International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 11 | Nov - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Nest – Empowering Campus Startups through Data-Driven Intelligence and AI-Powered Decision Support

Pranoti Chakwate[1], Siddhi Borawake[2], Aditi Mane[3], Priya Kumari[4] , Dr. S. R. Ganorkar[5]

[1]*Student, Information Technology Department, Sinhgad College of Engineering, Pune, India*
[2]*Student, Information Technology Department, Sinhgad College of Engineering, Pune, India*
[3]*Student, Information Technology Department, Sinhgad College of Engineering, Pune, India*
[4]*Student, Information Technology Department, Sinhgad College of Engineering, Pune, India*
[5]*Head of Department, Information Technology Department, Sinhgad College of Engineering, Pune, India*

Abstract - College students often have innovative business ideas but lack accessible platforms to transform them into successful ventures. To address this challenge, Nest has been developed as a full-stack web platform that empowers campus startups through data-driven intelligence and AI-powered decision support. The system integrates entrepreneurship, collaboration, and artificial intelligence within a unified campus ecosystem. It features an online marketplace where student entrepreneurs can showcase and sell their products or services, supported by AI-driven recommendation algorithms based on TF-IDF and cosine similarity to personalize user experiences. A Linear Regression model predicts sales trends for data-driven decision-making, while a Naïve Bayes classifier performs sentiment analysis on customer reviews to enhance feedback interpretation. Furthermore, a Graph-Based Collaboration Network built using NetworkX connects students with complementary skills, fostering teamwork and co-founder discovery. The platform utilizes the MERN stack with secure authentication and integrated payment processing. *Nest* provides a scalable, intelligent, and collaborative environment that promotes innovation and entrepreneurial growth within college communities.

Key Words: AI, Student Entrepreneurship, Recommendation System, Linear Regression, Sentiment Analysis, Graph Network, MERN Stack, Predictive Analytics

1. INTRODUCTION

1.1 MOTIVATION

The development of Nest is motivated by the need for a dedicated digital ecosystem that supports student-led ventures. Nest provides an AI-driven platform where students can launch and manage businesses, receive personalized recommendations, forecast sales, analyze customer sentiment, and collaborate with peers. By integrating these features into a single accessible system, Nest aims to empower students, promote data-driven entrepreneurial growth, and strengthen the startup culture within campuses.

1.2 PROBLEM STATEMENT

Student entrepreneurship promotes innovation and practical learning, yet many college students struggle to establish and manage ventures due to limited resources, lack of mentorship, and minimal market exposure. Existing platforms are often complex, costly, and not tailored to student needs, and they typically lack data-driven or AI-based decision support. The core problem is the absence of an affordable, student-centric digital ecosystem that integrates marketplace features, business management tools, and AI-powered insights to support the launch and growth of campus-based startups.

1.3 OBJECTIVES

1. Analyze existing literature and current student entrepreneurship platforms.
2. Develop a full-stack web application supporting product listing, transactions, and venture management.
3. Implement AI techniques including content-based recommendations, linear regression for sales forecasting, and Naïve Bayes for sentiment analysis.
4. Construct a graph-based collaboration network using NetworkX to recommend potential co-founders or partners.
5. Ensure secure authentication, real-time data processing, and a responsive interface using React.js, Node.js, Express, and MongoDB.
6. Evaluate the system's performance and scalability for deployment in a campus environment.

1.4 BRIEF DESCRIPTION

This project introduces Nest, an AI-driven web platform that supports student entrepreneurs within a campus ecosystem. The platform allows users to create profiles, list products or services, manage business operations, and network with peers. A Content-Based Recommendation System (TF-IDF with Cosine Similarity) suggests relevant products and services, while Linear Regression forecasts sales trends and a Naïve Bayes Classifier performs sentiment analysis on customer

reviews. Additionally, NetworkX graph algorithms recommend potential collaborators based on shared interests and business domains.

Nest is developed using React.js, Node.js, Express, and MongoDB Atlas, with JWT authentication and bcrypt for secure access, and Razorpay for payment processing. The result is a scalable, secure, and intelligent platform that enhances the growth and management of student-led ventures.

2. LITERATURE SURVEY

2.1 REVIEW OF EXISTING SYSTEMS

Several research studies have focused on improving fairness, trust, and user experience in online marketplaces.

Zikun Ye et al. (2023) proposed a seller-side outcome fairness model to ensure equal visibility for less-exposed sellers by optimizing recommendation algorithms. This approach introduces a fair ranking mechanism that balances exposure among all sellers.

M. Gorton et al. (2024) examined users' processing of marketplace listings for both high and low involvement goods. Their study explored how reviews, product images, and price cues influence user decision-making behavior in online marketplaces.

Wei et al. (2023) conducted a study on college students' second-hand trading platforms, emphasizing sustainable trading habits and the creation of campus-based marketplaces. The research provided insights into student engagement in online thrift and resale systems.

N. Peña-García et al. (2024) analyzed reviews, trust, and customer experience in online marketplaces, highlighting how fake reviews and review quality affect user trust and confidence in e-commerce transactions.

These studies contribute to understanding different aspects of online marketplace design, including fairness, trust, sustainability, and user perception.

2.2 LIMITATIONS OF EXISTING SYSTEMS

Despite valuable insights from prior research, existing systems exhibit several limitations when applied to small-scale or student-oriented platforms:

1. Most studies require large datasets and complex computational algorithms, making them unsuitable for smaller applications.
2. Research on user perception (e.g., review and trust mechanisms) focuses primarily on buyers, overlooking seller-side fairness and exposure.
3. Campus-based trading models are often limited in scalability and security, and not easily adaptable beyond one institution.
4. Few studies integrate both fairness and trust mechanisms in a unified platform, which are essential for sustainable small-scale marketplaces.

Therefore, there is a clear need for a system that ensures fair visibility for sellers, secure transactions, and balanced user engagement—particularly in student-focused online marketplaces.

3. METHODOLOGY

3.1 SYSTEM ARCHITECTURE

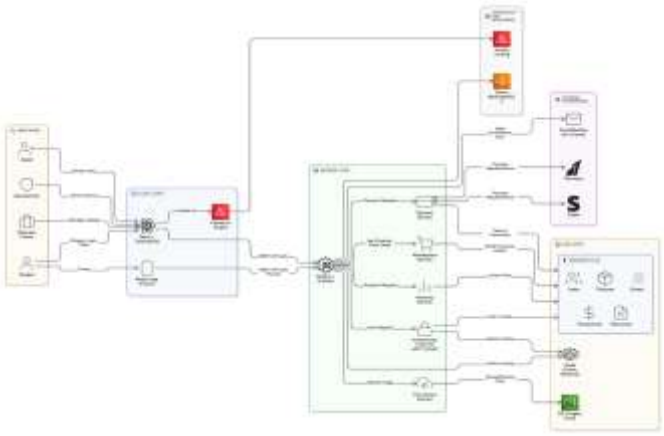

Fig -1: System Architecture Diagram

System Features:

1. User Authentication and Profile Management – Secure sign-up/login using JWT and bcrypt encryption.
2. Product and Service Marketplace – Create, edit, and display listings with images, prices, and descriptions.
3. AI-Powered Recommendation Engine – Suggest related items or services based on user interests and purchase history.
4. Sales Forecasting Module – Predict future performance trends from historical data.
5. Sentiment Analysis Engine – Evaluate customer reviews to determine product reputation.
6. Collaboration Network – Recommend students with similar skills or business categories using NetworkX analysis.
7. Secure Payment Integration – Enable seamless online transactions through Razorpay or Stripe APIs.
8. Admin Dashboard – Manage users, moderate listings, and view analytics reports.

3.2 MODULES DESCRIPTION

3.2.1 USER ROLES:

This layer defines different user types and their access levels within the platform:

1. Guest: Can only browse listings without performing transactions.
2. Student: Can browse products, log in, and place orders.
3. Business Owner: Can manage listings, upload products, and view orders.
4. Administrator: Has privileges for moderation, approval, and analytics management.

3.2.2 CLIENT LAYER:

This layer handles all user-facing interactions, providing the interface through which users view content, perform actions, and communicate with the system.

1. Built using React.js and Tailwind CSS, providing a responsive and user-friendly interface.
2. Hosted on AWS Amplify for continuous deployment and scalability.
3. A mobile app (future extension) will interact with the same backend via REST APIs.
4. The client interacts with backend services using secured API calls for all core operations like login, product management, and payments.

3.2.3 BACKEND LAYER:

This layer manages the core business logic, processing requests from the client, enforcing rules, and coordinating between different services.

1. Developed using Node.js and Express, the backend acts as the core processing engine, consisting of modular services:
2. Authentication Service: Handles user login, registration, and token-based (JWT) session management using bcrypt for encryption.
3. Marketplace Service: Manages product listings, order placement, and updates using CRUD operations.
4. Payment Service: Processes payment requests and responses using integrated gateways.
5. Analytics Service: Generates reports and performance insights for admin and business users.
6. File Upload Service: Handles image/document uploads to the storage system.

3.2.4 DATA LAYER:

This layer stores, organizes, and manages all platform data, ensuring reliability, consistency, and fast retrieval.

1. MongoDB Atlas: Stores collections for Users, Products, Orders, and Transactions.
2. Redis Cache: Used for fast data retrieval, session management, and temporary caching.
3. AWS S3: Stores product images, documents, and other static resources securely.
4. This layered data approach ensures high availability and quick access to frequently used information.
5. Payment Gateways (Razorpay & Stripe): Securely handle all financial transactions.
6. Email/Notification Services (Future Integration): Send order confirmations, alerts, and promotional messages.
7. AWS Elastic Beanstalk/EC2: Used for backend deployment, monitoring, and load balancing.

3.3 ALGORITHMS USED

3.3.1 Content based recommendation system using TF-IDF + cosine similarity

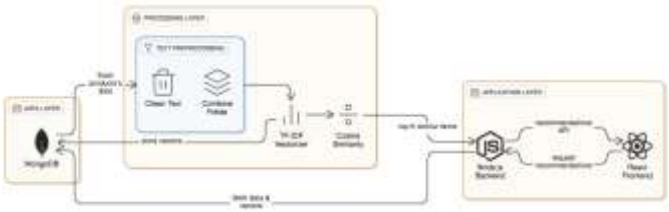

Fig -2: Recommendation System Flow Diagram

This system takes product data from MongoDB and runs it through a processing pipeline where the text is cleaned, unnecessary characters are removed, and important fields are combined into a single meaningful description. That processed text is then converted into numerical vectors using a TF-IDF vectorizer, which basically measures how important each word is across all products. Using these vectors, the system calculates cosine similarity to figure out which items are most alike. The top-K similar products are stored or fetched when needed. When the React frontend asks for recommendations, the Node.js backend fetches the product vectors and similarity results, computes or retrieves the closest matches, and sends them back as a clean recommendations API response. So the whole flow is: data in MongoDB gets transformed into vectors in the processing layer, similarity is computed, and the backend serves those recommendation results to the client.

3.3.2 Linear Regression–based Sales Prediction System

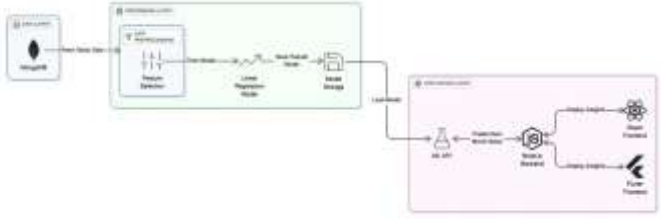

Fig -3: Sales Prediction System Flow Diagram

In this setup, past sales data stored in MongoDB is fetched and preprocessed to clean it, select the right features, and prepare it for model training. A linear regression algorithm is then trained on these historical patterns, and the resulting model is saved for future use. When the backend needs to predict next month's sales, it loads the stored model and passes it the latest relevant inputs. The prediction is then sent to the frontend apps, such as React or Flutter, where the insights are displayed visually to users. Overall, the system takes past sales patterns, trains a forecasting model, stores it, and uses it to generate quick, data-driven predictions on demand.

3.3.3 Naive Bayes Sentiment Analysis

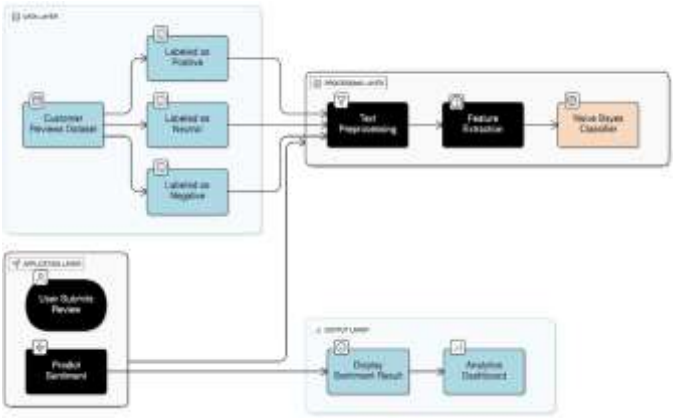

Fig -4: Sentiment Analysis Flow Diagram

This system takes customer review data stored in the data layer and uses labeled examples marked as positive, neutral, or negative to train a Naive Bayes classifier. The text first goes through preprocessing where noise is removed and words are standardized, then important features are extracted to represent each review numerically. The classifier learns patterns from these features and stores the trained model. When a user submits a new review from the application layer, the model predicts its sentiment, and the output layer displays the result to the user while also feeding insights into an analytics dashboard. The whole flow is essentially: labeled data becomes cleaned features, the model gets trained, and every incoming review gets classified instantly.

3.3.4 Graph based Collaboration Network

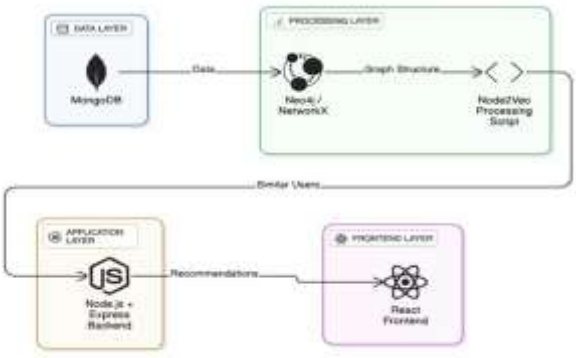

Fig -5: Collaboration Network Flow Diagram

Here, user data from MongoDB flows into the processing layer, where tools like Neo4j or NetworkX convert it into a graph structure that captures relationships such as co-visits, friendships, or similar activity. This graph is then passed through a Node2Vec script, which learns vector embeddings for each user based on their connections and patterns in the network. Using these embeddings, the system identifies clusters of similar users. When the backend receives a request for recommendations, it fetches the nearest users in vector space and returns recommendations to the React frontend. So the entire pipeline turns raw user relationship data into graph embeddings, finds similar users, and serves those insights back to the application layer.

4. UML DIAGRAMS

The Unified Modeling Language (UML) is used to visualize, specify, and document the structure and behavior of software systems. The following UML diagrams were prepared for the Nest platform:

4.1 USE CASE DIAGRAM

The Use Case Diagram outlines the primary interactions between users and the system:

1. Student User registers/logs in, browses products, places orders, makes payments, and views order history.
2. Business Owner uploads and manages product listings and views received orders.
3. Admin manages users and listings and monitors platform activity.
4. Payment Gateway processes transactions and confirms payments.

This diagram provides a high-level view of user roles and system functionalities.

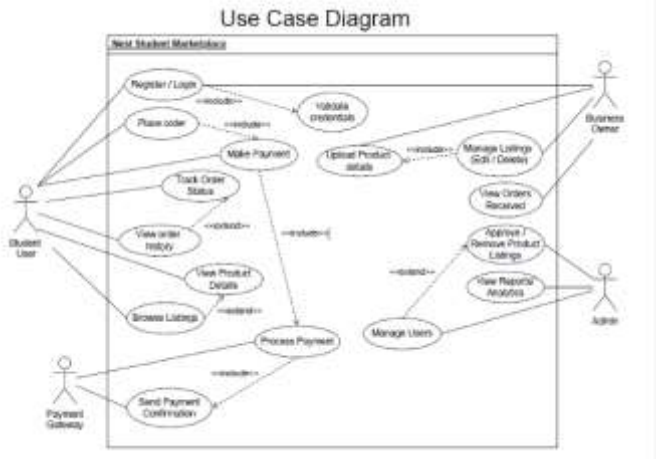

Fig -6: Use Case Diagram

4.2 ACTIVITY DIAGRAM

The Activity Diagram shows the system workflow:

1. Users authenticate and browse listings.
2. Students place orders and complete payments through the payment gateway.
3. The system records transactions and sends confirmations.
4. Business Owners manage product entries.
5. Admins review and moderate listings and generate reports.

This illustrates overall process flow and parallel roles within the platform.

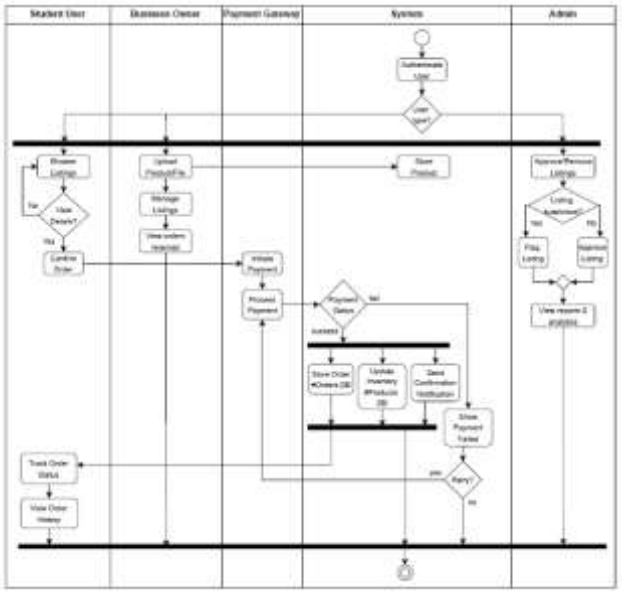

Fig -7: Activity Diagram

4.3 SEQUENCE DIAGRAM

The Sequence Diagram depicts the order placement process:

1. The student initiates an order via the UI.
2. The controller checks product availability and forwards payment details to the payment gateway.
3. Upon payment success, the system stores order information and triggers a confirmation notification.

This shows the interaction flow among key system components.

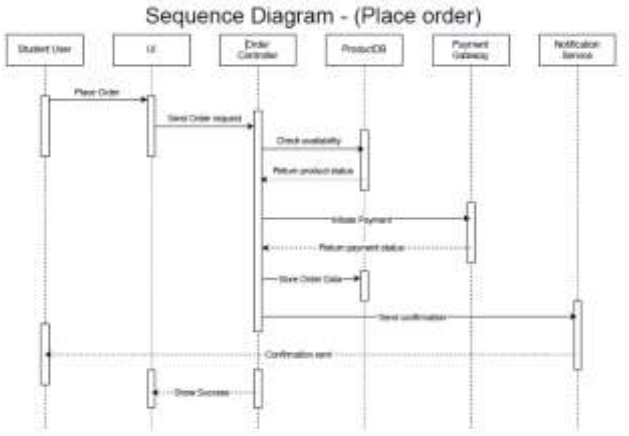

Fig -8: Sequence Diagram

4.4 CLASS DIAGRAM

The Class Diagram highlights core entities and their relationships:

1. User is the base class for StudentUser, BusinessOwner, and Admin.
2. Product stores listing details and is associated with BusinessOwner.
3. Order links StudentUser and Product and tracks status.
4. Payment handles payment processing and is linked one-to-one with Order.

These relationships define the structural foundation of the Nest platform.

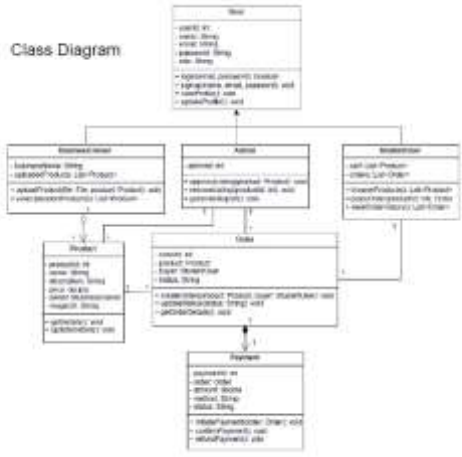

Fig -9: Class Diagram

5. CONCLUSION

Nest successfully delivers an AI-driven platform for student entrepreneurs, integrating personalized recommendations, sales forecasting, and sentiment analysis. Its graph-based collaboration network connects like-minded peers, while secure marketplace features and intuitive dashboards enable efficient business

management. Overall, Nest fosters innovation, data-driven decisions, and growth within campus startups.

ACKNOWLEDGEMENT

We would like to express our sincere thanks to our guide and Head of Department, **Dr. S. R. Ganorkar**, Department of Information Technology, for his consistent guidance and support throughout our Final Year Project. His clear directions, timely feedback, and constructive suggestions helped us complete the project smoothly. We also extend our gratitude to **Dr. S. D. Lokhande**, Principal, Sinhgad College of Engineering, for providing a supportive environment and for his cooperation during the course of our work.

REFERENCES

1. Bhatia-Kalluri, P., et al., "E-Commerce for Rural Micro-Entrepreneurs," arXiv preprint arXiv:2105.12345, 2021. Studies barriers/opportunities for rural sellers adopting e-commerce, highlighting affordability, trust, logistics, and UI/UX considerations.
2. Jain, A., Sharma, A., & Khandelwal, A. (2020). A Marketplace Specifically for Students — College Stalls. International Journal of Computer Applications (IJCA), 176(32), 1–4.
3. Schafer, J. B., Konstan, J. A., & Riedl, J., "Recommender Systems in E-Commerce," Proceedings of the 1st ACM Conference on Electronic Commerce, pp. 158–166, 1999, ISBN: 1581131763. Foundational survey of recommender systems, with applications to personalization in e-commerce.
4. Zawawi, A. A., Rahim, N. Z. A., Hashim, N. A., & Mahmud, M. (2020, December). Student online marketplace for university community. Faculty of Computer & Mathematical Sciences, Universiti Teknologi MARA (UiTM). Accepted: December 29, 2020.