Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

SolutioNet.al: A Community-Driven Platform for Discovering and Evolving Early-Stage Real-World Problem Ideas

Dr.I. Sundara Siva Rao (Professor), S. Bhuvaneswari, SK. Sandani, J. Janu Kalyan Gowd,
V. Naga Bala Vamsi
Department of Computer Science and Engineering (Data Science)
Anil Neerukonda Institute of Technology and Sciences
India

***Abstract*—Many real-life problem-solving activities originate at the conception level but the currently available systems deal mainly with final solutions or published work. This causes a gap, where problem conceptions go mostly undocumented. In this paper, we propose SolutioNet.al – a problem-focused social networking website that will be used for structuring and sharing such problem ideas by allowing users to submit, explore, and discuss the problem ideas using features like problem recognition, classification, and identification. The platform also incorporates AI-powered summary generation of solutions provided by users as well as problem ideas recommended based on the problem solutions suggested by users. By moving the emphasis away from solution-oriented systems towards problem-oriented discussions, our solution allows for collaborative problem-solving.**

***Index Terms*—Community-driven problem discovery, Collaborative idea evolution, Problem-centric innovation, Real-world problem exploration, AI-assisted problem understanding, Intelligent content summarization, Knowledge sharing platforms**

I. Introduction

The first step of innovation is recognizing the problem itself. Nevertheless, in practice, initial ideas about potential problems are quite scattered and not structured enough. People usually come up with some issues or interesting concepts but do not have an appropriate venue to discuss them without being forced to suggest immediate solutions. Thus, a lot of good ideas get wasted because of this problem.

There exist numerous venues for solved problems, coding challenges, and peer-reviewed scientific papers. These are useful initiatives, however, they belong to a different phase of the innovation cycle since they help in solving problems rather than finding them. In addition, such initiatives cannot help people looking for practical problems to solve, which is typical for students and innovators who just started their journey.

To bridge this gap, this paper introduces SolutioNet.al, which is a community-based system that supports the process of structured problem identification and collaborative improvement of ideas. This tool enables its users to share, browse and interact with concrete problem statements by means of such functionalities as standardized identification, classification, and community-driven processes including voting and bookmarking.

Moreover, some intelligent features are included in the process, such as problem-related solution summary, automatically generated via machine learning technology. The focus of the system is shifted from solutions to problems, thus promoting openness and collaboration.

II. Literature Survey

Current scientific literature stresses the growing relevance of collaboration, open innovation, and open knowledge dissemination in solving problems in the real world. There is a trend in scientific publications toward investigating systems linking people, knowledge, and research, yet all the discussed cases concentrate mostly on either solution or knowledge production.

[1] ResearchBot suggests developing a platform to link scientific articles with programming by searching through articles and creating an evidence-based answer to questions put forward by users. Though ResearchBot makes scientific research more accessible to people, it concentrates mainly on responding to queries rather than fostering open-ended discussion.

[2] Open science systems like MyResearchChallenge prove that involving people into scientific work at its early stages, namely at the stage of finding problems to be solved, is feasible. However, there exist some issues which should be overcome when developing such systems: scalability, moderation, and domain coverage are only several of them.

[3] Studies concerning impactful research have focused on the divide between academic soundness and practical value. Current frameworks have been known to favor theory over practicality and societally relevant implications, thereby pointing out the necessity for mediums that can help link concrete problems with joint exploration efforts.

[4] The online ecosystem Kaggle is one example of how communities can collectively build knowledge through iteration and insight sharing. Although such ecosystems are competitive in nature, they enable collaboration and creativity.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

However, they are mostly problem-solving mediums that necessitate clearly defined problems, making them less suitable for exploring nascent ideas.

[5] Studies related to GitHub repositories associated with scientific publications point out the importance of traceability from research to implementation. Although such connections promote reproducibility, they are scarce and standardized, reflecting the disconnect between knowledge gained from research and application ecosystems.

[6] Platforms for global exchange of ideas have also been suggested to promote interdisciplinarity and innovation. These platforms have features like submission of ideas, feedback mechanisms, and recommendation engines. Nevertheless, these platforms lack practical proof of concept, structure for problem identification, and long-term engagement of users.

[7] The notion of Open Innovation in Science underscores the requirement for a participative and collaborative scientific approach. This theory stresses the inclusion of multiple actors at different stages of research, including idea generation. But the majority of frameworks are theoretical in nature and lack implementation.

[8] Collaboration platforms that are student-centric illustrate the significance of sharing projects, artificial intelligence-assisted summarization, and collaboration across institutions. These platforms foster collaboration; however, they are mostly concerned with projects and solutions rather than problems.

[9] From studies related to internal crowdsourcing, one finds that solution-driven contributions positively impact idea acceptance while an over-emphasis on the problem discussion aspect could be detrimental to making any progress.

[10] Last but not the least, research regarding grand scientific challenges highlights the need for formulating meaningful problems which act as a motivating factor to foster innovation and encourage inter-disciplinary collaboration. Formulating such problems in themselves can prove to be very helpful in the process of scientific discovery.

III. Problem Statement

In today's innovative ecosystem, there is an evident void in the first stage of problem exploration and discovery. There are quite a number of platforms aimed at solving problems, code practice, and publication of research articles; however, there is no platform allowing one to share the idea of problems and explore it from different angles before the actual research takes place. Often, students, constituting an important group of up-and-coming innovators, face difficulties in finding relevant topics for their research. On the other hand, when people encounter certain challenges or even ideas that need addressing, they do not have a means to share and develop them without the necessity to solve it immediately.

Existing systems tend to be solution-focused and formalized, which is why people are unwilling to discuss and share their raw, unorganized ideas. It is often the case that problem formulations are either left out of the process, neglected, or forgotten due to which people end up working on them again and again because of lack of awareness about previous studies. This may significantly hamper innovation as well as reduce the efficiency of students' projects. Thus, a system allowing one to discover, collaborate, and develop problem ideas would be quite helpful.

IV. Objectives

The primary goal of SolutioNet.al is to design a community-oriented framework for the exchange and discussion of early-stage problems from real life. This tool will facilitate the process for users, especially students, to discover interesting problems to tackle, as well as enable everyone to post and modify problem statements in a structured manner. Moreover, it will focus on making the process of problem discovery more efficient with the aid of numbered, filtered, voted-on, bookmarked, associated problems, and AI-powered summaries of several perspectives on solutions.

V. Proposed Methodology

The proposed framework SolutioNet.al will be a community-based web portal where users will be able to post, browse, and collaborate on emerging ideas of problems. The methodology will involve organizing the unstructured problem inputs, interactions, and discovery based on intelligent techniques.

A. Overview of the System

The proposed system will implement a client-server architecture where the client side directly communicates with the back-end services. Users will be able to log-in to the portal for authenticating themselves, submitting new problems, interacting with posted problems, and providing solutions to the problems.

B. Problem Submission Process

Following is the step-wise process carried out during the submission of a problem statement:

- Authentication of the user through a session.
- Collection of problem submission fields such as title, problem domain, description, abstract, proposed solutions, keywords, and optionally external citations.
- Insertion of the problem details collected into the database along with user identification.
- Generation of PID for the problem which should be unique and humanly readable to refer to the problem.
- Restricting creation/modification capabilities of the user based on access policies.

C. Interaction with the Problem

During interaction with the problem, the platform provides information on:

- Details of the problem and information about the author
- Community activity, which includes comments, solution perspectives, etc.
- Voting data, which is upvoting and downvoting
- User status, such as if the problem is saved or followed by the user

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

Moreover, the platform allows for real-time updates. Thus, the user does not have to manually reload the page in order to see the changes made.

D. Information Storage

In order to manage and organize data effectively, the platform uses a relational model. Key entities are:

- Users and their profiles
- Problem descriptions
- Information about votes and bookmarks
- Comments and discussion threads
- Contributions of solutions related to the problems

Thus, the platform can provide an efficient user experience.

E. AI Features

In order to improve user experience and make problem finding easier, the platform uses two main AI features:

1) ***Solution Summarization****:* A summary is automatically generated by combining all different user-provided perspectives on the solution of a given problem. Information overload is avoided, making it possible for users to grasp important concepts by simply reading an overall summary instead of looking at individual solutions.

2) ***Related Problem Recommendation****:* Problem statements are embedded and analyzed based on their semantic similarity to determine related problems. This makes it possible to find similar concepts within SolutioNet.al that a given user might be interested in exploring. Fallback strategies like domain and keyword matching are employed as a last resort in case AI-driven techniques cannot be used for some reason.

F. System Architecture

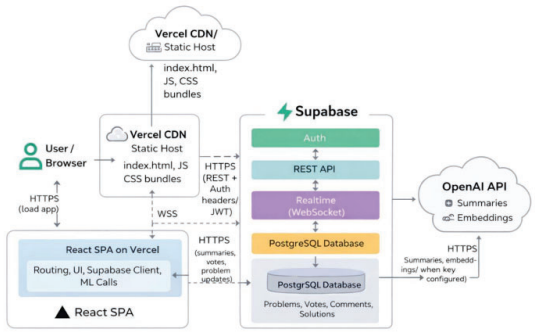

Fig. 1: System Architecture of SolutioNet.al

SolutioNet.al uses a three-tier cloud architecture, which comprises a client-side single-page application, backend-as-a-service capabilities, and external AI-based services. A React application is used as a client side, hosted by Vercel. All other functionalities of the backend server are provided by Supabase.

Integrations of external AI services are provided using OpenAI APIs for solution summarization and recommendations based on problem similarities. There is no need for a special application server; communication between the client, backend, and AI components is directly provided by the system.

The system architecture is designed according to the modern web model comprising three main layers:

1) ***Presentation Layer****:* It represents a single-page React application that renders a user interface and manages client-side actions and routing. It is hosted on Vercel to provide efficient content delivery and scalable performance.

2) ***Client Logic Layer****:* This layer manages the application's logic running in the browser, implementing the user's authentication state, navigation, interaction with the backend services, and other tasks. It also contains the modules for AI-related features that call external AI services.

3) ***Backend-as-a-Service Layer****:* The application backend is built with the help of Supabase framework that provides the following:

- User authentication via session tokens
- Access to database through the REST API
- Synchronization with the backend using WebSocket
- Policies of row-level security

4) ***Data Layer****:* The system makes use of the PostgreSQL database provided by Supabase. It contains data in a structured form like user profiles, problem definitions, voting, bookmarking, comments, and problem resolutions. The database is set up with triggers that produce unique problem IDs and calculate derived columns like voting scores.

5) ***AI Services Layer****:* Third-party AI services are leveraged to improve system intelligence. This includes:

- Solution viewpoints summarization using a small language model
- Problem recommendations based on semantic similarity through text embeddings

These services are called through secure HTTP calls originating from the browser client.

G. Actors and Interaction Diagram

The system actors include:

- User / Browser: Starts interaction sessions and uses the platform
- Vercel Hosting Layer: Provides the static frontend code
- React Frontend: Renders UI and handles user interaction
- Supabase Backend Services: Manages authentications, data, and real-time communications
- OpenAI Services: Uses AI capabilities to summarize and recommend problems

1) Methods of Communication:

- Browser → Vercel: HTTPS requests for fetching app resources
- Browser → Supabase: HTTPS requests for performing CRUD operations with authentication
- Browser → Supabase Realtime: WebSocket connections for live notifications
- Browser → OpenAI: HTTPS requests for AI features

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

ALGORITHM 1: SEMANTIC SIMILARITY-BASED PROBLEM RECOMMENDATION

Algorithm 1: Related Problem Recommendation

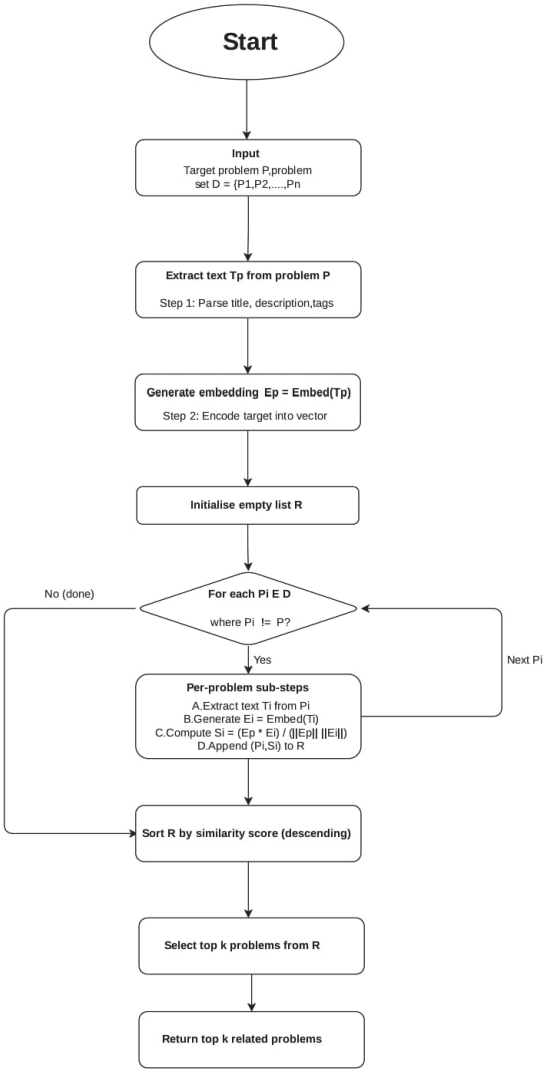

Fig. 2: Semantic Similarity-Based Problem Recommendation

Input:
Target problem P, Set of problems $D = \{P_1, P_2, ..., P_n\}$
Output:
List of top k related problems
Steps:

1) Extract textual content T_p from problem P
2) Generate embedding vector $E_p = Embed(T_p)$
3) Initialize empty list R
4) For each problem $P_i \in D$, where $P_i \neq P$:
   a) Extract textual content T_i
   b) Generate embedding vector $E_i = Embed(T_i)$
   c) Compute similarity score:

$$S_i = \frac{E_p \cdot E_i}{\|E_p\| \|E_i\|}$$

   d) Append (P_i, S_i) to R
5) Sort list R in descending order of similarity scores
6) Select top k problems from R
7) Return selected problems

ALGORITHM 2: AI-BASED SOLUTION SUMMARIZATION

Algorithm 2: Multi-Solution Summarization

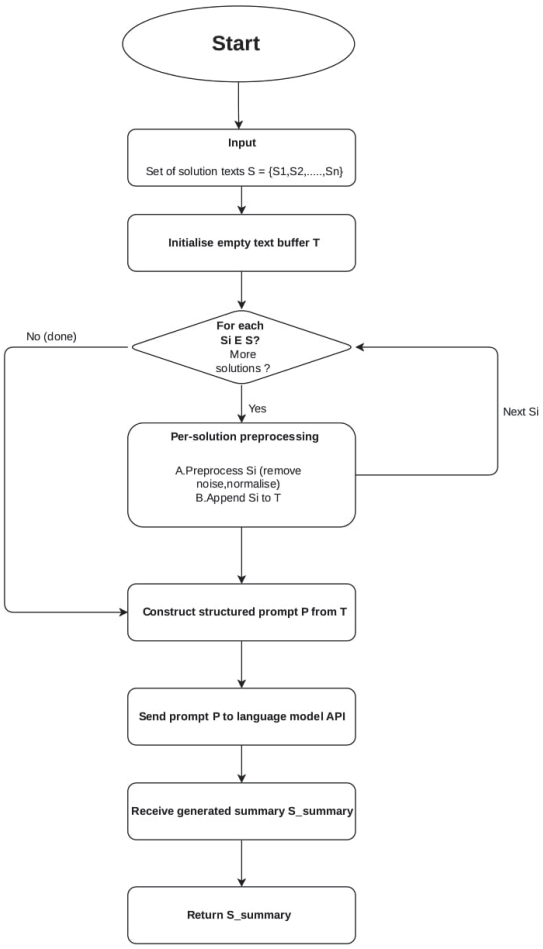

Fig. 3: AI-Based Solution Summarization

Input:
Set of solution texts $S = \{S_1, S_2, ..., S_n\}$
Output:
Concise summary $S_{summary}$
Steps:

1) Initialize empty text T
2) For each solution $S_i \in S$:
   a) Preprocess S_i (remove noise, normalize text)
   b) Append S_i to T
3) Construct structured prompt P using T
4) Send prompt P to language model API

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

5) Receive generated summary $S_{summary}$
6) Return $S_{summary}$

H. Data Flow

*1) **Problem Submissions Data Flow**:* A user submits a problem using the UI of the front end application. The request is authenticated and forwarded to the back end server. The problem is stored into the database. A unique identifier is created and the user is redirected to the problem page.

*2) **Problem Discovery Data Flow**:* When a user opens a problem page, data retrieval takes place. This includes the problem itself, comments, solutions, and any other information associated with it. Data subscriptions allow for real-time updates. If required, AI services can be utilized to perform tasks like generating summaries and recommending problems.

VI. Implementation

The implementation strategy of SolutioNet.al involves developing a scalable and interactive web application that supports real-time functionalities using current web frameworks and backend as a service architecture.

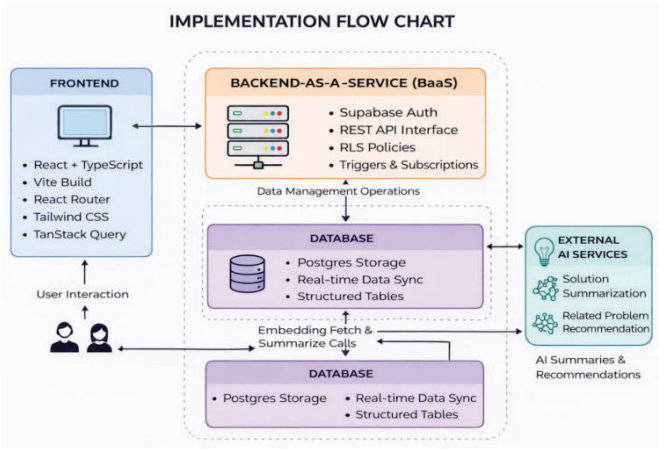

Fig. 4: Flowchart of Implementation Strategy of SolutioNet.al showing interaction between frontend, backend services, database, and AI components

A. Frontend Implementation

For the frontend, we utilize the React framework with Typescript to provide code reusability and modularity while developing an advanced user interface. We use Vite as our build tool to speed up the development process and maximize the performance of the website. Routing between different views such as list of problems, problem detail view, and user views is handled by React Router.

For styling, we use the Tailwind CSS framework alongside various libraries and reusable UI primitives. Data loading and state management are done using the query-based approach to maximize synchronization with backend data.

B. Backend Implementation

Supabase backend-as-a-service solution is utilized in the current system. This technology allows us to avoid setting up our own servers by providing features such as authentication, database connectivity, and real-time events handling.

Authentication on the application side is done using session-based mechanisms allowing us to control access to different functions provided by the platform. Row-level security is enforced to guarantee that a user cannot change any data other than his own.

The backend implements a REST-like API that enables frontend interaction with the database. The application also has real-time event subscription capabilities provided via WebSocket protocol.

C. Database Design

The platform uses PostgreSQL database to store application data. The database structure includes the following entities: users, profiles, problems, votes, bookmarks, comments, and solutions.

A unique human-readable identifier for each problem is generated via triggers inside the database. Triggers are also used to update some calculated fields, including a problem's votes.

Relational nature of the database allows implementing all required functionality, including filtering and sorting.

D. Integration with External AI Services

AI services are leveraged by the platform to improve usability and intelligent discovery.

- **Solution Summarization:** In cases where there are different perspectives on how a solution can be found, the input from multiple sources is combined, and a brief summary is generated via a lightweight language model. This enables users to easily grasp the major concepts without having to read all the submitted work.
- **Problem Recommendation:** Using the power of text embeddings models, the problem statement is vectorized and compared using cosine similarity. This allows the system to identify and recommend problems that are conceptually similar.

Fallbacks have been built to provide consistent behavior in case the AI service fails. These include domain and tag-based matching.

E. Deployment and Integration

Deployment is handled using the cloud-hosted infrastructure. The frontend is deployed as a static web application, whereas the backend services run in the Supabase ecosystem.

Environment-specific configuration is used to safely store API keys and other configurations.

VII. Results and Outputs

The system has been tested for performance, usability, and how well it enables structured problem identification and collaboration. This evaluation will include information about system performance and usability.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

A. System Performance Metrics

This system displays high-performance capabilities and fast response times for users. Performance metrics for this system have been collected using various sources.

Component	Average Response Time
Problem Page Load (API)	∼85 - 95 ms
Related Problem Fetch	∼0.93 sec
AI Summarization	∼5.1 sec
Time to First Byte (Desktop)	∼0.37 sec
Largest Contentful Paint (Desktop)	∼2.44 sec
Largest Contentful Paint	∼3.72 sec

TABLE I: System Performance Metrics

The following metrics have been explained as follows according to their definitions:

- **Problem Page Load (API):** The metric reflects how long it takes to load problem data using an API. It includes the database request and processing. According to our observations, latency is around 85-95 milliseconds, which demonstrates efficient back-end work.
- **Related Problem Fetch:** It shows how much time is needed to compute related problems on the basis of their semantic similarities. It involves embedding comparisons and rankings of related items. Response time of 0.93 seconds proves that this function performs well in near real-time.
- **AI Summarization:** It shows how long the AI-powered summarization algorithm works. As a rule, this feature requires about 5.1 seconds per process. However, since users use it manually, there is no need to be concerned about high latency.
- **Time to First Byte (TTFB):** TTFB refers to the duration that elapses between the moment a user sends a request to when the server responds by sending the first byte of information to the client side. The value of 0.37 seconds for the desktop device is quite commendable.
- **Largest Contentful Paint (LCP):** LCP refers to the duration required for the loading process of the website before displaying the main content of the web page to the user. The values of 2.44 seconds for the desktop and 3.72 seconds for the mobile device can be accepted as reasonable figures.

B. Performance Analysis

- Fast backend response times are provided (¡100ms) for core functionality like fetching problems, thereby making navigation easy and smooth.
- The recommendation system that recommends related problems performs in ¡ 1 second, thus allowing exploration of questions based on semantic similarities almost instantaneously.
- The high latency experienced by the AI-based summarization is ∼ 5 seconds and is expected as an external model is used. However, since this functionality is invoked manually by the user, there are no performance implications.
- Performance of frontend features indicates efficient loading speeds especially when compared to desktop versions.

C. Visual Representation of System Performance

Below is the graphical representation of the response times of the system components.

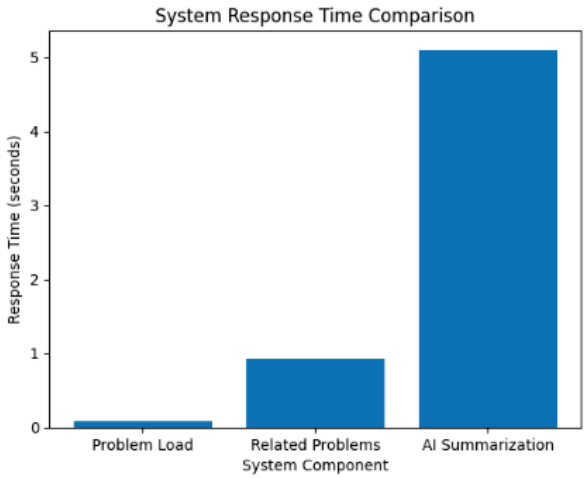

Fig. 5: Comparison of response times of system components

From Figure 3 above, it can be observed that the AI-based summarization takes more time than others while other system functionalities have very low response times.

D. Functional Outcomes

It should be noted that the platform successfully delivers:

- Structured submission and retrieval of problems
- Effective filtering and discovery of problems specific for a certain domain
- Real-time interactions using such actions as voting, bookmarking, discussion etc.
- Intelligent assistance provided by summarization and recommendation tools

All these outcomes clearly demonstrate the feasibility of the system for early stage problem exploration.

E. User Interface

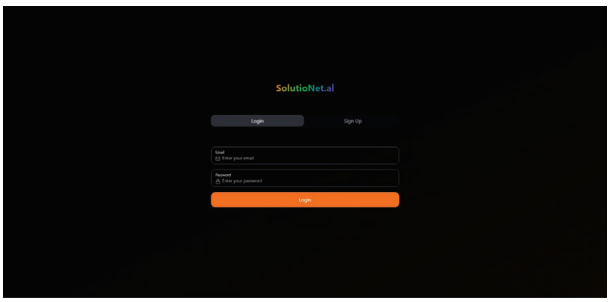

Fig. 6: User Login Page

The figure above (Fig.6) illustrates the user login page. User login page provides a secure way to enter into the SolutioNet.al

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

system with the help of a login form for user authentication. This tool ensures data privacy and controlled access while offering simplicity and efficient user interaction.

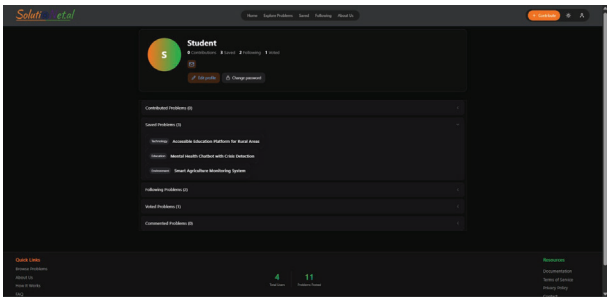

Fig. 7: User Details Page

As we can see from the Fig.7 below, the user details page offers an easy way to control personal data and activities performed on this platform, like contributions and bookmarks.

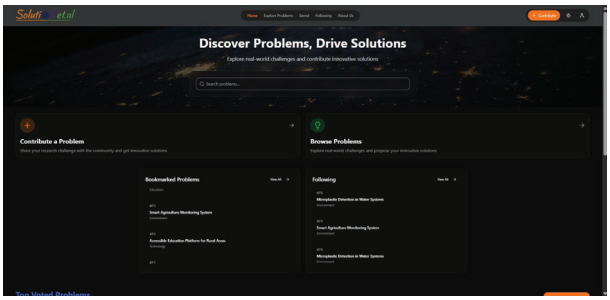

Fig. 8: Home Page

The Fig.8 is The Home Page is designed to be the central dashboard of the platform, offering easy navigation to users for quick access to essential features and updates on the recent problem. It gives an idea about the content available on the platform and helps users navigate and interact within it using a straightforward interface.

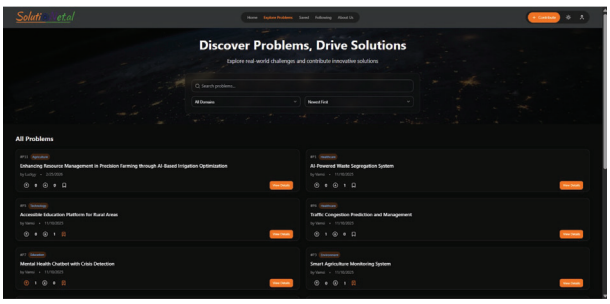

Fig. 9: Explore Problems Page

The Fig.9 is The Explore Problems Page is designed to allow users to explore and discover numerous problems available on the platform. The users can use filters and search options to find relevant problems, making it easy to explore and interact within the platform.

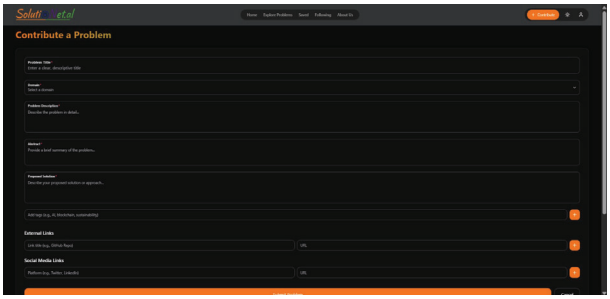

Fig. 10: Problem Contribution Page

The Fig.10 is The Problem Contribution Page provides the details needed to contribute a new problem to the SolutioNet.al platform.

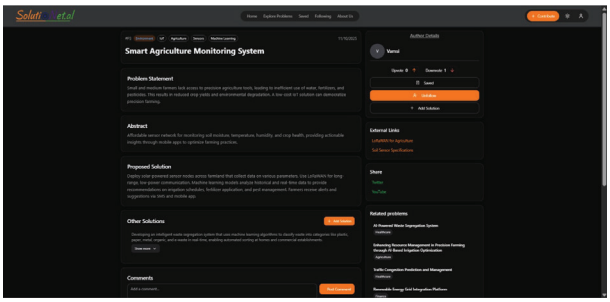

Fig. 11: Problem Details Page

The Fig.11 is the problem Details Page shows the entire details of a particular problem on SolutioNet.al.

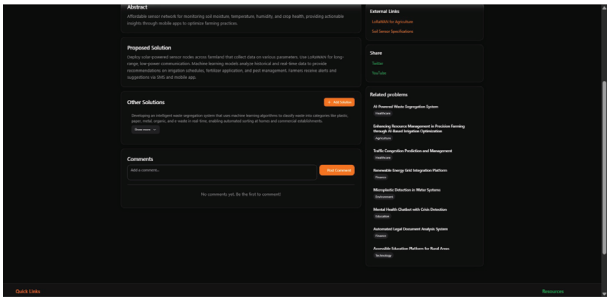

Fig. 12: Problem Details Page

The Fig.12 is the problem Details Page shows the entire details and information that are required to solve a particular problem on SolutioNet.al.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 07 , July - 2026

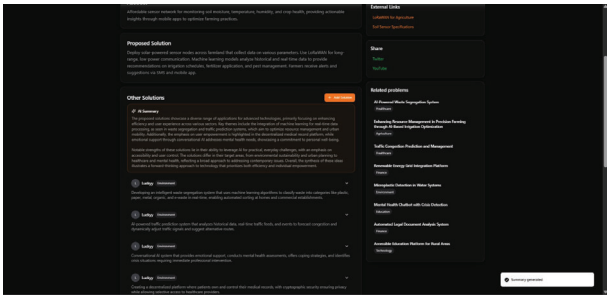

Fig. 13: AI Summarization Page

The Fig.13 is AI Summarization Page represents the feature that can summarize the Information of multiple solutions which are previously posted for a particular problem and provides a brief summary of each solution.

VIII. Conclusion

In this paper, the SolutioNet.al is introduced as an open community platform intended to fill the void between the stages of problem detection and collaborative idea generation in the problem-solving process. The novelty of this proposed platform consists in the accent on the capture and development of ideas about real-world problems in their unrefined form compared to the current platforms that target the implementation of already established ideas.

This system is intended to facilitate the submission, exploration, and analysis of problem descriptions, including functions for problem identification, classification, voting, and bookmarking. Moreover, AI-powered text summarization and semantic similarity-based recommendations make using the service more convenient.

The implementation proves that the system can maintain its efficiency and performance in the presence of interactive and intelligent capabilities. From experimentation, it is shown that there is minimum latency in the process and the response time for the AI functions is tolerable.

All in all, the presented approach is beneficial in creating an innovation ecosystem that will be inclusive and collaborative because rather than having a problem-solving focus, the emphasis lies in identifying problems through collaboration.

IX. Future Scope

Possible future improvements may include the inclusion of better moderation tools and spam prevention capabilities, a chatbot that is able to analyze user inputs and give ideas about solving certain issues through discussions, and a designated area for displaying necessary pre-requisites and learning materials. Such improvements would be able to make the platform more reliable and useful for its users.

References

[1] Liang, H., Prabhu, R., Farzanepour, S., Rajeev, S., and Brown, C. (2025). ResearchBot: Linking Academic Research with Practical Programming Communities. arXiv. https://arxiv.org/abs/2407.02643

[2] Weinhardt, C., Gau, M., Greif-Winzrieth, A., Maedche, A., and vom Brocke, J. (2025). Creating an open research framework for cooperative issue solving is one way to include citizen scientists. Electronic Markets, 35 (12). https://doi.org/10.1007/s12525-025-00757-z

[3] In 2023, Tafreshi, S., Atasoy, O., Huang, Y., Buchbinder, R., and Sinha, A. Research that has an impact: Where should we go and how far have we come? Association for Information Systems Journal, 24(6), 1482-1504. https://doi.org/10.17705/1jais.00816

[4] Anderson, A., Kleinberg, J., Huttenlocher, D., Leskovec, J. (2012). Using online competitions to raise community awareness. 21st World Wide Web International Conference Proceedings (pp. 1–10). ACM. https://doi.org/10.1145/2187836.2187839

[5] Wattanakriengkrai, S., Chinthanet, B., Hata, H., Kula, R. G., Treude, C., Guo, J., Matsumoto, K. (2022). Academic article connections in GitHub repositories: Evolution, traceability, and public access. Systems and Software Journal, 183, 111117. https://doi.org/10.1016/j.jss.2021.111117

[6] Jamali, H., Dascalu, S. M., & Harris, F. C., Jr. (n.d.). Fostering joint innovation: A global online platform for ideas sharing and collaboration. University of Nevada, Reno.

[7] Martinez, R., Afanador, L., Bernert, M., Bertino, S., D'Este, P., Fecher, B., Fiedler, M., Gülker, S., Hagenhoff, S., Heller, L., Menter, M., Mönch, C., Neumann, L., Palmen, R., Poetz, M. K., Rimmert, C., Rüffin, N., Simmchen, J., Wöhlert, R. (2022). An approach to collaborative conceptualization in the realm of open innovation in science research. Innovation and Industry, 29(2), 136-185. https://doi.org/10.1080/13662716.2021.1977827

[8] Kulkarni, M., Jadhav, A., Koli, M., Magar, A., Navale, A., "A collaborative platform for university and college Students Project," International Journal of Innovative Research in Technology (IJIRT), Vol. 11, Issue 8, Pages 2478-2483, January 2025. https://ijirt.org/article?manuscript=172169

[9] Chen, Q., Magnusson, M., Björk, J. (2022). investigating the impact of information sharing about problems and solutions in internal crowdsourcing. Knowledge Management Journal, 26(8), 194-217. https://doi.org/10.1108/JKM-10-2021-0769

[10] Helbing, D. (2012). creating huge scientific challenges to accelerate scientific discoveries. Preprint arXiv:1208.3883, European Physical Journal. https://arxiv.org/abs/1208.3883