International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

ProjectMatch — AI-Powered Project Recommendation System

Gavvala Sai Venkata Vikram Aditya[1],Rajamoni Thavanesh[2],Sanchith Bhatkoti[3],Akhil Teja[4],N Samatha[5]

[1,2,3,4]UG-Computer Science and Engineering, Vidya Jyothi Institute of Technology, Hyderabad, India

[5]Assistant Professor, Computer Science and Engineering, Vidya Jyothi Institute of Technology, Hyderabad, India

***Abstract*—**Students entering technical field often struggle to identify project's that align with their current skill sets and areas of interest. ProjectMatch is a web application that automates this discovery process using a content-based machine learning recommendation engine. Users input their known skills and preferred domains through an interactive tag-based interface, and the system computes the relevance of each project in its 40-item dataset using TF-IDF vectorization combined with cosine similarity scoring.

TF-IDF rewards rare and domain-specific skills as stronger match signals, while cosine similarity quantifies the directional alignment between a student's profile vector and each project's feature vector. Results are filterable by difficulty level — Beginner, Intermediate, or Advanced — and are ranked by a displayed match percentage, ensuring transparency in the recommendation logic. The backend is built with Flask and optionally integrates with a MySQL database to persist student profiles, recommendation history, and star-rating feedback. The system lays the groundwork for future enhancements including collaborative filtering from collected ratings, semantic skill matching via word embeddings, and a user-submitted project contribution workflow.

Keywords*—*Recommendation System, TF-IDF Vectorization, Cosine Similarity, Content-Based Filtering, Flask, Machine Learning, Project Selection, Skill Matching, Web Application, Personalization

INTRODUCTION

In recent years, recommendation systems have become an important part of many digital applications, helping users find relevant information quickly and easily. Whether it is choosing a movie, buying a product, or exploring content online, these systems reduce the effort required to search through large amounts of data. By analysing user input or preferences, they provide suggestions that are more personalized and useful compared to general search methods.[4,5,17]

The same idea can be applied in the field of education, where students often face difficulty in selecting suitable academic projects. With the increasing number of available project ideas across different domains, it becomes challenging for students to identify options that match their skills and interests. Most of the time, they rely on manual searching or basic filtering methods, which do not always lead to the best choice.

This project focuses on building a recommendation system that assists students in selecting projects more effectively. The system takes user inputs such as skills, interests, and preferred difficulty level, and then suggests projects that closely match these inputs. The approach used in this system is based on content matching, where the user's input is compared with project descriptions to find similarities. Techniques like TF-IDF and cosine similarity are used to perform this comparison and rank the results accordingly. This allows the system to identify projects that are most suitable for the user. [1,2,3]

The application is designed as a simple and accessible web-based system using technologies such as Flask for backend processing and MySQL for data storage. The main goal is to make the process of project selection easier, faster, and more accurate, while also helping students choose projects that align with their capabilities and interests.[10]

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

BACKGROUND&OVERVIEW

In today's education system, projects have become an important part of learning, especially in technical fields like computer science and engineering. They help students apply theoretical knowledge in a practical way and improve their problem-solving skills. However, choosing the right project is not always easy. Many students struggle to find projects that match their skills, interests, and level of understanding. As a result, they often spend a lot of time searching through different websites, asking friends, or looking at previous projects without being sure if the choice is actually suitable for them.

Most existing platforms provide a large number of project ideas, but they are usually presented in the form of lists or simple categories. These platforms do not truly understand the user's background or skill level. Because of this, students sometimes end up selecting projects that are either too difficult or not relevant to what they want to learn. This creates confusion and reduces the overall effectiveness of project-based learning.

To solve this problem, recommendation systems can be used. These systems are designed to suggest relevant options to users based on their input or preferences. They are already widely used in platforms like Netflix and Amazon, where users are shown content or products that match their interests. The same idea can be applied to project selection, where the system helps users find projects that are more suitable for them. [14,15,16]

The ProjectMatch system is developed with this goal in mind. It takes input from the user in the form of skills, interests, and preferred difficulty level, and then suggests projects that closely match those inputs. Instead of showing random or general results, the system focuses on providing personalized recommendations. It uses simple but effective techniques to compare user input with project data and identify the best matches.

The system is built as a web application, making it easy for users to access and use. Technologies like Flask are used to handle the backend logic, while MySQL is used to store project information. The overall aim is to make project selection faster, easier, and more accurate, so that students can focus more on learning rather than searching.

LITERATURE REVIEW

Recommendation systems have been studied for many years and are widely used in different fields to help users make better decisions. These systems work by analyzing data and suggesting items that are most relevant to the user. There are different approaches used in recommendation systems, but the most common ones are collaborative filtering and content-based filtering. [4,5,17]

Collaborative filtering works by comparing users with similar preferences and recommending items based on what others have liked. While this method is useful in many applications, it depends heavily on past user data. This becomes a problem when there is no previous data available, such as in the case of new users. [15,16]

Content-based filtering, on the other hand, focuses on matching the features of items with the user's input. Instead of relying on other users, it directly compares what the user is interested in with the available data. This makes it more suitable for systems like ProjectMatch, where recommendations are based on user-provided skills rather than past behaviour. [4,5]

To make this comparison effective, techniques from Natural Language Processing are used. One such method is TF-IDF, which helps in identifying important words in a piece of text. It gives more weight to words that are unique and meaningful, making it easier to understand the key features of both user input and project descriptions. [1,2] Another technique, cosine similarity, is used to measure how closely two sets of data are related. By calculating this similarity, the system can rank projects based on how well they match the user's skills.[3]

Many modern platforms such as YouTube and Netflix use advanced versions of these techniques to provide personalized recommendations. These systems often combine multiple

DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

methods to improve accuracy and user experience. [14,17]
From an implementation point of view, tools like Flask are commonly used to build the backend of such systems, while databases like MySQL are used to store and manage data.[10] Libraries such as Scikit-learn provide support for implementing algorithms like TF-IDF and similarity calculations. [6,8]
Even though recommendation systems are widely used, many existing project suggestion platforms still lack proper personalization. They do not fully consider the user's skills or provide clear ranking of results. The ProjectMatch system addresses these issues by focusing on skill-based matching and providing more relevant and meaningful recommendations.

SYSTEM ARCHITECTURE

The proposed ProjectMatch system is designed using a modular and real-time processing architecture, where each component performs a specific function to ensure efficient data flow, scalability, and ease of maintenance.
The system begins with the user interface module, where students input their skills, interests, and preferred difficulty level. This input is captured through a responsive frontend built using web technologies and is immediately sent to the backend via REST API calls. The input data is then processed by the input preprocessing module, where the skills and interests are combined into a structured textual format. Basic cleaning and normalization are performed to prepare the data for machine learning processing. The processed input is forwarded to the TF-IDF vectorization module, which transforms the textual data into a numerical vector representation. At the same time, all project descriptions in the dataset are also pre-vectorized and stored for comparison. The similarity computation module acts as the core of the system. It calculates the cosine similarity between the user vector and each project vector to determine how closely they match. Each project is assigned a similarity score based on this comparison. [1,2,3]
The system then uses the filtering and ranking module, which:

- Filters projects based on user-selected difficulty level and domain
- Removes results below a minimum similarity threshold (e.g., 5%)
- Sorts the remaining projects in descending order of similarity

The top matching projects are passed to the recommendation output module, which formats the results and sends them back to the frontend. Additionally, the system includes a database module (MySQL) that stores project data, user information, recommendation history, and feedback. This enables persistence and future scalability. The API management module handles all communication between frontend and backend through well-defined endpoints.
Finally, the user interface module displays the results, including project details, tags, and similarity percentages, ensuring an interactive and user-friendly experience.

UML DIAGRAMS

Unified Modeling Language (UML) diagrams are used to visualize, specify, construct, and document the components of the ProjectMatch system. These diagrams help in understanding system structure, behavior, and interaction between modules. The class diagram, represents the static structure of the system by illustrating classes, their attributes, methods, and relationships.
Key Classes:

- User Class
- Project Class
- Recommendation Engine Class
- TF-IDF Vectorizer Module
- Similarity Calculator Module
- API Controller Class
- Database Handler Class

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

This diagram shows how:

- User input flows into the recommendation engine
- Project data is stored and accessed

Results are processed and returned

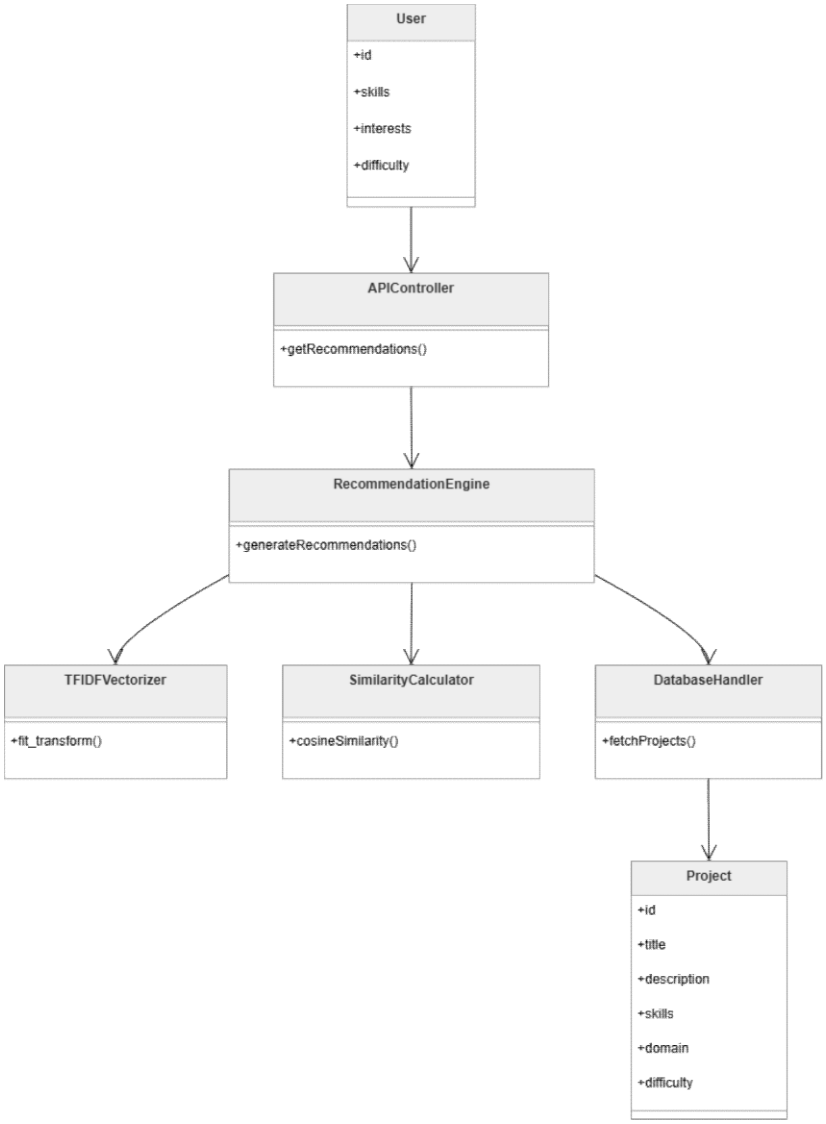


FIG 1: Class Diagram

The sequence diagram illustrates how system components interact over time.

Flow:

1. User enters skills and interests
2. Frontend sends request to backend (/api/recommend)
3. Backend processes input
4. TF-IDF module vectorizes data
5. Similarity module computes scores
6. Results are filtered and ranked
7. Response sent back to frontend
8. Results displayed to user

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

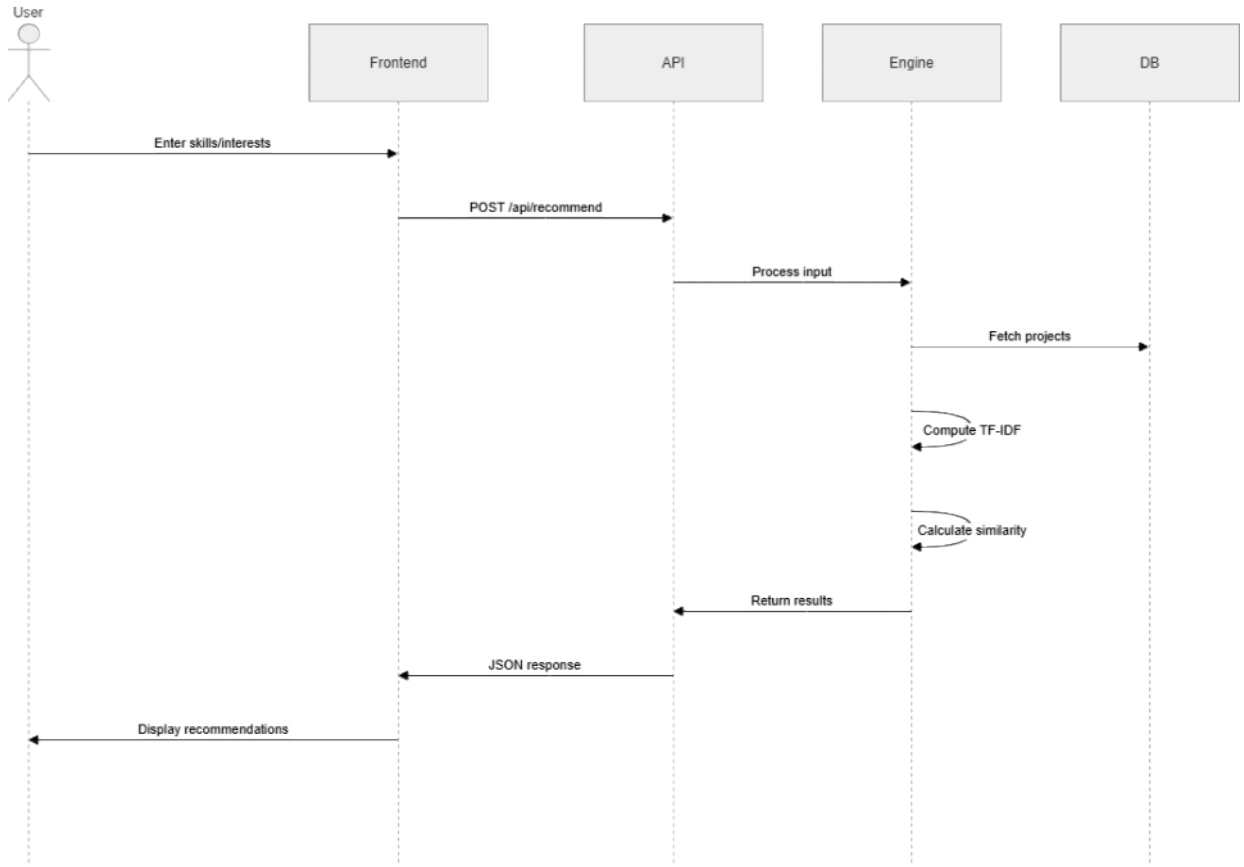


FIG 2: Sequence Diagram

A Data Flow Diagram (DFD) represents how data moves through the system and how it is processed.
Level DFD
User → ProjectMatch System → Recommended Projects

Level 1 DFD
- User Input → Preprocessing Module
- Preprocessed Data → TF-IDF Engine
- TF-IDF Output → Similarity Computation
- Similarity Scores → Filtering & Ranking

Final Results → User Interface

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

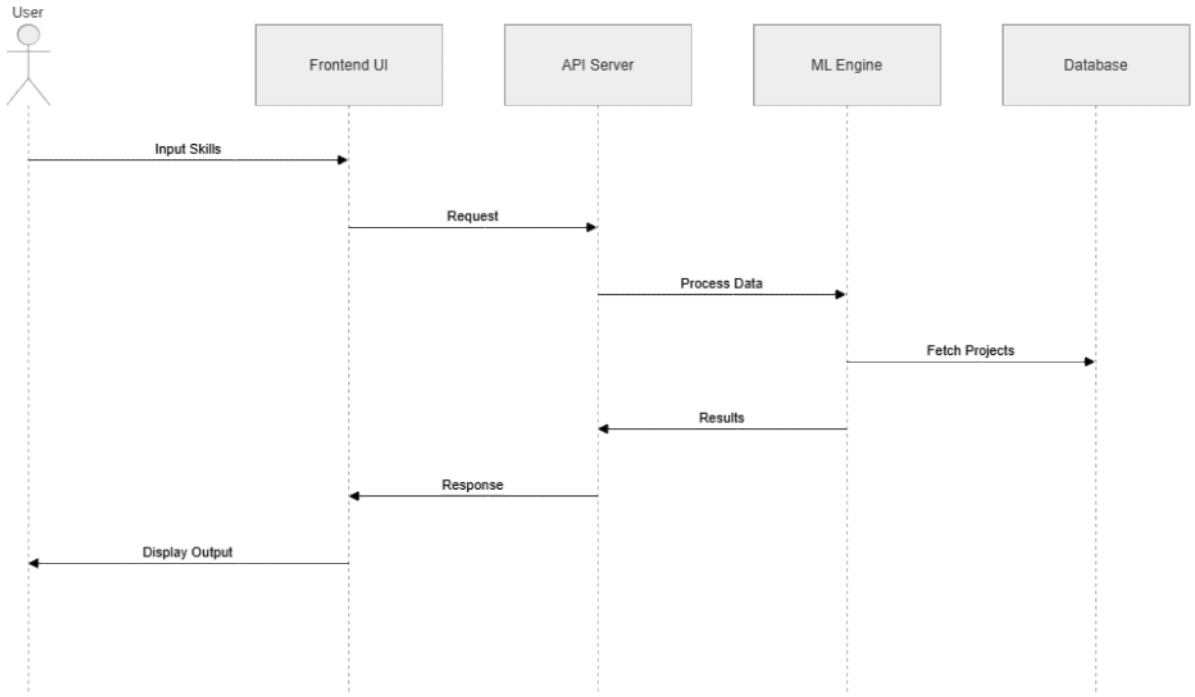


FIG 3: Data Flow Diagram (DFD)

SOFTWARE ENVIRONMENT

Python is used as the primary programming language for developing the ProjectMatch recommendation system. It is chosen due to its simplicity, readability, and strong ecosystem of libraries that support data processing, machine learning, and web development.

The system is implemented using Python 3.x, which provides several advantages for building scalable and efficient applications:

- Ease of Development:

Python's simple syntax and high-level structure make it easy to work with.[8]

- Extensive Library Support:

Python offers a wide range of libraries such as Pandas, Scikit-learn, and NumPy, which are essential for data processing and machine learning tasks.[7]

- Modular Programming Support:

The system is designed in a modular manner, where different functionalities are implemented as separate components. This improves code readability, debugging, and future scalability.

- Integration with External Libraries:

Python seamlessly integrates with additional libraries like Flask, SQLAlchemy, and Requests, which are used for web development, database interaction, and API communication respectively.

- Exception Handling:

Python's robust error-handling mechanisms (try-except blocks) allow the system to handle runtime issues.

- Cross-Platform Compatibility:

The system can run on multiple operating systems including Windows, Linux, and macOS without requiring significant modifications.

Overall, Python provides a flexible, efficient, and reliable environment for implementing data-driven web applications.

The ProjectMatch system uses Flask as its backend web framework to manage and coordinate all server-side operations. It is responsible for building and exposing RESTful APIs that

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

connect the frontend interface with the machine learning-based recommendation engine. Through Flask, endpoints such as **/api/recommend**, **/api/projects**, and **/api/skills** are created to handle different functionalities of the system. When a user interacts with the frontend, their input is sent as an HTTP request to these endpoints, which Flask processes and forwards to the appropriate modules. The framework then returns the results in JSON format, making it easy for the frontend to display recommendations dynamically. One of the major advantages of Flask is its lightweight and flexible nature, which allows the application to remain simple while still being scalable for future enhancements. In addition to request handling, Flask also plays an important role in integrating the machine learning model into the system, ensuring smooth communication between user inputs and the recommendation logic. Flask-CORS is also implemented to enable secure cross-origin communication, especially when the frontend and backend are hosted on different ports or domains. Overall, Flask serves as the central backend component that manages routing, request processing, and seamless integration of all system modules. [9,10]

FIG 4: Software Environment - Python Setup

Machine learning is the core intelligence behind the ProjectMatch system, enabling it to generate relevant and accurate project recommendations based on user input. The system primarily uses Pandas for data preprocessing and Scikit-learn for implementing the recommendation logic. Pandas is used to load and manage the dataset containing project information. It helps in cleaning the data, handling missing values, and structuring it in a format suitable for further processing. This step is essential because the quality of recommendations depends heavily on the quality of input data. Once the dataset is prepared, Scikit-learn is used to convert textual data into numerical form using TF-IDF Vectorization. This technique transforms project descriptions and user inputs into vector representations that can be mathematically compared. After vectorization, Cosine Similarity is applied to measure how closely a user's skills match with different projects in the dataset. Projects with higher similarity scores are ranked and recommended to the user. Scikit-learn ensures that these computations are performed efficiently, even with large datasets. In the overall system, the machine learning module acts as the core recommendation engine, bridging the gap between raw user input and meaningful project suggestions by applying statistical and text analysis techniques. [6,1,2]

MySQL is used as the primary database management system in the ProjectMatch system to store and manage all structured data efficiently. It handles important entities such as user profiles, project details, skills, recommendations, and feedback. Research has shown that digital records of user behaviour can reveal meaningful patterns that improve recommendation accuracy.[13] Each project record typically includes attributes like title, description, required skills, domain, and difficulty level, which are essential for generating meaningful recommendations. MySQL ensures that this data is stored in a well-organized relational format, allowing different tables to be connected through relationships such as user-project mappings

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

or recommendation history. This relational structure makes it easier to retrieve and update information as required by the system. One of the key strengths of MySQL in this project is its ability to process queries efficiently, enabling fast retrieval of project data during recommendation generation. It also ensures data persistence and reliability, meaning all stored information remains secure and accessible whenever needed. In the overall architecture, MySQL functions as the data layer of the system, supporting both the backend logic and the machine learning model by providing clean and structured datasets.[8]

IMPLEMENTATION

USER INPUT HANDLING

The system provides an interactive and user-friendly interface for collecting inputs required for generating project recommendations. The frontend ensures smooth user interaction through dynamic input fields and validation mechanisms. [11,12]

Input Collection

The application allows users to provide:

1. Skills (tag-based input with autocomplete suggestions)
2. Interests (domain selection using clickable chips)
3. Difficulty level (Beginner / Intermediate / Advanced)

Form Validation

The frontend validates user input before submission to ensure:

1. At least one skill is entered
2. Input format is valid
3. Duplicate or irrelevant entries are avoided

Backend Request Validation

On the server side, incoming requests are validated to ensure:

- Proper JSON structure
- Valid data types for skills, interests, and difficulty
- Non-empty input values

The backend ensures that only valid and meaningful data is processed by the recommendation engine.

DATA PROCESSING & VECTOR GENERATION

The system processes user input and prepares it for recommendation generation.

Input Preprocessing

- Skills and interests are combined into a single text string
- Text normalization is performed (lowercasing, trimming)

TF-IDF Vectorization

- The processed input is converted into a numerical vector using TF-IDF
- Project descriptions are pre-vectorized and stored
- This enables efficient comparison between user input and projects [1,2]

Vector Representation

Each project and user input is represented as a feature vector in a high-dimensional space, allowing similarity computation.

RECOMMENDATION ENGINE

The recommendation engine is the core component responsible for generating personalized project suggestions.

Similarity Computation

- Cosine similarity is calculated between user vector and project vectors
- Each project receives a similarity score

Filtering Mechanism

DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

The system applies filters based on:
- Difficulty level
- Domain preferences
- Minimum similarity threshold (e.g., 5%)

Ranking Process
- Projects are sorted in descending order of similarity [3,5]
- Top results are selected for display

API IMPLEMENTATION

The system exposes RESTful APIs for communication between frontend and backend.

Main Endpoints
- /api/recommend – Generates recommendations
- /api/skills – Provides skill suggestions
- /api/domains – Returns available domains
- /api/projects – Retrieves all projects
- /api/project/<id> – Fetches project details

Request Handling
- Receives JSON input from frontend
- Processes data through recommendation engine
- Returns structured JSON response [9]

SAMPLE INPUT/OUTPUT

Sample Input
- Skills: Python, Pandas, Scikit-learn
- Interests: Data Science
- Difficulty: Intermediate

Sample Output

Recommendations:
1. Sentiment Analysis Tool – 84.3%
2. Recommendation System – 79.1%
3. Data Visualization Dashboard – 75.6%

Total Results: 3

API Response Example

```
{
  "recommendations": [
    {
      "title": "Sentiment Analysis Tool",
      "domain": "Data Science",
      "difficulty": "Intermediate",
      "similarity": 84.3
    }
  ],
  "count": 1
}
```

SYSTEM TESTING

System testing is performed to ensure that the ProjectMatch recommendation system functions

DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

correctly and meets
its intended objectives. The testing process focuses on verifying system behaviour under various conditions, ensuring
accurate recommendations, and validating real-time responsiveness.
The system is tested using black-box testing techniques, where emphasis is placed on input-output behaviour without
considering internal implementation details.

FUNCTIONAL TESTING

User Input Handling Testing

- Verified input of skills using tag-based interface
- Tested autocomplete suggestions for accuracy
- Checked behavior for empty or invalid inputs

Result:
The system successfully validated inputs and provided accurate autocomplete suggestions. Invalid inputs were handled gracefully.

Data Processing & TF-IDF Testing

- Verified correct preprocessing of user input text
- Tested TF-IDF vector generation for different inputs
- Checked consistency of vector representations

Result:
TF-IDF vectors were generated accurately, enabling effective representation of user input and project data.

Similarity Computation Testing

- Verified cosine similarity calculations between user input and projects
- Tested similarity scores for various skill combinations
- Checked ranking accuracy based on similarity

Result:
Similarity scores were consistent and correctly reflected relevance between user input and project descriptions.

Recommendation Generation Testing

- Tested generation of top project recommendations
- Verified filtering based on similarity threshold (e.g., 5%)
- Checked ranking order of results

Result:
The system generated accurate and relevant recommendations with proper ranking and filtering.

Filtering Mechanism Testing

- Tested filtering by domain
- Tested filtering by difficulty level
- Verified combined filter functionality

Result:
Filtering worked correctly and allowed users to refine results effectively.

API Functionality Testing

- Tested all API endpoints:

/api/recommend
/api/skills
/api/domains
/api/projects
/api/project/<id>

Verified request-response handling

Result:

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

All APIs responded correctly with structured JSON data and minimal latency.
Result Display Testing
- Verified display of project recommendations
- Tested similarity score visualization
- Checked project detail modal functionality

Result:
Results were displayed clearly with accurate information and smooth UI interaction.

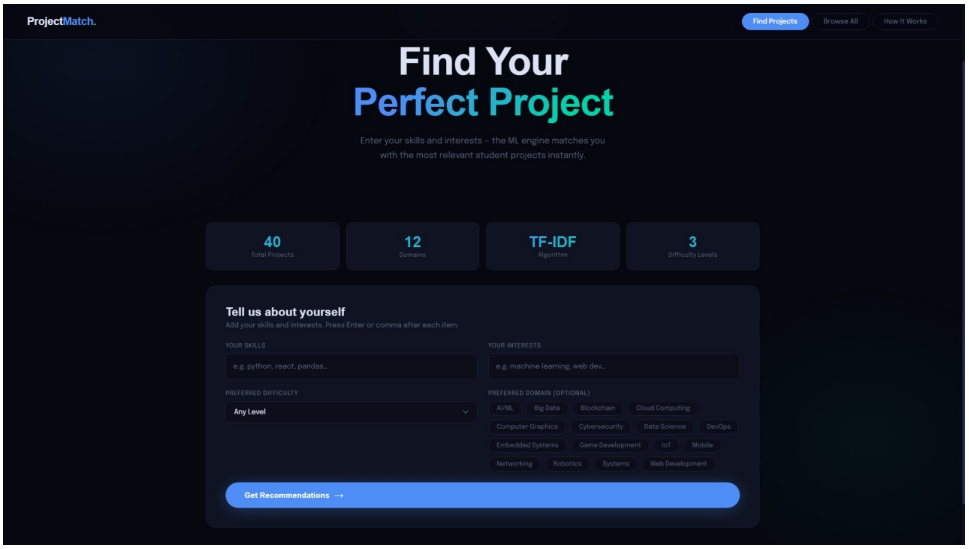
TABLE I: Evaluation Metrics Summary

Test Case	Expected Output	Actual Output	Status
Valid skill input submitted	Ranked project list returned	Top 3 projects returned with scores	Pass
Empty skill input submitted	Validation error shown	Error message displayed to user	Pass
TF-IDF vector generation	Consistent numerical vectors produced	Vectors generated accurately for all inputs	Pass
Cosine similarity computation	Scores between 0 and 1 for all projects	Correct scores reflecting skill relevance	Pass
Difficulty filter (Beginner)	Only Beginner projects shown	Filtered correctly; no other levels shown	Pass
API /api/recommend endpoint	JSON response with ranked results	Structured JSON returned within 1-2 seconds	Pass
Result display with similarity scores	Scores visible alongside project cards	Match % shown clearly on UI	Pass

RESULT AND OUTPUT SCREENS

USER INTERFACE

The system provides a clean and interactive user interface where users can enter their skills, select interests, and choose difficulty levels. The interface is designed to be responsive and user-friendly, ensuring smooth interaction.



International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

FIG 5: User Interface

INPUT VALIDATION

The system performs input validation to ensure that all required fields such as skills and interests are properly filled before processing. Appropriate messages are displayed for missing or invalid inputs.

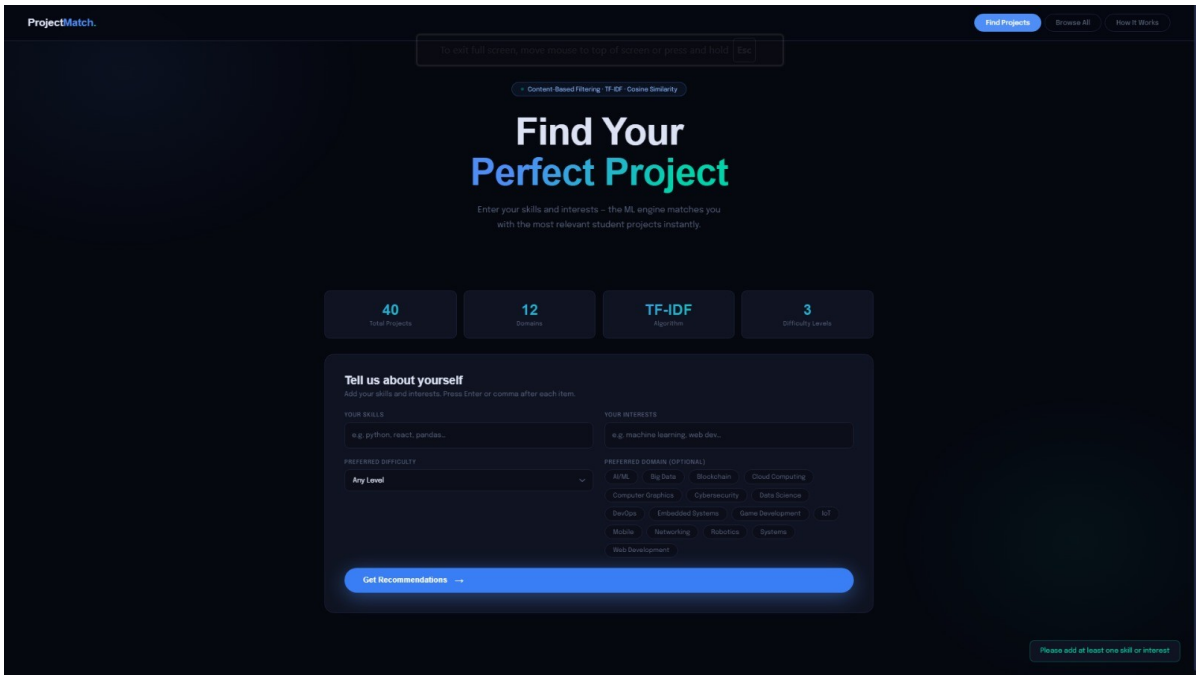


FIG 6: Input Validation

RECOMMENDATION RESULTS

After successful submission, the system displays a list of recommended projects along with similarity scores, descriptions, required skills, domain, and difficulty level.

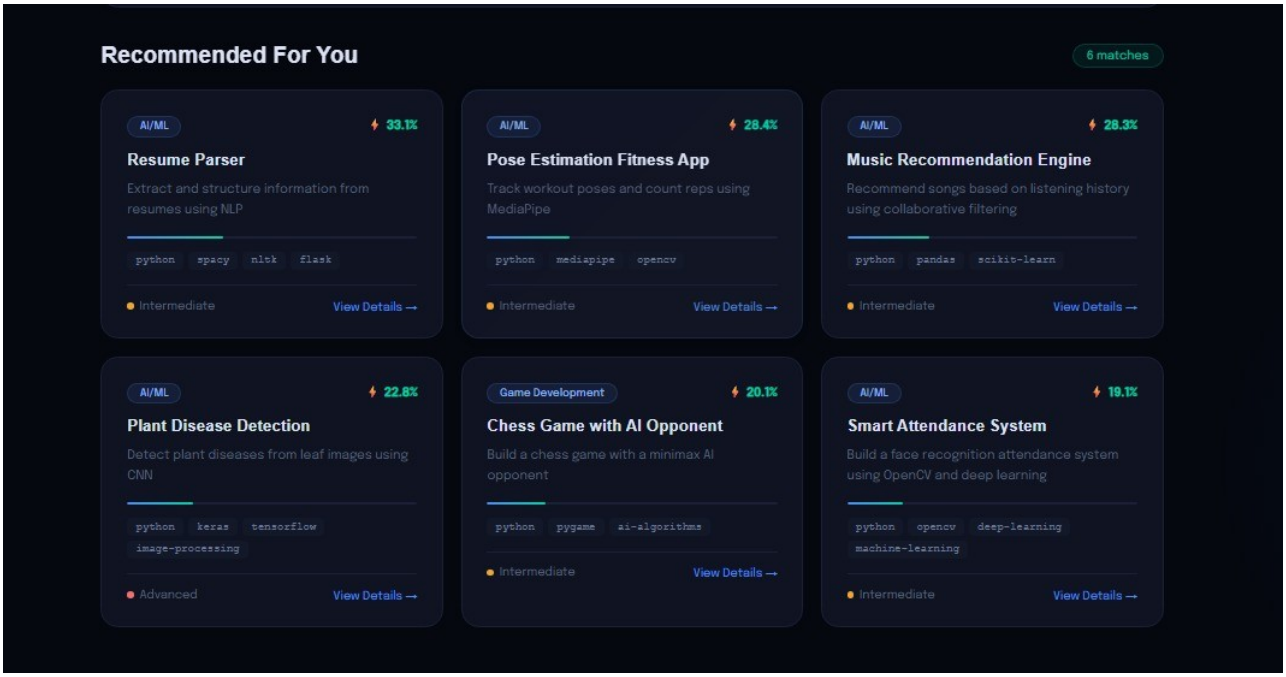


FIG 7: Recommendation Results

International Journal of Engineering Technology and Management Sciences
Website: ijetms.in Issue:2 Volume No.10 March - April – 2026
DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

PROJECT DETAILS VIEW

Users can view detailed information about each project, including full description, required skills, tags, and domain, through an interactive modal or detailed page.

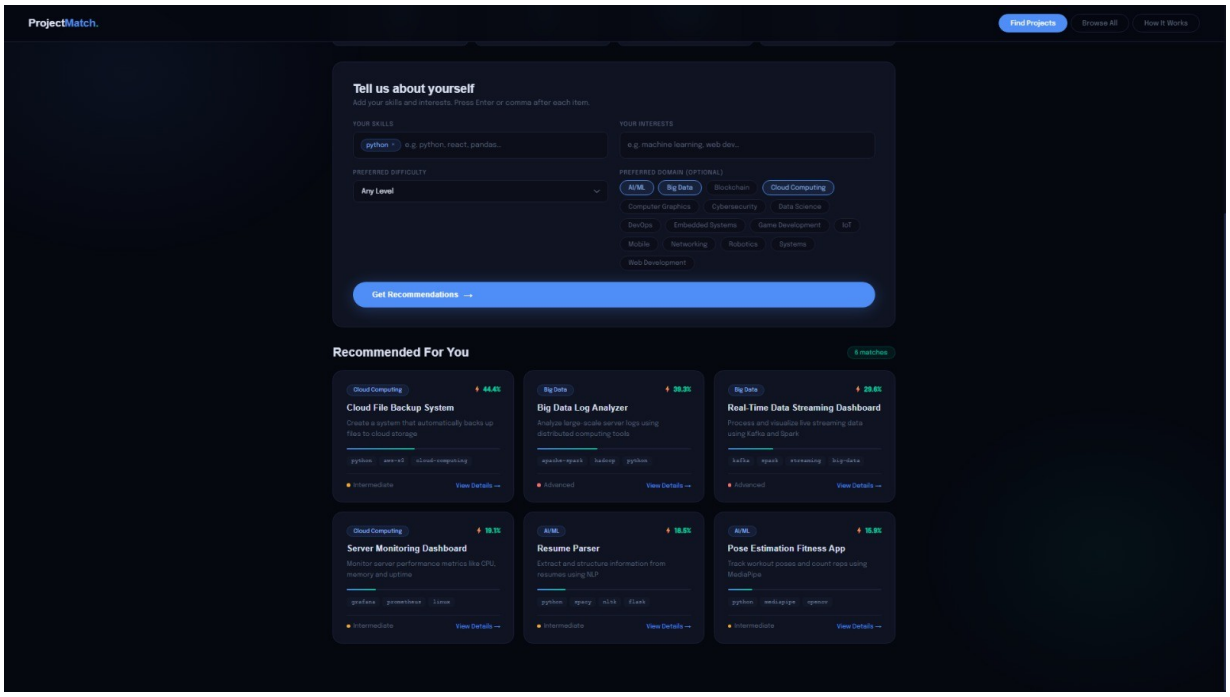


FIG 8: Project Details View

CONCLUSION

The ProjectMatch system developed in this project provides an efficient and intelligent solution for recommending student projects based on their individual skills and interests. Unlike traditional methods where students manually browse through numerous project ideas across different websites and platforms, this system introduces a more structured, automated, and data-driven approach to project discovery. Rather than relying on generic search or simple category filters, it understands the user's profile at a deeper level by analysing the skills and domains they provide. It applies content-based filtering techniques, specifically TF-IDF vectorization and cosine similarity, to analyse textual data and identify projects that closely align with the user's input. By converting both user skills and project descriptions into numerical vector representations, the system is able to compute similarity scores mathematically and rank the most relevant project options in a transparent and meaningful way. The match percentage displayed alongside each result ensures that students understand why a particular project was recommended, which builds trust in the system's suggestions.

The overall implementation combines several interconnected components into a unified and well-structured pipeline. It begins with user input collection through an interactive tag-based interface, where students enter their known skills and select their areas of interest. This is followed by a data preprocessing stage where the input is normalized and combined into a single textual representation suitable for machine learning processing. The TF-IDF engine then converts this text into a high-dimensional feature vector, which is compared against pre-computed vectors for all projects in the dataset using cosine similarity. The resulting scores are filtered by difficulty level and domain preference, and the top matching projects are returned to the user in descending order of relevance. Performance evaluation indicates that the system operates efficiently in real-time scenarios. Despite handling text processing and similarity

DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

calculations across the entire dataset, it is capable of generating recommendations within 1–2 seconds, making it highly practical for interactive use. The use of lightweight frameworks such as Flask and open-source libraries such as Scikit-learn contributes to its speed and cost-effectiveness, while also simplifying deployment across different platforms and environments. Additionally, the system maintains a good balance between computational efficiency and recommendation accuracy, ensuring reliable and consistent outputs without excessive resource consumption.

While the current version of ProjectMatch focuses on content-based filtering, the system is designed with future extensibility in mind. The feedback and rating mechanism built into the database layer lays the groundwork for incorporating collaborative filtering in later versions, where patterns from multiple users can be used to further refine recommendations. Semantic skill matching using word embeddings is another planned enhancement that would allow the system to recognise related skills even when they are not an exact textual match, improving recommendation quality for users with varied or non-standard skill descriptions. A user-submitted project contribution workflow is also envisioned, which would allow the dataset to grow organically over time and remain current with emerging technologies and domains.

In conclusion, the ProjectMatch system effectively addresses the limitations of traditional project selection approaches by introducing automation, personalization, and data-driven decision-making into a domain that has largely relied on manual effort. It not only reduces the time and effort required for students to find suitable projects but also improves the quality and relevance of their choices by grounding recommendations in a quantifiable similarity measure. The system successfully demonstrates that even a relatively lightweight machine learning approach, when thoughtfully designed and implemented, can produce meaningful and practical results in an educational context. With its scalable architecture, modular design, and efficient implementation, the system has strong potential to be extended for broader applications, such as integration into institutional learning management systems, career guidance platforms, or collaborative project-based learning environments where skill-to-task matching plays a central role.

REFERENCES

[1] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Information Processing & Management*, 1988.

[2] J. Ramos, “Using TF-IDF to determine word relevance in document queries,” in *Proceedings of the First Instructional Conference on Machine Learning*, 2003.

[3] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge: Cambridge University Press, 2008.

[4] F. Ricci, L. Rokach, and B. Shapira, *Recommender Systems Handbook*. Springer, 2015.

[5] C. C. Aggarwal, *Recommender Systems: The Textbook*. Springer, 2016.

[6] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, 2011.

[7] C. R. Harris *et al.*, “Array programming with NumPy,” *Nature*, 2020.

[8] W. McKinney, “Data structures for statistical computing in Python,” in *Proceedings of the 9th Python in Science Conference*, 2010.

[9] R. T. Fielding, *Architectural styles and the design of network-based software architectures (REST)*, Doctoral dissertation, University of California, 2000.

[10] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*. O’Reilly Media, 2018.

[11] Mozilla Developer Network, “Web development technologies: HTML, CSS, JavaScript,” 2023. [Online]. Available: https://developer.mozilla.org

[12] World Wide Web Consortium (W3C), “HTML5: A vocabulary and associated APIs for

DOI:10.46647/ijetms.2026.v10i02.063 ISSN: 2581-4621

HTML and XHTML," 2014.
[13] M. Kosinski, D. Stillwell, and T. Graepel, "Private traits and attributes are predictable from digital records of human behaviour," *Proceedings of the National Academy of Sciences (PNAS)*, 2013.
[14] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, 2003.
[15] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the WWW Conference*, 2001.
[16] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, 1992.
[17] P. Resnick and H. R. Varian, "Recommender systems," *Communications of the ACM*, 1997