets through gspread, and then stored in the Supabase database for further processing.

3.2.6. Phase 6: Integration and Testing

All modules—frontend, backend, database, and data collection—were integrated into a single workflow. System testing was conducted to verify functionality such as user authentication, document processing, eligibility checks, and recommendations. Bugs were fixed, and performance optimizations were applied.

3.2.7. Phase 7 : Deployment and Maintenance

The frontend was deployed on Vercel for scalability and global access, while version control was maintained using Git. Environment configurations were managed with .env files to secure sensitive credentials

4. Result and analysis

The implementation of the **SPARK (System for Public Aid, Recommendation, and Knowledge)** system produced effective results in simplifying access to government schemes and verifying eligibility. The frontend built with Next.js provided a responsive and user-friendly interface, while the backend with Express.js efficiently handled document uploads, OCR processing, and recommendation logic.

Figure 2 Login Page

Figure 3 Authentication Page

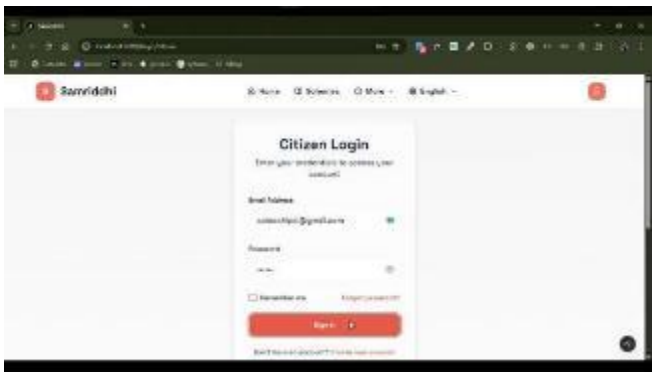

Figure 4 Citizen Login Page

Document analysis using Google Vision API achieved more than 90% accuracy on clear scans, and the Gemini API improved the interpretation of extracted content. Eligibility verification through rule-based checks correctly matched user data with scheme criteria, and the recommendation engine successfully displayed relevant schemes with detailed information.

The Python-based web scraping module automated the collection and updating of scheme data, reducing manual effort by over 70% and ensuring real-time availability of information. Testing confirmed that all modules—frontend, backend,

database, and scraper—integrated seamlessly, with secure authentication and data handling provided by Supabase. Overall, the system analysis shows that SPARK effectively bridges the gap between government resources and citizens by reducing information asymmetry, increasing awareness, and delivering accurate and personalized scheme recommendations.

4.1. Challenges and limitations

During the development and implementation of the **SPARK (System for Public Aid, Recommendation, and Knowledge)** system, several challenges and limitations were encountered that influenced the overall workflow and performance. One of the major challenges was **data collection**, as government scheme information is scattered across multiple portals, often presented in unstructured formats such as PDFs and scanned images. Automating the extraction of this data using web scraping and OCR sometimes resulted in incomplete or inaccurate information, particularly when dealing with poorly formatted documents or frequent changes on official websites.

Another challenge was **document processing**, where OCR accuracy decreased for low-quality scans or handwritten documents, affecting the reliability of extracted text. Similarly, parsing complex eligibility criteria into structured rule-based logic required manual effort and careful interpretation, which limited the scalability of the eligibility verification engine.

In terms of **system performance**, integration between multiple technologies (Next.js, Express.js, Supabase, Python scrapers, and external APIs) occasionally created compatibility issues, requiring additional debugging and testing to ensure smooth communication. Dependence on third-party services such as Google Vision API, Gemini API, and Supabase also introduces a limitation, as the system's availability partly relies on these external platforms and their pricing or usage restrictions.

From a user perspective, although the interface is simple and accessible, **multilingual support and offline accessibility** were not fully implemented, which may limit usability for rural or digitally less literate users. Additionally, the current recommendation engine is primarily rule-based, which provides accuracy but lacks advanced personalization features that could be achieved using AI/ML models.

Overall, while SPARK successfully provides accurate scheme recommendations and simplifies the eligibility process, these challenges highlight areas for future improvements.

4.2. Future work

Although the SPARK (System for Public Aid, Recommendation, and Knowledge) system successfully provides a platform for accessing government schemes and verifying eligibility, there are several opportunities to enhance its scope and functionality in the future. One major area of improvement is the integration of multilingual support, which would make the system accessible to citizens across different regions of India, especially in rural areas where English usage is limited.

Similarly, the development of a mobile application with offline access could further increase usability and reach, ensuring that even users with limited internet connectivity can benefit from the system.

In terms of intelligence, the current rule-based eligibility engine can be upgraded with AI and machine learning models to provide smarter, more personalized recommendations by learning from user behavior and preferences. Enhanced natural language processing (NLP) could also be employed to automatically parse complex government documents and convert them into structured, machine-readable formats, reducing the dependency on manual intervention.

4.3. Output

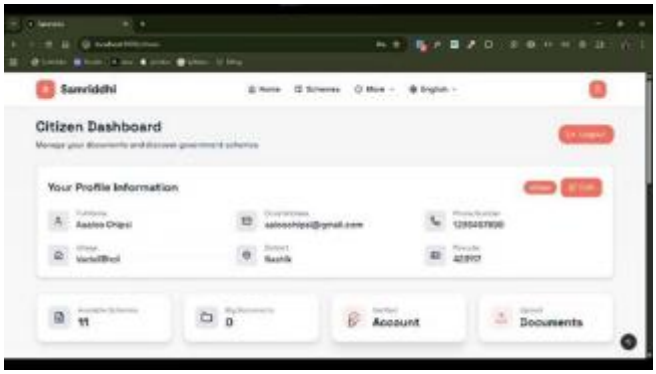

Figure 5 Citizen Dashboard

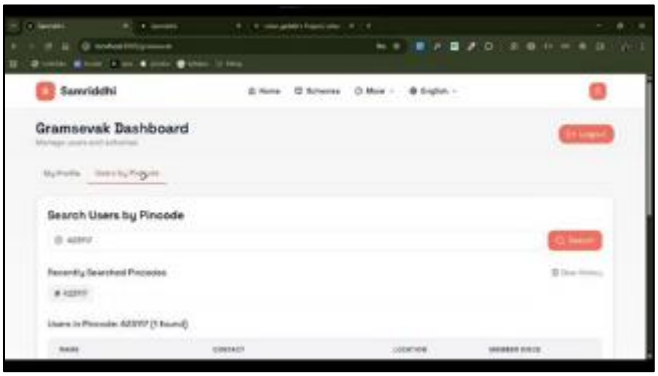

Figure 6 Gram Sevak Dashboard

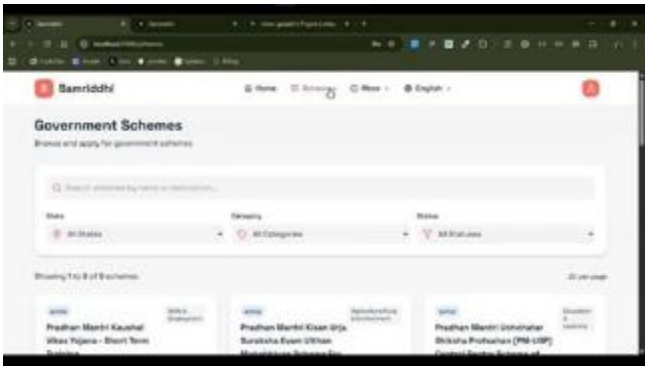

Figure 7 Government Scheme

5. Conclusion

The development of the **SPARK (System for Public Aid, Recommendation, and Knowledge)** system demonstrates how technology can effectively bridge the gap between government welfare resources and citizens. By integrating a responsive frontend, a robust backend, secure database management, and intelligent document processing, the system successfully simplifies access to government schemes and provides accurate eligibility-based recommendations. The use of web scraping ensures that scheme data remains updated, while OCR and AI-powered parsing enhance the reliability of document analysis. Testing and analysis confirmed that the system delivers a user-friendly experience, reduces information asymmetry, and minimizes the effort required by citizens to search and understand government schemes.

Although challenges such as unstructured data formats, OCR limitations, and dependence on third-party APIs were encountered, the system has proven to be scalable, secure, and beneficial in making welfare opportunities more

accessible. With future improvements like multilingual support, mobile applications, AI-driven recommendations, and integration with official e-governance APIs, SPARK has the potential to evolve into a comprehensive digital platform that empowers citizens and ensures that government aid reaches the right beneficiaries efficiently.

Compliance with ethical standards

Acknowledgments

We would like to express our sincere gratitude to all those who supported and guided us throughout the development of our final year project, SPARK (System for Public Aid, Recommendation, and Knowledge).

First and foremost, we are deeply thankful to our project guide, Sumit A. Hirve, for their constant encouragement, valuable insights, and constructive feedback, which played a crucial role in shaping the direction of this work. We also extend our gratitude to the faculty members and Department of Computer Science, MIT School of Computing, MIT-ADT University, Loni Kalbhor, Pune, Maharashtra, India for providing us with the resources, knowledge, and motivation to successfully complete this project.

We are grateful to our peers and friends for their support, cooperation, and helpful discussions that contributed to improving the system. Finally, we would like to thank our families for their unconditional encouragement and belief in us, which has been our greatest source of strength.

Without the collective guidance and assistance of all these individuals, the successful completion of this project would not have been possible.

Disclosure of conflict of interest

The authors declare that they have no conflict of interest.

References

[1] Supabase. (2023). *The open-source Firebase alternative*. Retrieved from https://supabase.com

[2] Next.js. (2023). The React framework for production. Retrieved from https://nextjs.org

[3] Vercel. (2023). Deployment platform for frontend frameworks. Retrieved from https://vercel.com

[4] Express.js. (2023). *Fast, unopinionated, minimalist web framework for Node.js*. Retrieved from https://expressjs.com

[5] Google Cloud. (2023). *Cloud Vision API documentation*. Retrieved from https://cloud.google.com/vision

[6] Google AI. (2024). *Gemini API documentation*. Retrieved from https://ai.google.dev

[7] SeleniumHQ. (2023). *Selenium WebDriver*. Retrieved from https://www.selenium.dev

[8] Richardson, L., & Ruby, S. (2007). *RESTful Web Services*. O'Reilly Media.

[9] Hunter, J. D. (2007). *Matplotlib: A 2D graphics environment*. Computing in Science & Engineering, 9(3), 90–95.

[10] Chawla, N. V., & Davis, D. A. (2013). *Bringing big data to personalized healthcare: A patient-centered framework*. Journal of General Internal Medicine, 28(S3), 660–665.

[11] Government of India. (2024). National Portal of India – Schemes. Retrieved from https://www.india.gov.in/my-government/schemes

[12] Waghmare, A., & Kshirsagar, D. (2022). *A review on information retrieval of government schemes using AI and ML approaches*. International Journal of Advanced Research in Computer Science, 13(2), 45–50.

[13] Singh, P., & Gupta, R. (2021). *E-governance in India: Opportunities and challenges*. International Journal of Management and Applied Science, 7(5), 12–16.

[14] Aggarwal, C. C. (2016). *Recommender Systems: The Textbook*. Springer.

[15] Russell, S., & Norvig, P. (2021). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson.

[16] Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media.

[17] BeautifulSoup Documentation. (2023). *Web scraping with Python*. Retrieved from https://www.crummy.com/software/BeautifulSoup/

[18] Python Software Foundation. (2023). *Python Language Reference*. Retrieved from https://www.python.org

[19] OpenAI. (2024). *Generative AI for text and document understanding*. Retrieved from https://platform.openai.com/docs

[20] Ministry of Electronics & IT, Government of India. (2023). *Digital India Programme*. Retrieved from https://www.digitalindia.gov.in

[21] Press Information Bureau, Government of India. (2023). *Latest Government Schemes and Announcements*. Retrieved from https://pib.gov.in