International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Fake News Detection System

Machine Learning and Web-Based Integrated Platform for Automated Misinformation Identification

Zubair Fayaz Shaikh

Department of Computer Science, M.Sc. Computer Science

Roll No. 15 | Academic Year 2026–2027

ABSTRACT-Digitalization has transformed the way information is created, shared, and consumed across social media platforms and online news portals. The rapid proliferation of misinformation and fake news has become a critical societal challenge, influencing public opinion, democratic processes, and public health. Small news organizations and individual users frequently lack access to reliable verification tools and depend on manual fact-checking, which is time-consuming and difficult to scale. This paper proposes "Fake News Detection System", a web-based integrated platform designed to automatically classify news articles and social media posts as genuine or fake using machine learning and natural language processing techniques.

The proposed architecture separates the presentation, application, and data layers. A React.js web client communicates with a Python Flask REST API, which integrates machine learning models trained on textual features. MySQL provides centralized relational storage for user accounts, news records, prediction history, and feedback data. JWT-based authentication and role-based authorization control access for administrators and registered users. The system provides an administrative dashboard for model monitoring, prediction analytics, and user management.

The research follows an applied software engineering methodology involving requirement analysis, review of existing approaches, system modelling, database design, prototype implementation, and evaluation planning. The reference paper uses a progression from problem motivation and related work to research gap, proposed framework, methodology, implementation, evaluation, limitations, and conclusion; this paper follows the same research pattern while developing original content for the Fake News Detection project. The reference emphasizes separating proposed evaluation from completed experimental

Keywords-fake news detection, misinformation, machine learning, natural language processing, web application, Flask, React.js, MySQL, REST API, text classification, TF-IDF, Naive Bayes, logistic regression.

1. INTRODUCTION-News and information are now produced and circulated at unprecedented speed. Social media platforms, messaging applications, and online news portals allow any user to publish content without editorial verification. While this democratization of information has benefits, it has also enabled the rapid spread of fake news, misinformation, and disinformation. Fake news can influence elections, damage public health efforts, incite communal tension, and erode trust in legitimate journalism.

Manual fact-checking by journalists and fact-checking organizations is essential but cannot keep pace with the volume of content published daily. Automated approaches using machine learning and natural language processing offer a scalable way to flag potentially misleading content and assist human reviewers.

The proposed Fake News Detection System combines a machine learning backend with a user-friendly web interface. Users can submit news

text or article content and receive an automated prediction indicating whether the content is likely genuine or fake, along with a confidence score. Administrators can monitor prediction history, manage datasets, retrain models, and review user feedback. A common backend, database, authentication layer, and reporting mechanism reduce duplication while keeping administrative and user workflows separate.

The main contributions are:

· Integrated architecture for automated fake news detection with web-based access.

· Role-specific workflows for general users and administrators.

· Centralized relational data model for users, news submissions, predictions, feedback, and model metadata.

· REST API communication between web client and machine learning backend.

· Machine learning pipeline using TF-IDF vectorization with Naive Bayes and Logistic Regression classifiers.

· Evaluation framework covering classification accuracy, precision, recall, F1-score, API performance, usability, and security.

2. RELATED WORK

2.1 Digital Misinformation and Online News

Online news systems commonly provide article publication, sharing, and commenting. The absence of verification mechanisms allows fabricated stories to spread rapidly. Digital platforms increasingly explore automated flagging, source credibility scoring, and user reporting. Small operators and individual users may still rely on manual judgment or multiple disconnected tools for verification. The proposed system treats news content, prediction results, and user feedback as related records in one database.

2.2 Machine Learning for Text Classification

Text classification approaches commonly use feature extraction methods such as bag-of-words, TF-IDF, and word embeddings, combined with classifiers such as Naive Bayes, Logistic Regression, Support Vector Machines, and deep neural networks. A fake news detection application can train models on labeled datasets containing genuine and fake articles. Controlled evaluation is essential: models should be assessed on held-out data using accuracy, precision, recall, and F1-score, while acknowledging dataset bias and domain shift.

2.3 Natural Language Processing Techniques

NLP techniques enable preprocessing steps such as tokenization, stop-word removal, stemming, and lemmatization. A detection application can transform raw text into structured features and present a prediction with confidence. For an academic prototype, classical machine learning models with TF-IDF features are simpler to implement and evaluate than large transformer models, while still demonstrating the core methodology.

2.4 Web and Mobile Architecture

A separate frontend and backend architecture allows multiple clients to reuse the same business services. A web client can serve administrators and general users. REST APIs provide a standardized communication layer and make additional clients easier to add later.

2.5 Research Gap

The research gap addressed here is not the absence of fake news detection research or web platforms individually. It is the lack of a single academic platform designed around the combined requirements of automated detection, user-friendly access, administrative monitoring, and feedback collection. The project investigates how shared identity, prediction, reporting, and feedback services can support both end users and administrators without mixing their operational rules.

Area	Existing Approach	Proposed Gap Addressed
Fake news detection	Manual fact-checking, isolated ML scripts	Integrated detection with web access
User interaction	No direct user submission	User submission with prediction and confidence
Communication	Email/manual reporting	Centralized prediction history and notifications
Data	Separate files/notebooks	Centralized relational database
Clients	Single interface	Web client with role-specific views
Monitoring	Manual reports	Dashboard and structured prediction history

3. PROPOSED SYSTEM

3.1 System Overview

The system is organized as a multi-client service platform. Customers use tourism services through web/mobile interfaces. Parents use school transportation functions, drivers use mobile workflows for assigned trips, and administrators use a dashboard to manage both service areas. Protected requests pass through authentication and authorization.

[FIGURE: Figure1.OverallSystemArchitecture]
Customer / Parent / Driver / Admin → React Web / Flutter Mobile → Spring Boot REST API → MySQL; optional Maps/GPS and Notification services.

3.2 Major Components

Component	Responsibilities
Web Frontend	News submission display, user account, admin dashboard, reports
Mobile App	Parent and driver workflows; optional customer access
Authentication	Registration, login, token validation, role permissions
Detection Module	Text preprocessing, feature extraction, model prediction, confidence scoring
Dataset Module	Training data storage
Tracking	feature extraction, model prediction
Notifications	results, feedback acknowledgments
Database	Persistent storage and relationships
Admin Module	Master data, monitoring and reports

3.3 User Roles

User: registers, submits news text, views prediction history and confidence scores, provides feedback.

Administrator: manages users, datasets, models, predictions, feedback, and reports.

4. METHODOLOGY

4.1 Problem Formulation

Let U represent users, N news submissions, P predictions, F feedback records, M model metadata, and D training datasets. The system receives requests through authenticated clients and processes them using role-specific business rules. The principal objective is to maintain consistent records and expose only authorized information.

Simplified flow: Request → Authentication → Authorization → Validation → Business Logic → Database/ML Service → Response

4.3 Database Methodology

MySQL is proposed as the relational database. Primary and foreign keys represent entity relationships. Master data such as users, students, vehicles, routes and packages is separated from operational records such as bookings, trips, tracking events and notifications.

Table	Important Fields
users	user_id, name, email, phone, password_hash, role
News	news_submissions news_id, user_id, title,
Content	submitted_at, source_url
Prediction	prediction_id, news_id, label,
Feedback	model_version, predicted_at
Data set	dataset_id, name, description,
Models	model_id, model_name, version, accuracy, precision, recall, f1_score, trained_at
notifications	notification_id, user_id, type, message, read_status

4.4 **Backend methodology**

Python Flask is proposed for server-side implementation. Routes expose REST endpoints, services contain business rules, and repositories communicate with MySQL. Machine learning components load trained models and perform predictions. DTOs can separate API payloads from database entities, and centralized exception handling can provide consistent error responses.

Example API groups:

· /api/auth – registration and authentication

· /api/news – news submission and history

· /api/predict – prediction requests

· /api/feedback – user feedback

· /api/datasets – dataset management

· /api/models – model metadata and retraining

· /api/admin – dashboards and reports

4.5 Frontend and Mobile Methodology

React.js is proposed for the responsive web interface. The client consumes REST APIs so that authentication, validation, business rules, machine learning operations, and database operations remain centralized.

4.6 Machine learning Methodology

When a news submission is received, the backend preprocesses the text using tokenization, lowercasing, stop-word removal, and stemming. TF-IDF vectorization converts the text into numerical features. Trained classifiers such as Multinomial Naive Bayes and Logistic Regression produce a label (genuine or fake) and a confidence score. The prediction is stored with the model version for traceability**4.7 Security Methodology**

- Use secure password hashing rather than plain-text passwords.
- Use JWT or secure sessions for protected API requests.
- Apply role-based authorization.
- Validate input at client and server layers.
- Prevent cross-parent access to protected student records.
- Define retention rules for location and student information.
- Maintain audit logs for important administrative actions.

5. SYSTEM IMPLEMENTATION

5.1 Technology Stack

Layer	Technology
Web frontend	React.js, HTML5, CSS3, JavaScript
Backend	Python, Flask

Data access	Data access SQLAlchemy / Flask-SQLAlchemy
Database	MySQL
Authentication	Spring Security + JWT
API	REST / JSON
Notifications	Email or in-app notifications
Version control	Git / GitHub

5.2 Procedural Flow

[FIGURE: Figure 2. Tour Booking Flow]
Login → Browse Packages → Select Package → Date/Travellers → Validate → Confirm/Payment → Booking → Notification → History.

[FIGURE: Figure 3. School Van Flow]
Driver Login → Assigned Route → Start Trip → Location Updates → Pickup Student → Next Stop → School → Drop Student → End Trip.

5.4 Er design

FIGURE: Figure 4. Conceptual E-R Design] USER–NEWS_SUBMISSION–PREDICTION; PREDICTION–FEEDBACK; DATASET–MODEL; USER–NOTIFICATION.

[5.5 Data Flow Design

At context level, users and administrators exchange data with the Fake News Detection System. The system exchanges data with the database and machine learning model. Level 1 processes are authentication, news submission, prediction, feedback, notifications, and reporting.

5.6 Sequence design

User authenticates with the web application.

2. Application requests an access token.

3. User submits news text.

4. Backend preprocesses the text.

5. Features are extracted using TF-IDF.

6. Model predicts label and confidence.

7. Prediction is stored in the database.

8. Result is returned to the user.

9. Authorized admin views prediction history and feedback.

10. Web application displays prediction and confidence.

6. EVALUATION METHODOLOGY

The reference paper distinguishes planned evaluation from completed experimental results. This project follows the same principle: the following metrics are an evaluation plan and should only be reported as measured results after implementation and testing.

6.1 Functional Testing

Test Area	Metric
Authentication	Successful/failed login cases
News submission	Valid/invalid booking cases
Prediction	Create/update/delete validation
Feedback status	Submit/update/delete validation
Admin	Accepted/rejected validate
Authorization	Unauthorized access attempts
Notifications	Correct event triggering/delivery

6.2 Performance Metrics

- API response time
- · Database query response time
- · Prediction processing latency
- · Web page loading time
- · Controlled concurrent request handling

6.3 Usability and Reliability

Usability can be assessed by representative users performing predefined tasks. Reliability can be assessed through repeated booking, login, trip and API tests. Failures and corrective actions should be

documented instead of claiming improvements without measurements.

6.4 Security Evaluation

Testing can include authentication, authorization, invalid-input handling, token/session management, password-storage verification, and checks that one parent cannot access another student's protected records. OWASP guidance can be used as a general security checklist.

7. LIMITATIONS AND FUTURE WORK

7.1 Limitations

GPS depends on permissions, connectivity, battery and server availability.

Map/notification services may have quotas or costs.

Prototype data may not represent production scale.

Advanced route optimization is outside the minimum implementation.

Payment may initially be simulated.

Student and location data require careful privacy and access control.

Production deployment requires operational and legal review.

7.2 Future Work

•Transformer-based models such as BERT for improved accuracy

· Explainable AI techniques for prediction transparency

· Multilingual support for regional languages

· Source credibility scoring and URL analysis

· Real-time social media integration

· User feedback-driven continuous learning

· Analytics for misinformation trends and model performance

· Cloud deployment and monitoring

· Privacy-preserving data retention

8. CONCLUSION- This paper proposed Fake News Detection System, an integrated web-based platform for automated misinformation identification. The system connects general users and administrators through centralized services and a relational database.

The proposed architecture uses React.js for the web interface, Python Flask for REST services, Scikit-learn for machine learning, and MySQL for structured storage. Machine learning and natural language processing techniques extend the platform toward an intelligent verification service. The project demonstrates M.Sc. Computer Science concepts including requirements engineering, system architecture, database modelling, API development, machine learning, natural language processing, authentication, security, and evaluation.

At the proposal stage, no production-level improvement is claimed. The next step is to implement the prototype, prepare test data, execute functional and performance tests, collect usability feedback, and report measured results. This distinction keeps proposed design separate from experimental evidence, consistent with the reference research-paper approach.

ACKNOWLEDGMENT- I thank Ms. Karishma Jain
„Department of Computer Science, Patkar-Varde College, for guidance on the project that this paper condenses, and the college for laboratory access. We also thank the Research Methodology faculty for the chapter structure that separated problem, method, and deferred work.

REFERENCES

[1]1] S. B. Ahmed et al., "Fake News Detection Using Machine Learning Ensemble Methods," IEEE Access, 2020.

[2] Flask, "Flask Documentation."

[3] Scikit-learn, "Scikit-learn Documentation."

[4] React, "React Documentation."

[5] MySQL, "MySQL Documentation."

[6] NLTK, "Natural Language Toolkit Documentation."

[7] OWASP Foundation, "OWASP Top 10: The Ten Most Critical Web Application Security Risks."

[8] Object Management Group, "Unified Modeling Language (UML) Specification."

[9] R. S. Pressman and B. R. Maxim, Software Engineering: A Practitioner's Approach, McGraw-Hill.

[10] I. Sommerville, Software Engineering, Pearson.

[11] R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, Pearson.

[12] T. Mikolov et al., "Efficient Estimation of Word Representations in Vector Space," arXiv, 2013.

[13] A. Vaswani et al., "Attention Is All You Need," NeurIPS, 2017