PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064

FAKE SOCIAL MEDIA ACCOUNTS AND THEIR DETECTION

A PROJECT REPORT

Submitted by,

DEEPIKA S	20221CCS0104
KRITIKA PRIYA	20221CCS0065
RISHIKA H J	20221CCS0059

Under the guidance of,

Dr. Sharmasth Vali Y

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY)

PRESIDENCY UNIVERSITY

BENGALURU

DECEMBER - 2025

PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this report "**FAKE SOCIAL MEDIA ACCOUNTS AND THEIR DETECTION**" is a bonafide work of "**Deepika S (20221CCS0104), Kritika Priya (20221CCS0065), Rishika H J (20221CCS0059)**", who have successfully carried out the project work and submitted the report for partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE ENGINEERING, CYBER SECURITY** during 2025–26.

Dr. Sampath A K	**Dr. Sharmasth Vali Y**	**Dr. Sampath A K**	**Dr. Anandaraj S P**
Project Guide	Program Project	**Dr. Geetha A**	Head of the Department
PSCS	Coordinator	School Project	PSCS
Presidency University	PSCS	Coordinators	Presidency University
	Presidency University	PSCS	
		Presidency University	

Dr. Shakkeera L	**Dr. Duraipandian N**
Associate Dean	Dean
PSCS	PSCS & PSIS
Presidency University	Presidency University

Name and Signature of the Examiners

1)

2)

PRESIDENCY UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We the students of final year B.Tech in **COMPUTER SCIENCE ENGINEERING, CYBER SECURITY** at Presidency University, Bengaluru, named Deepika S, Kritika Priya, Rishika H J hereby declare that the project work titled **"FAKE SOCIAL MEDIA ACCOUNTS AND THEIR DETECTION"** has been independently carried out by us and submitted in partial fulfillment for the award of the degree of B.Tech in COMPUTER SCIENCE ENGINEERING (CYBER SECURITY) during the academic year of 2025-26. Further, the matter embodied in the project has not been submitted previously by anybody for the award of any Degree or Diploma to any other institution.

Name	**Roll Number**	**Signature**
Deepika S	**20221CCS0104**	
Kritika Priya	**20221CCS0065**	
Rishika H J	**20221CCS0059**	

PLACE: BENGALURU

DATE: 1st December 2025

ACKNOWLEDGEMENT

For completing this project work, We/I have received the support and the guidance from many people whom I would like to mention with deep sense of gratitude and indebtedness. We extend our gratitude to our beloved **Chancellor, Pro-Vice Chancellor, and Registrar** for their support and encouragement in completion of the project.

I would like to sincerely thank my internal guide **Dr. Sharmasth Vali Y , Professor**, Presidency School of Computer Science and Engineering, Presidency University, for his/her moral support, motivation, timely guidance and encouragement provided to us during the period of our project work.

I am also thankful to **Dr. Anandaraj, Professor, Head of the Department, Presidency School of Computer Science and Engineering** Presidency University, for his mentorship and encouragement.

We express our cordial thanks to **Dr. Duraipandian N**, Dean PSCS & PSIS, **Dr. Shakkeera L**, Associate Dean, Presidency School of computer Science and Engineering and the Management of Presidency University for providing the required facilities and intellectually stimulating environment that aided in the completion of my project work.

We are grateful to **Dr. Sampath A K, and Dr. Geetha A, PSCS** Project Coordinators**, Dr. Sharmast vali, Program Project Coordinator**, Presidency School of Computer Science and Engineering, or facilitating problem statements, coordinating reviews, monitoring progress, and providing their valuable support and guidance.

We are also grateful to Teaching and Non-Teaching staff of Presidency School of Computer Science and Engineering and also staff from other departments who have extended their valuable help and cooperation.

Deepika S

Kritika Priya

Rishika H J

ABSTRACT

Fake social media profiles have become a growing threat on digital platforms, contributing to the spread of misinformation, cyber fraud, identity theft, and malicious activities. As these fake accounts continue to evolve, traditional manual and rule-based detection methods are no longer sufficient to identify sophisticated bot behaviors and fabricated user identities. This project addresses these challenges by developing an automated system capable of detecting fake profiles using machine learning techniques integrated into a web-based environment.

The system is built using Python and Django, supported by machine learning algorithms that analyze profile information, user behavior, network patterns, and textual content to identify suspicious accounts. Techniques such as classification models, feature engineering, natural language processing, and real-time data retrieval from social media APIs are used to improve detection accuracy. A structured web interface enables users and administrators to analyze profiles, view detection results, report suspicious accounts, and monitor patterns through visual dashboards.

The resulting system provides a reliable, scalable, and user-friendly solution for detecting fake profiles with enhanced accuracy. By integrating machine learning with web technologies, the project demonstrates how automated detection can significantly contribute to creating safer social media environments, supporting both end-users and platform administrators in identifying harmful or deceptive online identities.

Table of Content

List of Figures

(include list of all figure and page no

e.g.

List of Tables

(include list of all tables and page no

e.g.

Abbreviations

(include all abbreviations used in report in alphabetic order)

e.g.

IoT	Internet of Things
SDG	Sustainable Development Goal

Chapter 01

Introduction

1.1 Background

In today's digital environment social media is a significant component of everyday life, enabling many forms of connection, sharing and community on a global level, but can also allow for malicious attacks from compromised and/or fake accounts. Fake profiles on social media, or inauthentic accounts, are created with malicious intent to deceive the audience based on the messages shared, and these accounts may contain spam, disinformation, manipulation of public opinion, fraudulent scams and/or identity theft, depending on the user. Evidence suggests that machine-learning methods are being developed for identifying fake profiles, as many of the constraint-based frameworks that have been developed do not efficiently evaluate fake profiles nor accurately assess fake profiles. A machine learning-based study on social media fake profile detection showed boosting algorithms, such as XGBoost, were suitable to identify fake versus authentic social media accounts. SCIRP

The urgency of detecting and remediating fake profiles cannot be overstated as the threats are becoming more significant and the capability for malicious actors and less sophisticated entities is becoming more common. Crime does not solely consist of basic bots and fake profiles but also by complex actors, including those who engaged AI to create fake faces and conduct coordinated patterns of accounts and behaviors to look similar to real users.

1.2 Statistics

Fake or duplicate accounts are a major issue on social media platforms. For instance, Facebook Inc. individualized the percentage of such users to around 7-9% after 2012, but this number gradually reached about 13-14% by 2017. Moreover, a study that classified accounts on X (formerly Twitter) came up with a database of 1,420 accounts with GAN-generated images and prospected that the lower limit for such profile types is from 0.021% to 0.044% of active users, which equals approximately 10,000 daily active accounts in that database. Another investigation disclosed that roughly 80 % of the participants in the survey had come across suspicious or fake accounts on social media, and 77 % had been the recipients of unsolicited link requests from strangers. Taking these statistics into account and considering the fact that social media is immensely popular worldwide including in India there arises a necessity for the development of systems which can effectively detect and reduce the risks associated with fake profiles, particularly in the case of those regional contexts where the local language, usage patterns and threat vectors might be dissimilar.

1.3 Prior Existing Technologies

Detection of fake profiles has mostly been done through rule-based heuristics, manual moderation, black-list filtering, and simple statistical thresholds (e.g., uncommonly high friend-to-follower ratio); these methods, while helpful, usually have a hard time with the evoking threats that have a very similar behavior to legitimate users.

The last few years have seen the use of machine-learning techniques in this area. For instance, different classification algorithms like Decision Trees, Random Forests, Support Vector Machines, and boosting methods (e.g., XGBoost) have been tried in the unmasking of fake social media profiles. A research project called "Fake Profile Detection Using Machine Learning Techniques" used several models—including Random Forest and XGBoost—on Twitter profile datasets and discovered that the boosting algorithm yielded better results. A study titled SCIRP Another work introduced a "majority voting" ensemble of multiple algorithms (Decision Trees, Random Forests, K-Nearest Neighbours, AdaBoost, Logistic Regression) to classify fake vs. real accounts. publications.eai.eu While these technologies mark improvements, they still face limitations in scalability, explainability, real-time applicability, and adaptation to new attack patterns (for example, AI-generated profile faces or coordinated networks).

Thus, there is room for improving existing methods by integrating advanced feature engineering (profile, activity, network features), applying modern machine-learning models (including deep learning), and building a web-based system for real-time detection and administrator oversight.

1.4 Proposed Approach

Aim of the Project

The objective of this project is to create a scalable system that will automatically spot fake social media accounts by examining user behavior, profile characteristics, and network interactions etc. Additionally, a web-based interface will be made available for users and administrators to interact with detection results.

Motivation

The motivation is derived from the ever-growing threat of fake or malicious profiles on social networks, which in turn create a situation where trust in the platform is diminished and the situation can conveniently be used for misinformation, fraud, cyber-bullying, and identity theft. The rapidly growing and evolving fakeness in accounts combined with the inadequacy of purely manual or rule-based detection methods, push toward the integration of machine learning, web systems for detection, and reporting.

Proposed Approach

The project takes a multi-layered strategy:

- To start with, features based on profiles will be pulled out (e.g. account age, number of friends/followers, presence of profile picture, bio completeness).
- Then, activity-based features will be derived (e.g., posting frequency, engagement rate, comment sentiment).
- Subsequently, an investigation of feature-based networks will be performed (e.g., mutual connections, clustering coefficient, ratio of genuine to suspicious friends).
- The machine-learning classification model (e.g., XGBoost, Random Forest) will be trained on labeled datasets including real and fake profiles. Feature selection and performance tuning (cross-validation, hyperparameter optimization) will be carried out.
- At last, the detection model will be consolidated into a web application that is created using the Django web framework, allowing users to have interfaces for inputting or uploading profiles, seeing detection results, marking accounts as suspicious, and facilitating administrators with dashboards and reporting tools.

Applications of the Project

The detection module can be integrated by social media platforms to review or flag suspicious accounts ahead of time.

Brand or corporate social media mentions could be monitored by security teams in organizations and fake profiles that may impersonate or target their brand could be filtered out.

Based on the behavioural scoring, individuals can be alerted about suspicious connections or friend requests.

The system can be used by researchers as a tool to study fake account patterns, evolving adversary tactics, and feature-based detection's effectiveness.

Limitations of the Proposed Approach

- The model's effectiveness is based primarily on the provided high-quality labelled datasets; otherwise, its detection accuracy may be less if it does not have representative data covering the full range of fake account behaviours.
- Detecting patterns of fake behaviour is one of the things machine-learning models can do but an adversary might continue to evolve and com up with new behaviours (e.g., more sophisticated bots, generative-AI profile pictures) that demand retraining or redesign.
- Real-time API integration (for example, through social media APIs) may encounter different challenges depending on the platform such as rate limits, privacy restrictions or policy barriers.
- The web-based system could be a cause of user privacy, data security, and compliance with platform terms of service issues that would need to be thoroughly handled.
- Human oversight might still be required because there may be false positives (real users flagged as fake) or false negatives (fake accounts missed).

1.5 Objectives

1. To perform an analysis of the social media account data and to derive such features as behavioural, profile-based, and network-based that can determine authenticity or dishonesty.

2. To design and test the classification models based on machine learning for the purpose of fake profile detection, with a focus on comparing the mentioned algorithms: Logistic Regression, Decision Trees, Random Forest, Support Vector Machines, and Gradient Boosting (e.g. XGBoost), by means of the evaluated performance metrics like accuracy, precision, recall, and F1-score.

3. To design a web-based platform (with Django) for communication between users and administrators: letting the input or upload of profile data, showing the results of detection, marking suspicious accounts, and giving monitoring trend dashboards.

1.6 Sustainable Development Goals (SDGs)

This project is connected with various United Nations' Sustainable Development Goals (SDGs):

• SDG 16: Peace, Justice and Strong Institutions – The system, by revealing and reducing fake social media profiles, plays a part in creating online information environments that are less risky and more transparent and makes it easier for institutions to keep digital communication's trustworthiness.

• SDG 9: Industry, Innovation and Infrastructure – Advanced tech solution (machine learning, web systems) development is the innovation in the cyber security infrastructure and the digital platforms' strengthening.

• SDG 4: Quality Education – The project is raising the awareness of digital identity threats and at the same time giving education on the safe online practices for users, particularly in the educational or community contexts. Thus, the project not only takes on a technical challenge but also helps to create the social goal of trustworthy digital intercourse and responsible innovation.

1 NO POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
5 GENDER EQUALITY
6 CLEAN WATER AND SANITATION
7 AFFORDABLE AND CLEAN ENERGY
8 DECENT WORK AND ECONOMIC GROWTH
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE
10 REDUCED INEQUALITIES
11 SUSTAINABLE CITIES AND COMMUNITIES
12 RESPONSIBLE CONSUMPTION AND PRODUCTION
13 CLIMATE ACTION
14 LIFE BELOW WATER
15 LIFE ON LAND
16 PEACE, JUSTICE AND STRONG INSTITUTIONS
17 PARTNERSHIPS FOR THE GOALS
SUSTAINABLE DEVELOPMENT GOALS

Fig 1.1 Sustainable development goals [1]

1.7 Overview of Project Report

The first chapter introduces the theme of detection of fake social media profiles, elucidating its history, importance, present technologies, suggested method, goals, and United Nations' Sustainable Development Goals (SDGs) alignment. The second chapter offers an overview of the literature related to the current research in deceiving account detection, scrutiny of the users' digital footprints, and the role of machine learning in this process. The third chapter accounts for the methodology applied in the research project along with providing in-depth descriptions of the IT languages, frameworks, libraries, tools, data gathering, feature extraction, and model building. The fourth chapter layout the web-based system's design and implementation covering the system's architecture, the user interface, the API integration, and the deployment strategy. The fifth chapter brings in the results and assessments of the system: model performance metrics, visualizations, case studies, and the discussion of the results. The sixth chapter brings forward the conclusions reached, limitations of the research, future improvements and suggestions.

CHAPTER 02

Literature review

2.1 Introduction

The very fast development of social networking platforms such as Facebook, Instagram, Twitter (X), LinkedIn, and TikTok has made an enormous change to global communication. But, at the same time, these platforms have the same serious problem – fake social media accounts, which are very common and are a major factor in spreading lies, tormenting through the internet, stealing people's identities, swindling, and so on. The traditional rule-based detection systems and manual moderation are already becoming obsolete as they cannot cope with the realistic human-like behavior of fake accounts anymore. Thus, ML and AI-based techniques have been already applied in detecting fake profiles with much more accuracy, scalability, and versatility. In this section, the most important research works in the field of fake profile detection are reviewed. The various methods used in their paper are classified into feature engineering, building supervised ML models, employing the ensemble method, graph-based analysis, and the use of deep learning.

2.2 Fake Profiles and Their Impact on Social Networks

Due to their nature, fake profiles can normally be distributed into the following categories:

• Bot Accounts – Non-human accounts that will make a post, follow someone, and change the trend. • Spam Profiles – Accounts that post ads, phishing links, and do scams.

• Impersonation Accounts – People pretending to be somebody else online.

• Malicious Profiles – Accounts that are used for cyber-attacks, spreading lies, political manipulation, and social engineering.

Fake profiles are major threats to user privacy, lower the trust level in the platform, and can even sway the public through their influence.

2.3 Challenges in Fake Profile Detection

1. Evolving Nature of Fake Accounts – Novel tactics are regularly being employed by the intruders to escape detection.

2. Data Imbalance – There are far more real profiles than fake ones, which causes the ML training to be biased.

3. Behavioural Complexity – Sometimes, the impersonators are performing such human-like actions that it becomes difficult to distinguish them.

4. Real-Time Detection Difficulty – The majority of the algorithms do not support the functionality of real-time verification. Because of these challenges, the adoption of adaptive ML models has become increasingly popular among researchers, as opposed to static rule-based systems.

2.4 Traditional Approaches to Fake Profile Detection

2.4.1 Rule-Based and Heuristic Methods Predefined rules were, for the most part, the backbone of the earliest detection systems, including, but not limited to:

• Profile completeness checks (bio, picture, personal details)

• Posting pattern identification (too frequent or highly repetitive activity)

• Location-based anomalies (sudden login from distant regions) Limitations:

• High false positives

• Not adaptable to evolving fake profiles

• Manual moderation needed, which is slow and inefficient

Thus, researchers shifted towards automated ML-based solutions.

2.5 Machine Learning–Based Approaches

Present-day ML methodologies are strongly built on scrutinizing the patterns related to behavior, contents, and networks.

2.5.1 Feature Engineering

The ML approaches through profiles and user behaviors to find out the following relevant features: Profile-Based Features

• Duration of the account

• Completeness of the profile

• The number of people following and the number of people followed by Activity-Based Features

• Frequency of posting

• Rate of engagement

• Posts' sentiments through analysis Network-Based Features

• Proportion of mutual friends

• Community detection in networks (graph clustering)

• Identification of groups of fake accounts that are connected through others

Through the use of these features, it becomes possible to tell apart real accounts from unreal ones more easily.

2.6 Review of Research Articles

[1] Li et al. (2016) Li et al. put forward a clustering-based method for the detection of fake enjoyments that are together on social networks. Their plan was to cluster the accounts that had corresponding patterns of the interactions. Though the method was good at finding synchronized fake activities, it didn't perform well at times by identifying the non-fake users that were taking part in such activities due to their being on the popular trend.

[2] Van Der Walt & Nel-off (2018) In this paper, the authors always differ-ed bots from human profiles by means of occupational ML classifiers. The classifiers were fed behavioral features in the form of posting times and linguistic patterns. Ensemble models performed with high precision but the challenge of era-true deployment still existed because of the advanced feature extraction.

[3] Khaled et al. (2018) Khaled et al. made the use of SVM and Decision Trees models to do the detection of fake Facebook-like profiles in terms of both behaviors and texts. Their method gave more accurate results but was stopped by the small number of datasets as well as absence of the cross-platform validation.

[4] Sarfraz et al. (2022) The researchers applied XGBoost to identify fake social media profiles by gaining insights from both statistics and behavior. The model was accurate but faced difficulties when it came to AI-produced deepfake profile pictures.

[5] Karamu and Araka (2022) authors proposed a hybrid machine learning model that combined different classifiers. Their ensemble led to improvement in accuracy but on the other hand it asked for a lot of computational resources, which made it hard to do the detection in real-time.

[6] Ahmad & Tripathi (2023) This review pointed to behavioral signs such as sudden increases in followers and repetitive activity patterns. The authors talked about the necessity of retraining models every now and then since the attackers are always changing.

[7] Patil et al. (2023) Patil et al. came up with a majority voting ensemble model. This model improved consistency because it combined different classifiers, but on the other hand, performance was very much dependent on the accuracy of the individual models.

[8] Amankeldin et al. (2023) The research applied a NN to detect the fake profiles by behavioral sequence analysis. The model was highly accurate but as well it demanded large datasets and substantial computational power.

[09] Agravat et al. (2024) The authors created an entire ML-based fake profile detection and reporting system. Their model was ready for deployment but was limited by the API constraints for real-time data collection.

2.7 Summary of Literature

Table 2.1 Summary of Literature reviews

Article	**Year**	**Journal / Conference**	**Method Used**	**Key Features**	**Merits**	**Demerits**
Li et al.	2016	IEEE ICDM	Clustering	Group behaviour patterns	Detects coordinated fake engagement	May misclassify viral trends
Van Der Walt & Nel-off	2018	IEEE ICACCE	Supervised ML	Behavioural + linguistic features	High accuracy	Not real-time
Khaled et al.	2018	IEEE ICCES	ML classifiers	Text + behaviour	Improved detection accuracy	Small dataset
Sarfraz et al.	2022	IEEE Access	XGBoost	Behavioural + statistical	High performance	Weak against deepfake profiles
Habib et al.	2022	IEEE Access	Survey of ML/DL	Comprehensive feature review	Identifies gaps	No practical model

Article	**Year**	**Journal / Conference**	**Method Used**	**Key Features**	**Merits**	**Demerits**
Karamu & Araka	2022	IEEE Access	Hybrid ML ensemble	Multimodel features	Very accurate	High computation
Ahmad & Tripathi	2023	IEEE GUCON	Review	Behavioural attributes	Highlights evolving threats	No implementation
Patil et al.	2023	IEEE Access	Majority voting ensemble	Multiple classifiers	Stable predictions	Depends on base models
Amankeldin et al.	2023	IEEE Access	Deep Neural Network	Behavioural sequences	High accuracy	Requires large data
Agravat et al.	2024	IEEE ICCES	ML system implementation	Full detection pipeline	Deployable	API d

CHAPTER 03

Methodology

The Fake Social Media Profile Detection and Reporting System was built utilizing the V-Model Software Development Methodology, as it is a systematic and verification-oriented approach ensuring the reliability of machine-learning and web-based systems. The V-Model is appropriate for this project because every development step has an associated testing stage, thus maintaining accuracy in data preprocessing, model building, system integration, and deployment.

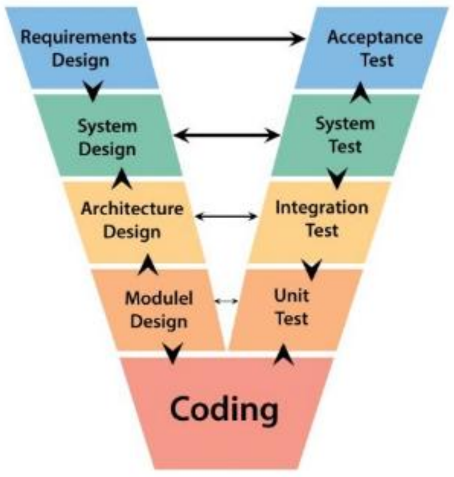


Fig 3.1 The V model methodology [4]

The project encompasses stages such as requirements analysis, system design, module implementation, unit testing, integration testing, verification, and validation. V-Model layers correspond to each of these stages thereby, guaranteeing quality, correctness, and the continuous evaluation of the project throughout the life cycle.

3.1 Requirement Analysis Phase

The basic requirements of the system were studied in the requirement phase of the V-model (Fig. 3.1). This involved:

• Understanding the characteristics of fake profiles through literature review

• Recognizing indicators that are behavioral, profile-based, and network-based

• Deciding on the datasets necessary for ML model training

• Specifying functional requirements such as profile input, prediction output, admin dashboard, and reporting features

• Identifying non-functional requirements such as accuracy, usability, performance, and scalability This phase lies directly to the verification phase, in which the requirements are checked for completeness and correctness.

3.2 System Design Phase

During the system design phase, the architecture of the Fake Profile Detection System was laid out. The other parts that make up the system include:

• Frontend (HTML, CSS, JavaScript) intended for the users interaction

• Backend (Django) which takes care of requests, data processing, and API communication

• Machine Learning Module that is responsible for classification using Python

• Database (SQLite/MySQL) that is used for the storage of user data and predictions

• Admin Dashboard which allows the viewing of flagged profiles and reports

In Fig. 3.2, the design phase is displayed that corresponds to high-level verification, where system architecture is checked against requirements. A block diagram that shows the system architecture will be drawn using Draw.io (Fig. 3.3).

3.3 Project Stages Mapped to V-Model Phases

The Fake Social Media Profile Detection and Reporting System was created by linking each activity to a corresponding V-Model phase. This made it possible to have systematic development, continuous verification, and structured validation at every project stage and for the whole project lifecycle.

Table 3.1 V model phases

V-Model Phase	Project Implementation Description
Requirement Analysis	Identification of system requirements based on literature, user needs, and threat analysis. This includes understanding types of fake profiles, required ML features (profile, activity, network), input data formats, user roles (normal user/admin), and expected system outputs. Requirements for accuracy, performance, privacy, and real-time detection were also defined.

V-Model Phase	Project Implementation Description
System Design	Designing the overall architecture of the Fake Profile Detection System, including ML pipeline structure, Django backend architecture, frontend interface, database schema, API integration points, authentication module, and admin dashboard layout. The design ensures modularity and smooth interaction between components.
Functional Design	Defining the functions of each system component such as: data preprocessing, feature extraction, ML model inference, user input handling, profile classification logic, reporting mechanism, visualization dashboards, and admin management tools. Each functional block is mapped to system requirements.
Unit Design	Designing individual modules such as ML training scripts, Django views, serializers, model files, HTML/JS UI components, database models, and API endpoints. This stage also includes defining validation rules for user inputs and ML prediction logic.
Coding / Implementation	Developing all system modules using Python, Django, HTML, CSS, JavaScript, and SQLite/MySQL. This includes implementing the ML model (e.g., XGBoost/Random Forest), creating the web interface, building authentication flows, integrating the ML engine into Django, and implementing admin features.
Unit Testing	Testing each module individually, including ML prediction functions, backend API routes, database operations, form validations, and frontend interactions. Ensures that each unit operates correctly before integration.
Integration Testing	Verifying end-to-end workflow from user profile input → backend processing → ML model prediction → results display → admin review. Ensures all modules work together without failures.
System Testing	Complete system-level testing with real-world profile samples. Includes performance testing of the ML model, load testing on backend requests, security testing for authentication, and validation of dashboard features.

V-Model Phase	Project Implementation Description
User Validation	Validating system outputs with expected behavior — checking if predictions (real/fake) are accurate, visualizations are meaningful, reporting works correctly, and the interface meets user expectations. Ensures the final system is ready for real usage and matches stakeholder requirements.

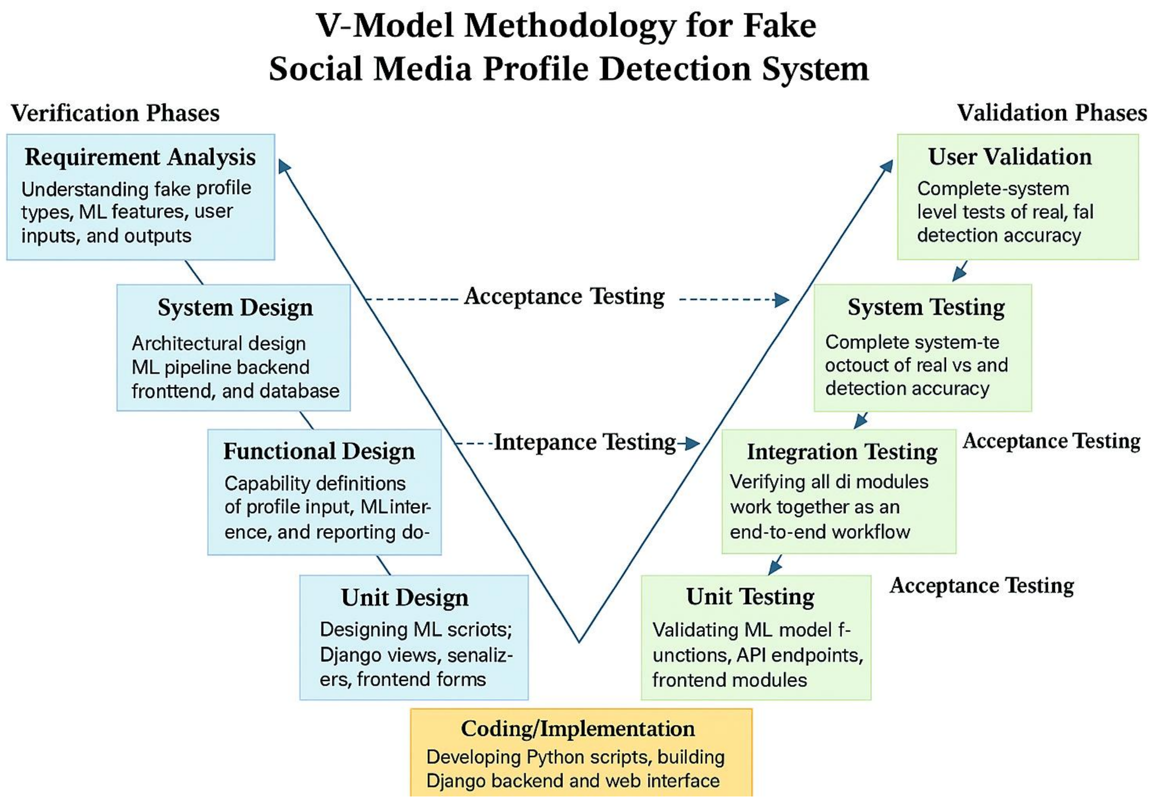

V-Model Methodology for Fake Social Media Profile Detection System
Verification Phases
Validation Phases
Requirement Analysis
Understanding fake profile types, ML features, user inputs, and outputs
System Design
Architectural design ML pipeline backend fronttend, and database
Functional Design
Capability definitions of profile input, MLinter-ence, and reporting do-
Unit Design
Designing ML scriots; Django views, senaliz-ers, frontend forms
Coding/Implementation
Developing Python scripts, building Django backend and web interface
User Validation
Complete-system level tests of real, fal detection accuracy
System Testing
Complete system-te octouct of real vs and detection accuracy
Integration Testing
Verifying all di modules work together as an end-to-end workflow
Unit Testing
Validating ML model f-unctions, API endpoints, frontend modules
Acceptance Testing
Intepance Testing
Acceptance Testing
Acceptance Testing

Fig. 3.3 System Mapping to V-Model

3.4 Justification for Using the V-Model

The V-Model was selected for the Fake Social Media Profile Detection and Reporting System because it matched the project's needs for structured development, accuracy and continuous validation. The main reasons are:

1.Strong Emphasis on Validation:

The ML model's predictions (real vs fake) needed constant validation against dataset benchmarks and test profiles. The V-model guarantees that every development phase has a corresponding test phase which increases detection trustworthiness.

2.High Reliability:

Fake profile detection requires high precision and low false positives. The V-Model aids in spotting defects during the requirements, design, and feature engineering steps before the model goes live.

3.Parallel Development and Testing:

Machine learning modules, Django backend components, and the frontend interface could be created separately and tested simultaneously, which is in line with the V-Model's verification–validation flow.

4.Structured Documentation:

Every process step—requirements, system design, feature extraction, ML pipeline, and testing—is well-documented. This promotes transparency, academic evaluation, and easier integration into research papers.

3.5 Alternative Methodologies Reviewed

Before arriving at the V-Model, several software development methodologies were scrutinized in terms of their compatibility with this ML+Django web application.

Table 3.2 Alternative methodologies

Methodology	**Strengths**	**Limitations for This Project**
Waterfall Model	Simple, linear, easy to document	Inflexible for refining ML features and retraining models
Spiral Model	Strong risk management, iterative updates	Too complex for structured ML pipeline development
DevOps	Fast iterations and deployment cycles	Not suitable since continuous deployment was not required

Methodology	**Strengths**	**Limitations for This Project**
Agile / Scrum	Adaptive, rapid development cycles	Difficulty managing ML model tuning in short sprints
Onion Model	Security-focused	Not fully aligned with ML pipeline and Django architecture needs

3.6 Comparison of Various Methodologies

Table 3.1 Comparison of Software Development Methodologies

Methodology	**Flexibility**	**Testing Focus**	**Documentation**	**Suitability for Project**
Waterfall	Low	High	High	Moderate
Spiral	Very High	High	Medium	Moderate
V-Model	Moderate	**Very High**	**Very High**	**Excellent**
DevOps	Very High	Continuous	Medium	Low
Agile / Scrum	High	Iterative	Low	Medium

3.7 Project Breakdown Structure

Table 3.2 Summary of Project Breakdown into Tasks

Phase	**Task Code**	**Task Description**
Requirement Analysis	R1	Identify fake profile indicators and dataset requirements
System Design	D1	Define overall architecture (ML model → Django backend → UI → Database)

Phase	**Task Code**	**Task Description**
Functional Design	D2	Design data flow, ML pipeline functions, and system behavior
Unit Development	U1	Develop ML scripts for feature extraction and model training
Unit Development	U2	Develop Django backend APIs and frontend UI screens
Testing	T1	Perform unit tests on ML model, API endpoints, and UI modules
Integration Testing	T2	Validate complete workflow from profile input to prediction
Validation	V1	Test system accuracy using real and fake profiles
Deployment	DP1	Host Django project and deploy final ML model

The V-Model, which is a software development model that shows the different stages of development, was chosen as the main method for the Fake Social Media Profile Detection System because of its clarity, structured development flow, and strong emphasis on testing at every stage.

A verification phase corresponding to each development task, ranging from ML dataset preparation to Django integration and result validation, was implemented to make sure that mistakes were caught at an early stage.

This strategy led to the following results for the system:

• High prediction accuracy

• Fake vs real profiles validation

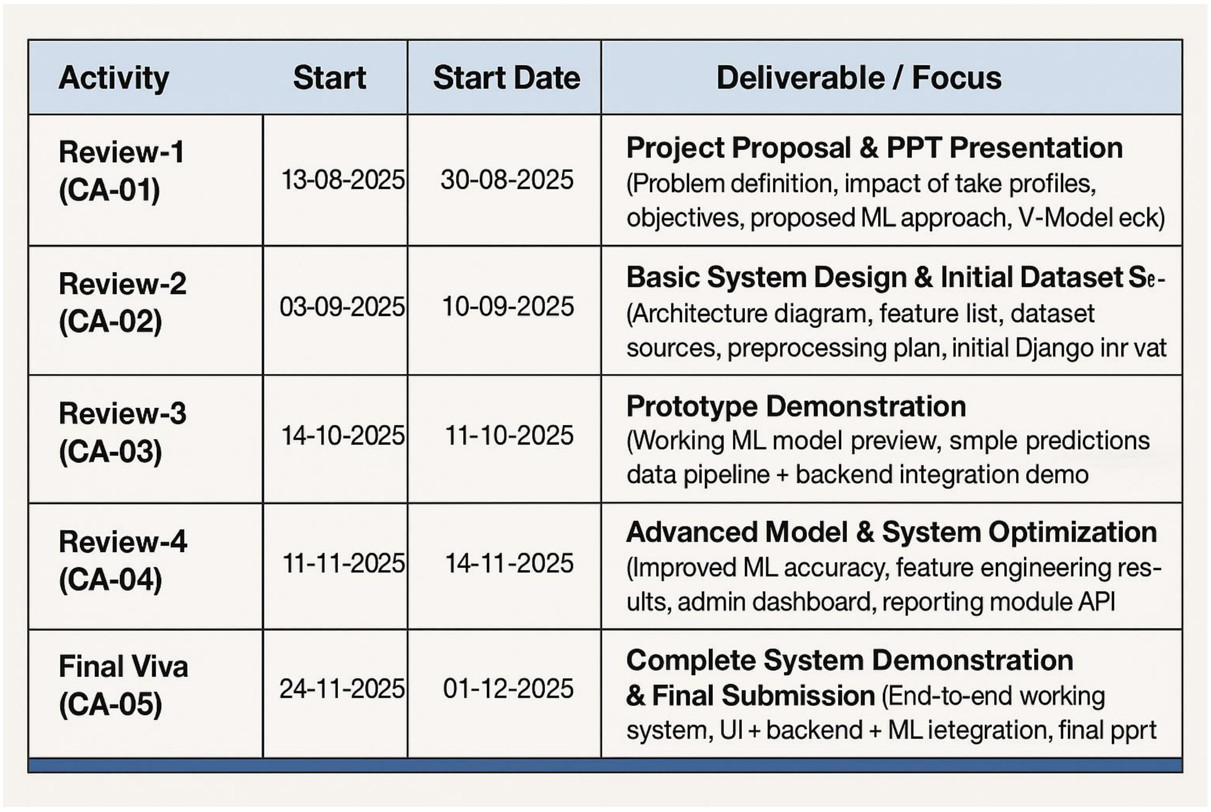
• Clear traceability for academic documentation

• Reliable deployment with minor defects The V-Model's systematic approach delivered consistency, reliability, and transparency—qualities that are indispensable for a machine-learning-based cyber-security system.

CHAPTER 04

Project Management

4.1 Project timeline

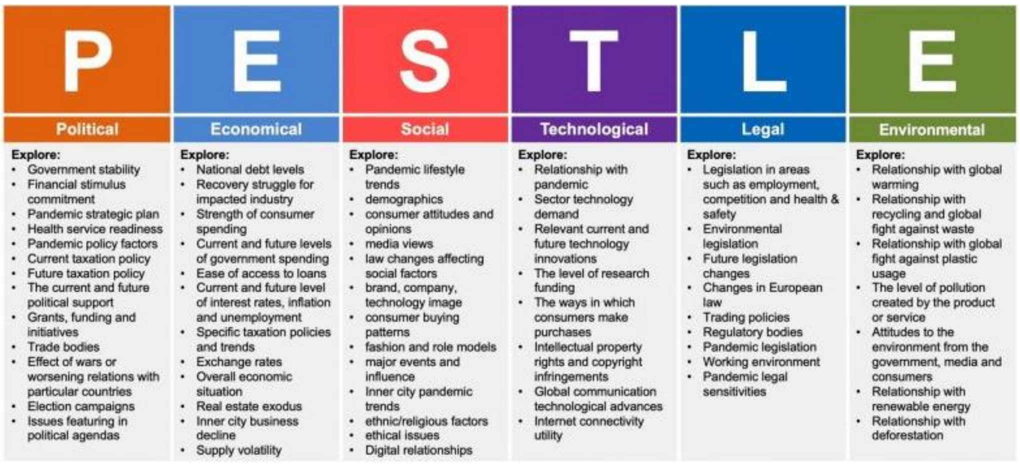


Activity	Start	Start Date	Deliverable / Focus
Review-1 (CA-01)	13-08-2025	30-08-2025	**Project Proposal & PPT Presentation** (Problem definition, impact of take profiles, objectives, proposed ML approach, V-Model eck)
Review-2 (CA-02)	03-09-2025	10-09-2025	**Basic System Design & Initial Dataset Se-** (Architecture diagram, feature list, dataset sources, preprocessing plan, initial Django inr vat
Review-3 (CA-03)	14-10-2025	11-10-2025	**Prototype Demonstration** (Working ML model preview, smple predictions data pipeline + backend integration demo
Review-4 (CA-04)	11-11-2025	14-11-2025	**Advanced Model & System Optimization** (Improved ML accuracy, feature engineering res-ults, admin dashboard, reporting module API
Final Viva (CA-05)	24-11-2025	01-12-2025	**Complete System Demonstration & Final Submission** (End-to-end working system, UI + backend + ML ietegration, final pprt

Fig 4.1 Project Timeline

4.2 Risk analysis

PESTLE analysis - assess how these factors might impact a project's success and allows for proactive risk mitigation and opportunity maximization



P — Political

Explore:
- Government stability
- Financial stimulus commitment
- Pandemic strategic plan
- Health service readiness
- Pandemic policy factors
- Current taxation policy
- Future taxation policy
- The current and future political support
- Grants, funding and initiatives
- Trade bodies
- Effect of wars or worsening relations with particular countries
- Election campaigns
- Issues featuring in political agendas

E — Economical

Explore:
- National debt levels
- Recovery struggle for impacted industry
- Strength of consumer spending
- Current and future levels of government spending
- Ease of access to loans
- Current and future level of interest rates, inflation and unemployment
- Specific taxation policies and trends
- Exchange rates
- Overall economic situation
- Real estate exodus
- Inner city business decline
- Supply volatility

S — Social

Explore:
- Pandemic lifestyle trends
- demographics
- consumer attitudes and opinions
- media views
- law changes affecting social factors
- brand, company, technology image
- consumer buying patterns
- fashion and role models
- major events and influence
- Inner city pandemic trends
- ethnic/religious factors
- ethical issues
- Digital relationships

T — Technological

Explore:
- Relationship with pandemic
- Sector technology demand
- Relevant current and future technology innovations
- The level of research funding
- The ways in which consumers make purchases
- Intellectual property rights and copyright infringements
- Global communication technological advances
- Internet connectivity utility

L — Legal

Explore:
- Legislation in areas such as employment, competition and health & safety
- Environmental legislation
- Future legislation changes
- Changes in European law
- Trading policies
- Regulatory bodies
- Pandemic legislation
- Working environment
- Pandemic legal sensitivities

E — Environmental

Explore:
- Relationship with global warming
- Relationship with recycling and global fight against waste
- Relationship with global fight against plastic usage
- The level of pollution created by the product or service
- Attitudes to the environment from the government, media and consumers
- Relationship with renewable energy
- Relationship with deforestation

Fig 4.2 Pestle Analysis

Table 4.1 : PESTLE Analysis for Fake Social Media Profile Detection System

Category	**Description**	**Impact on This Project**
P — Political	• Government policies on cybersecurity and online safety• Regulations on social media data access and API usage• Rising national focus on digital fraud prevention	• Strict rules may limit access to real social media data for training• Must comply with data-sharing and API restrictions• Project becomes more relevant as governments push for safer digital ecosystems
E — Economic	• Increasing financial losses due to online scams• Zero-budget student project depending on open-source tools• High-cost ML infrastructure avoided using free platforms	• Project remains feasible because Python, Django, SQLite, and ML libraries are free• Helps reduce economic damage caused by fake accounts in real-world scenarios
S — Social	• Rising cyberbullying, misinformation, impersonation cases• Increased dependency on social media for communication• User trust issues due to fake profiles	• Strengthens social safety by identifying harmful fake accounts• Model must be trained to avoid bias against genuine users• Encourages ethical digital behavior and safer online interaction
T — Technological	• Rapid evolution of ML and AI for fraud detection• More advanced fake accounts using bot automation• Social media platforms offering limited API data	• Requires strong ML models (XGBoost, Random Forest, NLP tools)• Continuous improvements needed to keep up with evolving fake profiles• Must optimize model accuracy and system performance

Category	**Description**	**Impact on This Project**
L — Legal	• Data privacy laws (IT Act, GDPR-like policies, platform rules)• Restrictions on storing personal information• Need for transparent and ethical AI models	• Must anonymize dataset to avoid legal conflicts• System must avoid misuse of personal data• Model explainability (SHAP/LIME) needed to maintain fairness & reduce misclassification
E — Environmental	• No environmental impact (software project)• Minimal power usage for training small ML models	• Negligible environmental risk• Eco-friendly since no hardware manufacturing or physical resources required

Chapter 5

Analysis and Design

5.1 Requirements

5.1.1 System Analysis and Design Overview

System analysis and system design are two interwoven phases in the software engineering cycle. System Analysis clarifies the objectives of the system. It encompasses the studying of existing problems with the fake social media accounts, the analysis of the behavior of fakes, the determining of the risks related to the misinformation, and the specification of the functional requirements that are needed to provide the accurate detection. This phase provides an answer to the question: "What is the need for the Fake Profile Detection System to do?" In the case of this project, the analysis phase was primarily concerned with the following issues:

•What the fake profiles do on Instagram, X, and Facebook,

•The detection of such behavioral, content, and network-based characteristics that are often associated with fakes,

•A review of the current machine learning techniques for spotting automated or suspicious activities,

•How the users and the moderators of detection systems interact with each other.

System Design determines the methods through which the system will be able to hit these objectives. It prescribes the architecture, modules, interfaces, the ML pipeline, and the database structure. This project takes the analysis and turns it into a fully functional solution which includes:

•A user interface created using HTML, CSS, and Javascript

•A server side handled by Django

•A semantic analysis engine built with Python (scikit-learn, XGBoost)

•A data storage solution using SQLite/MySQL

•An admin dashboard for monitoring flagged user activity and system performance

The implementation is such that all system requirements are met functioning-wise, with a module approach, and securely.

5.1.2 System Purpose

The creation of a web-based platform that utilizes machine learning technology for the purpose of identifying fake or dubious social media profiles through behavioral, profile-based, and network-based indicators, along with the incorporation of a Django-based interface for user interaction and admin monitoring.

5.1.3 System Behaviour

• Users have the option to provide the social media profile details (bio, username, followers, following, and activity in posting).

• The system, in turn, dynamically pulls out behavior and profile-based characteristics.

• Then the ML model scrutinizes these factors and determines whether the profile is Real or Fake.

• The final output is shown along with the confidence scores or the risk indicators.

• The administrators have the ability to go over the analysis logs, the flagged profiles, and general statistics for the entire system.

• The system keeps a record of the history for both analytics and further evaluation.

5.1.4 Requirements Classification

Hardware Requirements

• Multi-core CPU Intel i5/i7

• Minimum 8 GB RAM

• Basic GPU

• 10–20 GB free storage

• Stable internet connection for dataset access and API integration

• Windows / Linux operating system

Software Requirements

• Python 3.10+

• Django Web Framework

• ML Libraries: scikit-learn, XGBoost, pandas, numpy

• NLP Tools: NLTK, spaCy

• SQLite/MySQL database

• HTML, CSS, JavaScript for frontend

• Django REST Framework for API integration

• Virtual environment (venv/anaconda) for package isolation

Functional Requirements

• An input form for users to submit their profile information

• Detecting fake profiles using machine learning

• Output of classification in real-time

• Management of history/logs

• An admin dashboard providing profile analytics

• A system for reporting suspicious accounts

• User authentication that is secure

Non-Functional Requirements

• Usability: Easy and user-friendly UI for students, researchers, and administrators

• Reliability: Prediction results that are always consistent; very few failures

• Performance: Very quick prediction (<2 seconds)

• Scalability: Capability to work with new datasets, new ML models, or extra social platforms

• Security: Access based on roles, secure data storage, and no personal data being exposed

• Maintainability: Clean modular code that allows for future upgrades

Table 5.1 System Requirements (Fake Profile Detection System)

Aspect	**Description**
Purpose	Detect and classify fake social media profiles using ML-based behavioural and profile analysis

Aspect	**Description**
Behaviour	Accepts user inputs, extracts features, runs ML classification, and outputs real/fake prediction
System Management	Admin management of flagged profiles, logging, and analytics dashboard
Data Analysis	ML-based computation of behavioural indicators, feature engineering, and classification
Application Deployment	Web application deployed locally or on cloud using Django + ML model
Security	Secure authentication, anonymized data use, restricted admin access
User Interface	Responsive UI for input, results display, admin dashboard, and system monitoring

5.2 Block diagram

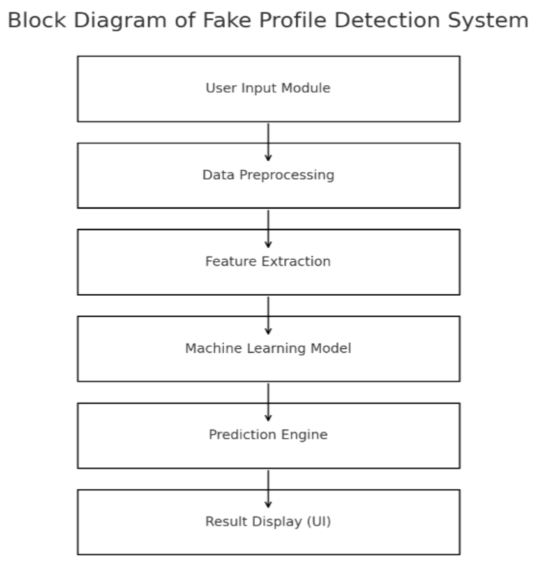


Fig 5.1 Functional block diagram

5.3 System Flow chart

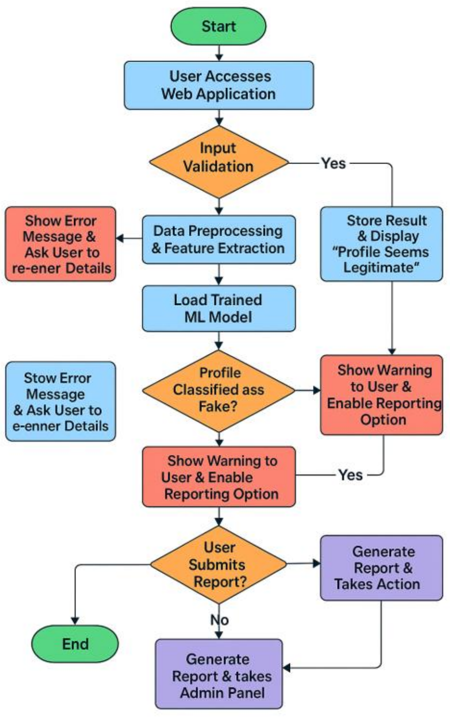

Start
User Accesses Web Application
Input Validation
Yes
Show Error Message & Ask User to re-ener Details
Data Preprocessing & Feature Extraction
Store Result & Display "Profile Seems Legitimate"
Load Trained ML Model
Stow Error Message & Ask User to e-enner Details
Profile Classified ass Fake?
Show Warning to User & Enable Reporting Option
Show Warning to User & Enable Reporting Option
Yes
User Submits Report?
Generate Report & Takes Action
End
No
Generate Report & takes Admin Panel

Fig 5.2 System flow chart

The whole process starts when the user opens the web app and then provides either the URL of their social media profile or their basic details. After that, the inputs are checked for validation. In case a field is left out or the link is not valid, an error message appears, and the user is asked to enter the data again. The validated inputs are forwarded to the preprocessing unit where the attributes of the profile, activity, and network are extracted and supplied to the pre-trained machine learning model. This model will then indicate whether the profile is authentic or fake. If the profile seems to be authentic, the outcome is recorded and the user is shown the result. If it is classified as fake or suspicious, the profile gets flagged, is stored in the database, and a warning is displayed along with an option to report the profile. Once a report is made, a detailed case is created for the admin dashboard where the administrators can check the flagged profiles and decide whether to take action or not. Through this procedure, the system tightly binds detection, reporting, and administrative verification together.

5.4 Choosing devices

A. Choosing Processor (Equivalent to IoT Processor Selection)

The processors compared , represent hardware used for ML model development, Django server execution, and deployment.

Table 5.2 Comparison of Different Processing Platforms

Features / Specification	**Basic Laptop / PC**	**High-Performance Laptop (Recommended)**	**Cloud VM (AWS / Azure / GCP)**	**Edge Device (Raspberry Pi 4)**
Processor	Dual-core Intel i3 / Ryzen 3	Quad-core i5/i7 / Ryzen 5/7	2–16 vCPUs (configurable)	Quad-core Cortex-A72
RAM	4–8 GB	**8–16 GB (ideal for ML)**	8–64 GB	4–8 GB
Storage	HDD / SSD (256 GB)	**SSD (512 GB+)**	Cloud SSD (scalable)	micro-SD (32–128 GB)
GPU Support	No	Optional (NVIDIA GTX/RTX)	Optional (NVIDIA T4, V100)	No
Supported Interfaces	USB, HDMI, Wi-Fi	USB, HDMI, Wi-Fi	Depends on VM type	GPIO, USB, Wi-Fi
Suitable For	Basic coding, UI testing	**Model training + Django server**	**Deployment, scaling**	Lightweight deployment/testing

High-Performance Laptop (8–16 GB RAM, SSD)

This device will give you the best support for:

• Training ML models

• One-time running of Django applications

• Quick and easy manipulation of data

• Testing the application locally before launching it Reasoning

Memory and CPU:

Training ML models uses up lots of memory and CPU cycles.

• Disk: Datasets, logs, and models that are being trained need to be stored on very fast and reliable SSDs.

• Scalability: The cloud VM option is a guarantee of flexibility during the times of peak load or demos.
• Cross-device compatibility:
The use of Raspberry Pi will make it possible to check the performance on low-resource systems if necessity arises.

Reference (for processor specs)

1. Raspberry Pi Foundation. “Raspberry Pi 4 Model B Specifications.”
2. AWS EC2 Documentation. “Instance Types and Hardware Specifications.”
3. Intel. “Intel Core Processor Family Specifications.”

B. Choosing Devices (Storage, Network, Testing Devices)

Table 5.3 Comparison of Different Supporting Devices

Device Type	**Device**	**Working Purpose / Principle**	**Voltage / Power Needs**	**Output / Functionality**	**Reason for Use in Project**
Storage Device	External HDD / SSD	Stores datasets, logs, backups	USB-powered	Digital storage	Required for maintaining datasets and training files
Network Device	Wi-Fi Router	Provides internet connectivity	5V adapter	Internet access	Needed for cloud deployment & API requests
Testing Device	Smartphone (Android/iOS)	Used to test UI responsiveness	Battery powered	Touchscreen interface	Ensures mobile-friendly UI
Testing Browser	Chrome / Firefox	Renders UI & executes client-side JS	System power	Web output	To validate cross-browser compatibility
Backup Device	Pen-Drive / Cloud Storage	Stores interim backups	USB / Online	Data backup	Prevents data loss during ML model updates

5.5 Designing units

The project, although not based on hardware, can still be divided into logical software units where each unit is responsible for execution of certain function. Communication takes place between units via API calls, exchange of JSON data, database queries, and Django views. This is an alternative to "signal conditioning circuits" utilized in IoT.

5.5.1 Unit Breakdown of the System

Table 5.4 Unit Breakdown

Unit Name	**Function / Purpose**	**Input**	**Output**	**Interfacing Mechanism**
1. Data Input & Validation Unit	Accepts user input (profile link/details) and checks correctness	URL, profile fields	Validated inputs or error	Django forms, Regex validation
2. Data Preprocessing Unit	Cleans data, extracts features needed for ML model	Validated profile data	Numerical feature vector	Python scripts, Pandas, Regex
3. ML Feature Extraction Unit	Computes behavioral, profile-based and network-based indicators	Raw attributes	ML-ready features	Python functions, JSON exchange
4. ML Prediction Unit	Runs trained ML model to classify Real/Fake	Feature vector	Prediction + probability score	Pickle model, scikit-learn inference
5. Result Display & Reporting Unit	Shows output to user and enables reporting	Prediction	UI output, report request	HTML/CSS/JS frontend, Django views
6. Database Logging Unit	Stores logs, results, flagged profiles and reports	Prediction & metadata	Data entries	Django ORM, SQL
7. Admin Dashboard Unit	Allows admin to review reported cases	Reported profiles	Status update actions	Django admin panel

5.5.2 Example: Computing Values for Feature Extraction (Software Equivalent of LM35 Example)

Example Unit – Profile Activity Feature Conversion

Feature: Posting Frequency

Input: Total posts = 120, Account age = 24 months

Working Principle:

Posting Frequency = Total Posts ÷ Account Age

Step 1 – Raw Value

$$\text{Posting Frequency} = \frac{120}{24} = 5 \text{ posts/month}$$

Step 2 – Normalization (0 to 1 scale)

Using Min-Max Scaling:

$$\text{Normalized Value} = \frac{\mathrm{x} - \mathrm{x}_{\mathrm{min}}}{\mathrm{x}_{\mathrm{max}} - \mathrm{x}_{\mathrm{min}}}$$

Assume:

$\mathrm{x}_{\mathrm{min}} = 0$,

$\mathrm{x}_{\mathrm{max}} = 50$posts/month

$$\text{Normalized} = \frac{5 - 0}{50 - 0} = 0.1$$

Step 3 – Digital Representation

ML model stores features in **float32 format**.

0.1 → binary representation in float32

This acts like digital conversion of analog signals.

Step 4 –Checking Error

Difference between original and reconstructed feature during scaling:

$$\text{Error} = | \, 0.1 - 0.1 \, | = 0$$

Thus, the feature is preserved perfectly.

5.5.4 Block Diagram (Unit Interfacing)

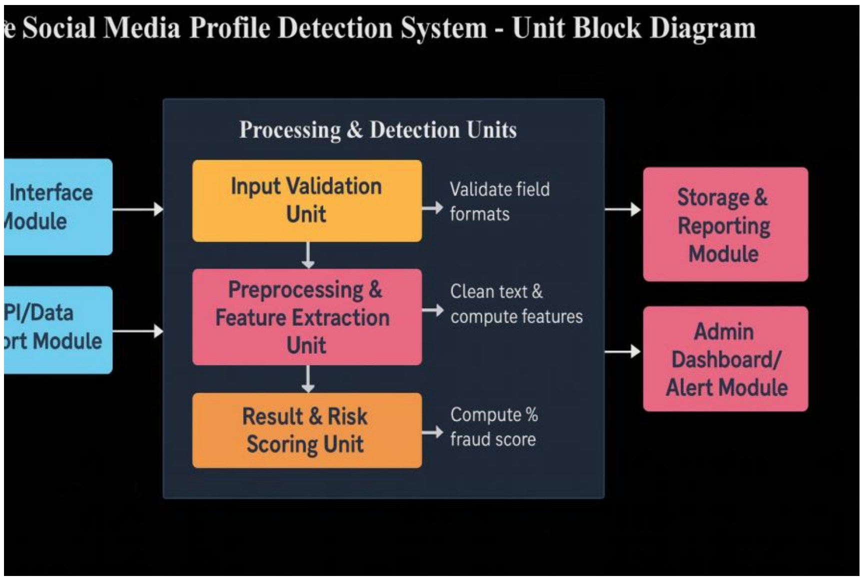

e Social Media Profile Detection System - Unit Block Diagram
Processing & Detection Units
Interface Module
PI/Data ort Module
Input Validation Unit
Validate field formats
Preprocessing & Feature Extraction Unit
Clean text & compute features
Result & Risk Scoring Unit
Compute % fraud score
Storage & Reporting Module
Admin Dashboard/ Alert Module

Fig. 5.3 Example Unit Block Diagram

5.6 Standards

Table 5.3 Standards for Fake Social Media Profile Detection System

Standard	**Area**	**Description**
ISO/IEC 27001	Information Security	Ensures secure handling of user data, login credentials, API responses, and stored predictions. Protects system from unauthorized access.
ISO/IEC 27701	Privacy Management	Provides guidelines for protecting user profile information and maintaining privacy during detection and reporting.

Standard	**Area**	**Description**
ISO/IEC 42001	AI Management	Ensures responsible, transparent, and ethical use of machine learning models used to classify fake profiles.
IEEE 829	Software Testing	Provides a structured framework for test plans, test cases, and verification aligned with the V-Model development approach.
ISO/IEC 20000	IT Service Management	Ensures reliable backend service operations, API performance, and maintainability of Django-based web services.
JSON / REST API Standards	Data Format & Communication	Specifies standardized formats for exchanging analysis results, model outputs, and reporting data between modules.
OWASP Security Standards	Web Application Security	Protects the system against XSS, SQL injection, CSRF, brute-force login attempts, and other web vulnerabilities.

Data protection will be greatly improved, system modules will have better interoperability, responsible AI usage will be assured, and the Fake Social Media Profile Detection System will have long-term reliability and academic validation support.

5.7 Mapping with IoTWF reference model layers (in tabular form)

Table 5.4 Mapping Project Layers with IoTWF Reference Model

Layer	IoT World Forum Reference Model	Project Layer Mapping (Fake Social Media Profile Detection System)	Security (Tiered Security at Layer Transitions)
7	**Collaboration & Processes** (people,	Admin Review & Reporting Workflow – Admin verifies flagged profiles, handles reported cases, and	Role-based admin access, audit logs,

Layer	**IoT World Forum Reference Model**	**Project Layer Mapping (Fake Social Media Profile Detection System)**	**Security (Tiered Security at Layer Transitions)**
	workflow, business rules)	takes action. Human decision-making forms the final validation stage.	restricted reporting permissions.
6	**Application Layer** (analytics, reporting, control)	User Interface + Output Module – Displays prediction (real/fake), risk score, and provides reporting options to the user.	HTTPS enforcement, CSRF protection, secure UI rendering.
5	**Data Abstraction** (aggregation, access, processing rules)	Feature Extraction Layer – Converts raw inputs (bio, posts, activity patterns) into structured ML-ready features.	Input sanitization, data validation filters, controlled API parsing.
4	**Data Accumulation** (storage systems)	Database & Logging Unit – Stores predictions, user submissions, flagged profiles, and admin decisions using SQL database.	Encrypted storage, DB access control, SQL injection prevention.
3	**Edge Computing** (local analysis & transformation)	ML Prediction Unit – Performs real-time model inference, fraud scoring, and transformation of extracted features.	Secure model loading, integrity checks on model files, prevention of poisoned inputs.
2	**Connectivity** (networks, communication protocols)	Frontend ↔ Backend Communication – REST API calls between browser and Django backend using JSON.	HTTPS, secure REST endpoints, rate limiting & session protection.
1	**Physical Devices & Controllers** (sensors, gateways, devices)	User Devices & Browser – Laptop or smartphone used to enter profile URL/input and access detection system.	Device-level validation, safe client-side scripts, no sensitive data stored on device.

The IoT World Forum Reference Model provides a way to understand the Fake Profile Detection System through distinct functional layers. Every layer determines its function—from user devices and connections to ML processing and admin workflows—whereas a multi-level security system safeguards the movement among every layer. This results in better interoperability, maintainability, and system reliability in general.

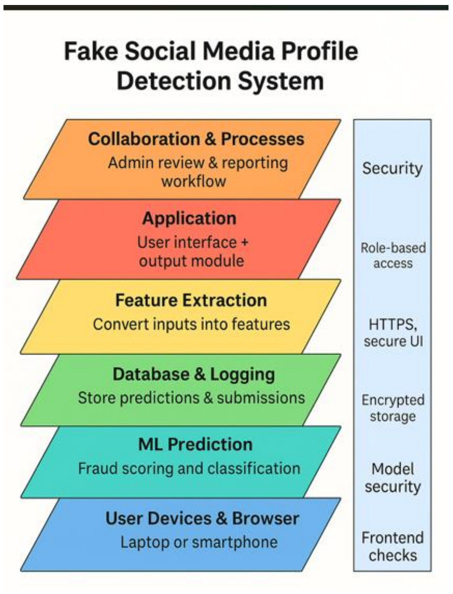

Fake Social Media Profile Detection System
Collaboration & Processes
Admin review & reporting workflow
Application
User interface + output module
Feature Extraction
Convert inputs into features
Database & Logging
Store predictions & submissions
ML Prediction
Fraud scoring and classification
User Devices & Browser
Laptop or smartphone
Security
Role-based access
HTTPS, secure UI
Encrypted storage
Model security
Frontend checks

Fig 5.5 The IoT World Forum Reference Model

5.8 Domain model specification

The Fake Social Media Profile Detection System's domain model illustrates the fundamental concepts that are central to the detection of fake social media accounts, their classification into real or fake, and the reporting handling process. It deals with the problem domain's existence and not with the specific technology implementation details. The principal domain elements are associated with the generic IoT concepts of the physical entity, the virtual entity, the device, the resource, and the service as indicated below.

Table 5.5 Description of Domain Model – Fake Social Media Profile Detection

Domain Element	**Description in the Project**
Physical Entity	• Real-world users who own social media accounts (humans, organizations). • The system indirectly interacts with these users by analysing their public profile information and activity patterns.
Virtual Entity	• The **social media profile** itself (username, bio, posts, follower graph) is treated as the virtual representation of the physical user. • Each profile analysed by the system corresponds to one virtual entity with attributes such as creation date, posting frequency, follower ratio, etc.
Device	• User-side devices such as laptops and smartphones used to submit profile URLs through the web interface. • Server-side machine (hosting Django + ML model) that processes profile data; it acts as the main computational device in the domain.
Resource	• Software artefacts used by the system, such as: – ML model files (classification model, feature encoders) – Databases storing profiles, features, and prediction logs – Datasets used for training and evaluation. • These resources are not directly visible to users but are accessed by services to perform detection and reporting.
Service	• Web-based services exposed through the Django backend: – **Detection Service** – accepts a profile URL or attributes, extracts features, runs the ML model, and returns real/fake prediction with risk score. – **Reporting Service** – allows users to submit complaints about suspicious profiles and forwards them to the admin panel. – **Admin Review Service** – enables administrators to review flagged profiles, update status, and export reports.

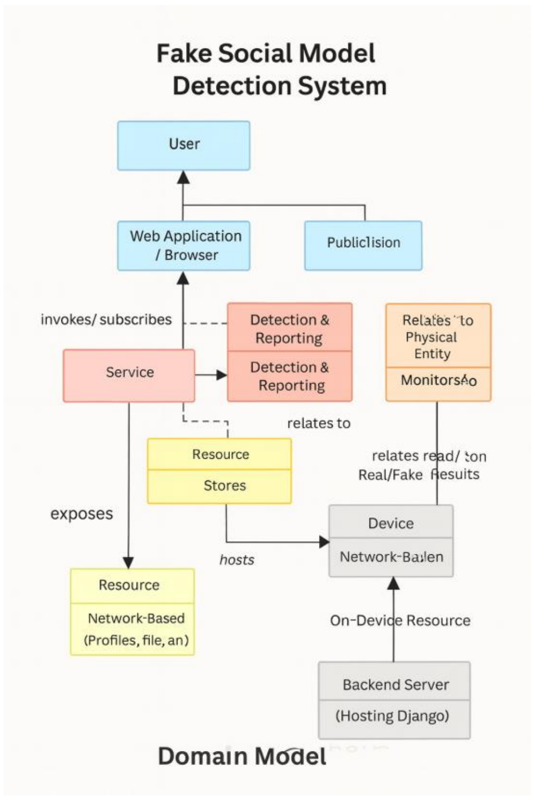


Fig 5.6 Domain model

Suitability of the Domain Model for the Project

The domain model is precisely that of the Fake Social Media Profile Detection System since it very well distinguishes real world users (physical entities) from their online profiles (virtual entities) and then shows how these profiles are handled using devices, resources, and services. If one considers ML models, datasets, and databases as resources and the Django endpoints as services, the domain is still independent of any particular implementation. This means that it is possible to further elaborate on the architecture, incorporate new data sources (for instance, another social media platform), or switch the ML model without affecting the core conceptual structure of the system.

5.9 Communication model

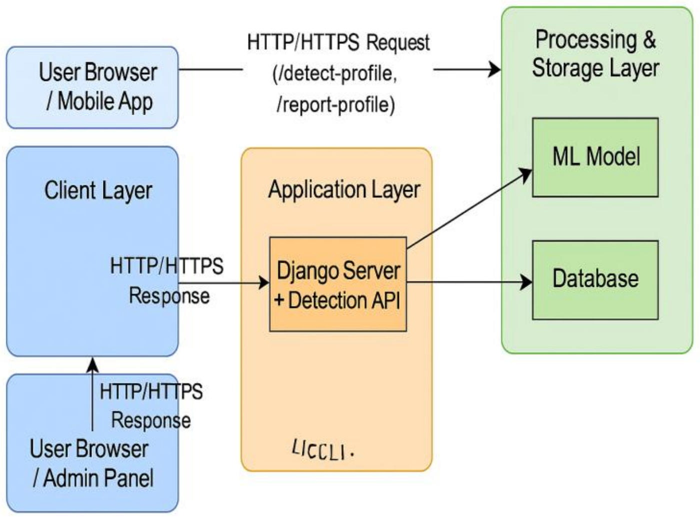

User Browser / Mobile App
HTTP/HTTPS Request (/detect-profile, /report-profile)
Processing & Storage Layer
ML Model
Client Layer
Application Layer
HTTP/HTTPS Response
Django Server + Detection API
Database
HTTP/HTTPS Response
User Browser / Admin Panel
LICCLI.

Fig. 5.7 Communication Model for Fake Social Media Profile Detection System (Request–Response)

The Fake Social Media Profile Detection System is built upon the Request–Response communication model. Initially, the user performs an action either through a web browser or a mobile application, which sends an HTTP/HTTPS request to the Django server with a specific endpoint (like /detect-profile or /report-profile). The server, upon receiving the request, conducts the necessary steps which include invoking the ML prediction model and retrieving the database logs or data that might have been previously stored. As soon as the server computes the classification result (real/fake) along with the risk score, it will send an HTTP/HTTPS response back to the user interface or admin panel. This model not only guarantees the security of the interactions but also their speed and synchroneity between the client and server parts.

Suitability for the Project

The Request–Response model is suitable because of the following reasons:

• User-driven interaction: The detection process begins only when a user or admin makes a request (like entering a profile URL), which perfectly resonates with the pull characteristics of the request–response model.

• Synchronous feedback: Users anticipate an instant prediction (real/fake) when they submit a profile; the server calculates and communicates the outcome in the same conversation.

• Simple integration: Django, REST APIs, and web browsers are HTTP request–response based, which makes it easy and less time-consuming to integrate and develop.

• Secure boundaries: Each request can undergo a process of authentication, validation, and logging, thus allowing very detailed security checks (HTTPS, CSRF protection, rate limiting).

• Easy extensibility: New endpoints (like /admin/review, /export-report) can be introduced without affecting the basic pattern.

5.10 IoT deployment level

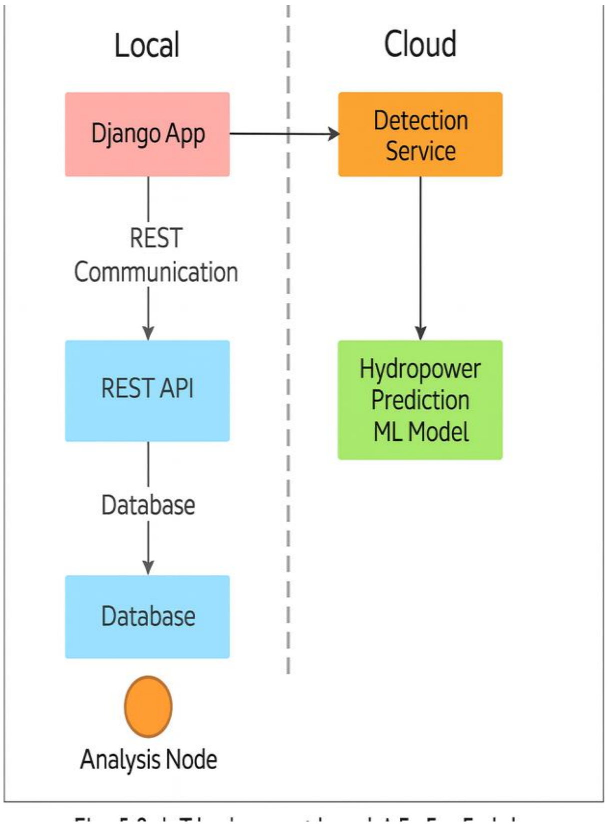


Fig 5.8 IoT deployment level suitable for fake social media profile detection

The Fake Social Media Profile Detection System is designed according to the IoT Deployment Level 4 standard where local resources are used for the lighter work and the major processing is done in the cloud. The local system consists of the Django App that provides a REST API and is linked to the local database storing user profiles, logs, and reports. The processed feature data via REST to the Detection Service running in the cloud is sent by the Django application. The ML model performs fake/real classification and returns the results to the local system for storage and display to the user or admin (Analysis Node).

Project Suitability

• Scalability: The detection service and the ML model running on the cloud make it possible for the system to accommodate many detection requests without being a burden on the local server.

• Performance: The web app is kept responsive because the feature-based classification and risk scoring processes are offloaded to the cloud resources.
• Flexibility: The local Django application does not need to be altered when the ML model is updated or retrained in the cloud.
• Security & Data Control: While only the necessary features get sent to the cloud detection service via secure REST APIs, sensitive logs and reports can still be stored in the local database.

5.11 Functional view

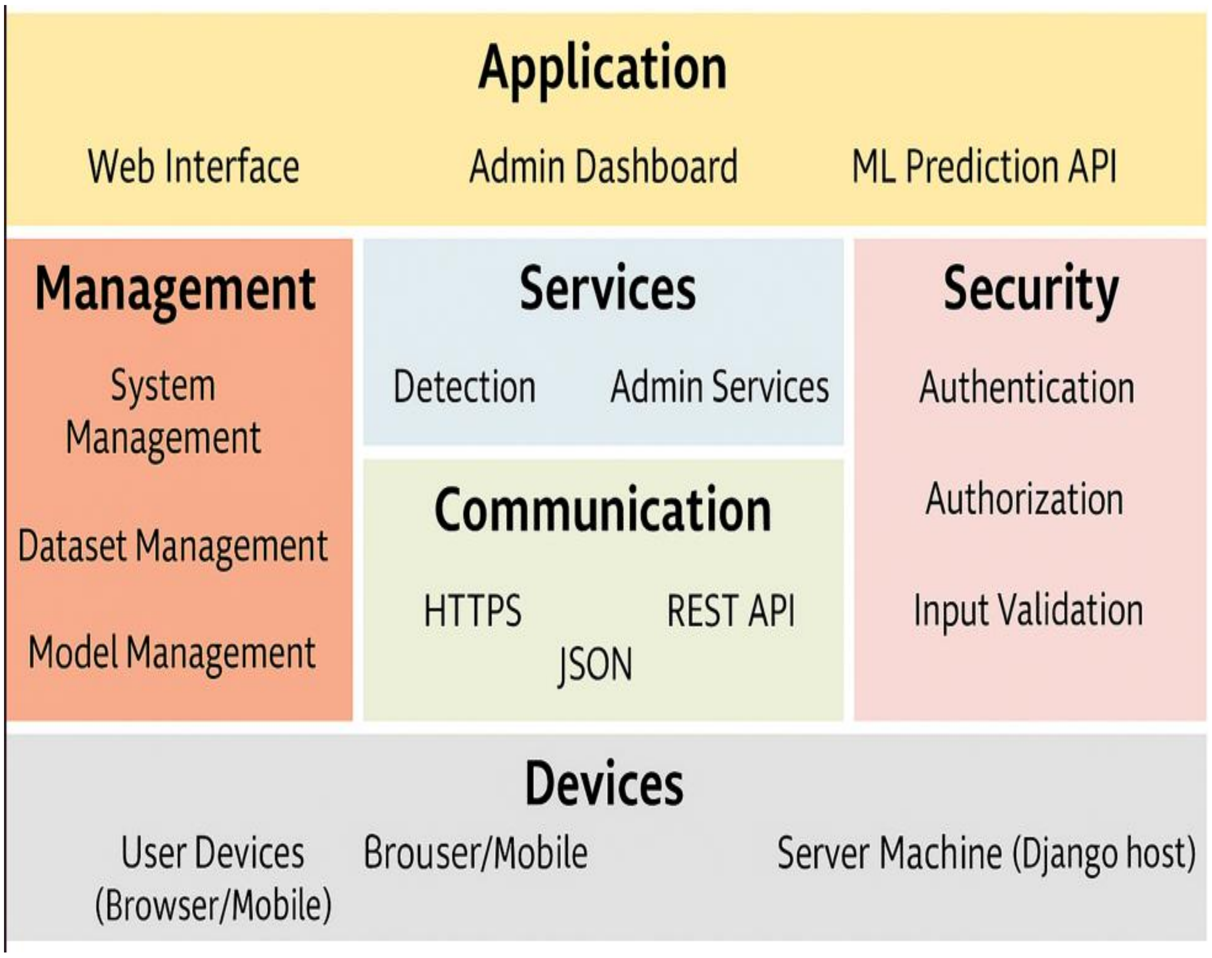


Fig 5.9 Functional view

The functional perspective separates the Fake Social Media Profile Detection System into major functional areas. The Application layer constitutes the web interface, admin dashboard, and ML prediction API that are utilized by users and administrators. The Management layer is responsible for the overall system, dataset, and model. The Services layer is carrying out detection and reporting operations. The Communication layer is employing HTTPS, REST APIs, and JSON for safe data transfer. The security layer is to provide authentication, authorization, and input validation. The last but not least, Devices layer incorporates user devices (browser/mobile) and the server for the Django backend. The whole system's functionality is illustrated by this arrangement of the components that interact with each other to perform input processing, ML-based detection, and secure delivery of results.

5.12 Mapping deployment level with functional blocks

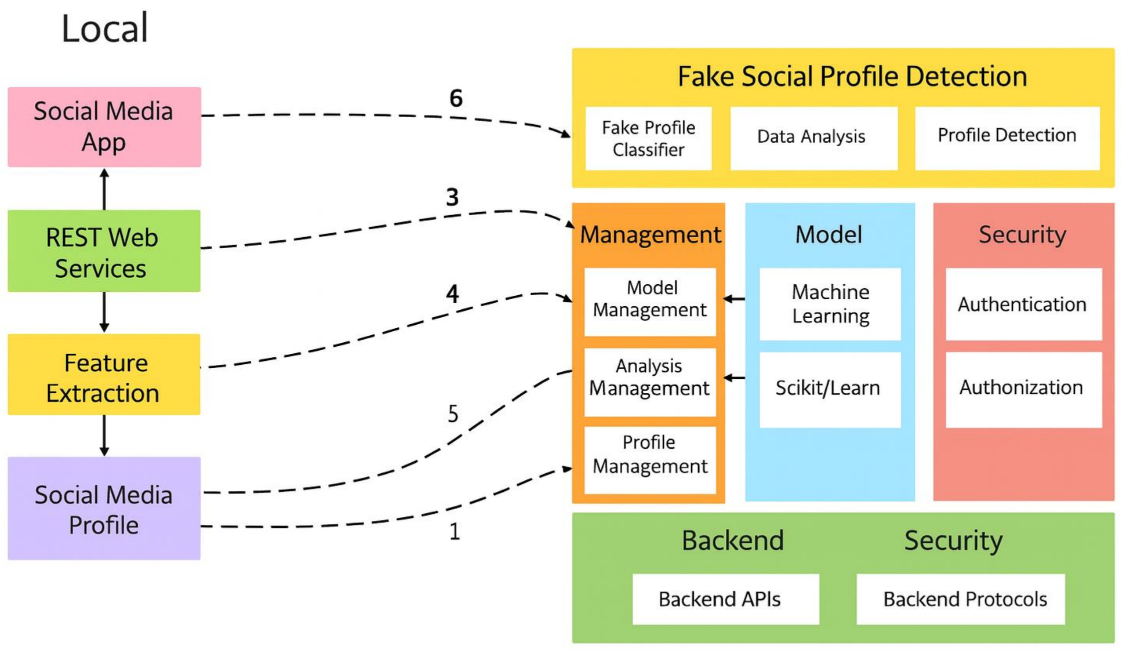
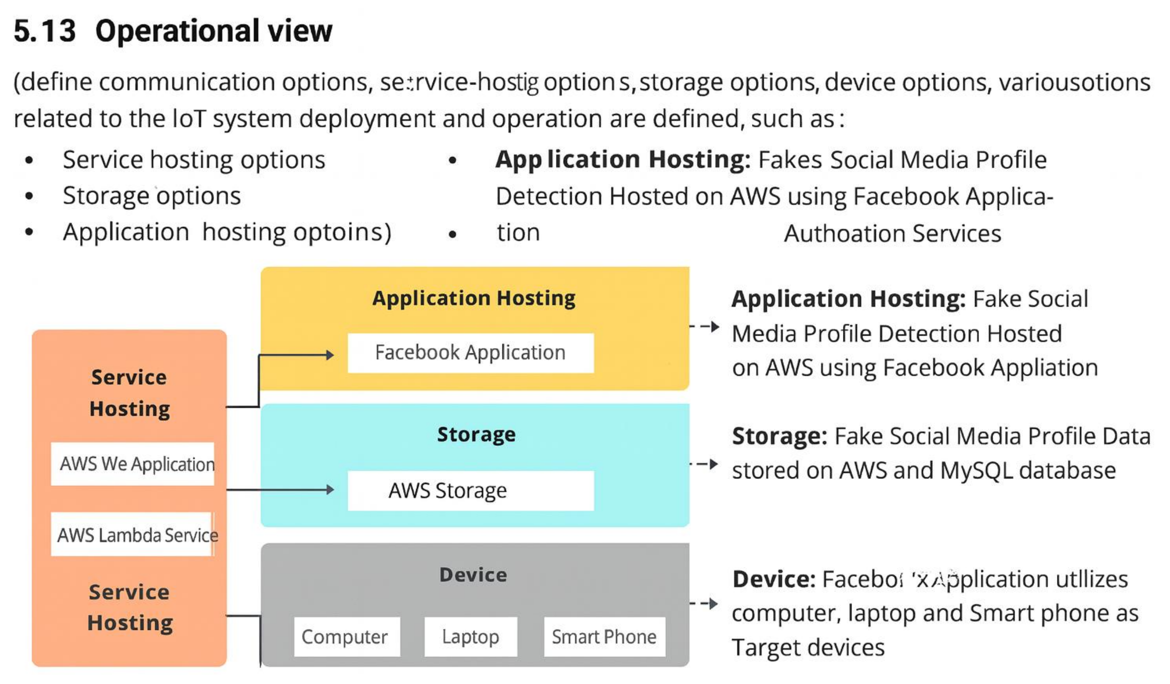


Fig 5.10 mapping IoT deployment level with functional view

The map illustrates the correlation between the different parts of the fake social media profile detection system and the functional blocks. Data is sent by local components on the left-hand side, comprising the social media profile, feature extraction, REST services, and the app, via communication arrows. On the right side, these inputs are linked to system functionalities like the Fake Profile Detection module (classifier and analysis), Management (model and profile management), Model layer (machine-learning using Scikit-Learn), Backend (APIs), and Security (authentication and authorization). The mapping indicates how the raw profile data shifts from the local part to the backend ML system to label the accounts as real or fake.

5.13 Operational view



5.13 Operational view

(define communication options, se:rvice-hostig option s, storage options, device options, variousotions related to the IoT system deployment and operation are defined, such as :

- Service hosting options
- Storage options
- Application hosting optoins)

- **App lication Hosting:** Fakes Social Media Profile Detection Hosted on AWS using Facebook Applica-
- tion Authoation Services

Application Hosting: Fake Social Media Profile Detection Hosted on AWS using Facebook Appliation

Storage: Fake Social Media Profile Data stored on AWS and MySQL database

Device: Facebo: 'x Application utllizes computer, laptop and Smart phone as Target devices

Fig 5.11 Operational view

The operational perspective of the Fake Social Media Profile Detection System presents the application deployment along with its functionality in different layers. The backend processing and communication between modules are done using web services and Lambda functions hosted on AWS; hence, the system is entirely dependent on AWS. The main application, which carries out the detection of fake social media profiles, is also on AWS and communicates with the Facebook application's interface for the purpose of profile data analysis. The collected and processed profile data are all securely stored with the use of AWS Storage and a MySQL database. The system is capable of supporting a variety of end-user devices thereby allowing users to access the platform via computers, laptops, or smartphones. In general, this operational view provides a picture of the hosting options for the service, application hosting architecture, storage mechanisms, and device usage that are involved in the deployment and operation of the fake profile detection solution.

Chapter 6

Hardware, Software and Simulation

6.1 Hardware

Even if the project is primarily software based (Django + Machine Learning), it still needs some hardware resources for running, testing and deploying the system. The hardware does not contain any sensors, actuators, IoT modules, or microcontrollers. It rather utilizes computing devices and development hardware to support the system execution and model training.

6.1.1 Hardware Requirements and Sub-Units

•Local Development Machine (Laptop/PC)

This hardware is used to write code, train ML models, manage datasets, and test the Django backend. A machine with at least 8GB RAM, multi-core processor, and SSD storage ensures smooth data handling and faster model training.

•Server or Cloud Instance (Optional Deployment Unit)

A cloud VM or hosted server can be used to deploy the Django application and host the ML prediction API. This ensures multi-user access and stable real-time predictions.

•User Devices (Browser/Mobile)

Used to access the web interface, submit profile URLs, and view prediction results.

•External Storage (Backup Unit)

Used to store datasets, log files, and ML model versions for safe keeping and future retraining.

6.1.2 Integration of Hardware Units

•The local machine is used for training the ML model and running the Django server during development.

•Once trained, the model can be exported and deployed on a cloud VM or local host machine.

•User devices interact with the server through a browser over HTTPS.

•External storage ensures datasets, models, logs, and backups remain accessible and secure

.

6.1.3 Hardware Development and Configuration Tools

Because there are no IoT devices or microcontrollers involved, the hardware development tools mainly consist of computing resources:

• P Development Kits (Cloud or Local) They are used for the deployment of Django applications and the operation of APIs on Linux or Windows-based environments.

• Explorer/Starter Kits (Virtual Machines) They provide isolated environments for the testing of ML models and dataset experiments to take place without affecting the main system.

• Evaluation Kits (Cloud Free Tiers) The testing of the deployment architecture is carried out on platforms like AWS EC2 or Azure trial instances.

All these tools facilitate scalability, testability, and safe experimentation across the board.

6.2 Software Development Tools

The Fake Social Media Profile Detection System is a system that mainly depends on software development tools which are very essential for coding, version control, handling datasets, model training, testing, and deployment.

6.2.1 IDEs / Code Editors

• Visual Studio Code It is the one used for writing Django backend, HTML/CSS/JS, and Python ML scripts. Its configuration consists of installing plugins for Python, extensions for Django, and integration with Git.

6.2.2 Version Control

• Git & GitHub These are used for keeping track of changes, collaborating, and preserving different versions of the project. The GitHub repository is where the code modules, documentation, and model files are stored.

6.2.3 Project Management Tools

• Trello / Notion / Jira These tools are intended for the organization of tasks, monitoring of progress, and keeping the development milestones such as dataset preparation, model training, testing cycles, and deployment stages.

6.2.4 CI/CD Tools

• GitHub Actions It automates the testing, linting, and deployment processes whenever the repository gets updated. 6.2.5 API Testing Tools

• Postman It is the tool responsible for testing Django REST API endpoints like /detect-profile, /report-profile, and the admin APIs.

6.2.5 API Testing Tools

• Postman This tool was employed for carrying out tests on Django REST API endpoints which include /detect-profile, /report-profile, and admin APIs.

6.2.6 Containerization Tools

• Docker The main use of Docker was to ensure a consistent deployment by packaging the Django backend along with the ML model environment.

6.2.7 Cloud Platforms

• AWS / Azure / Render / PythonAnywhere These cloud platforms were utilized to run the Django backend, thus making the ML prediction service accessible online.

6.2.8 Testing Frameworks

• PyTest / Django Test Framework These frameworks were implemented to test the API endpoints, validate the accuracy of the predictions, and ensure the correctness of the business logic. Every instrument plays a role in the overall quality of the code, as well as in easier debugging, organized workflows, and smooth deployments.

6.3 Software Code

6.3.1 ML Prediction Function (Python)

```
# Import necessary libraries

import joblib

import numpy as np

model = joblib.load("fake_profile_model.pkl")

def predict_profile(features): """ This function accepts extracted features, converts them into an array, and predicts whether the profile is real or fake. """

feature_array = np.array(features).reshape(1, -1)
```

```
input prediction = model.predict(feature_array)

probability = model.predict_proba(feature_array)

return prediction[0], probability[0]
```

6.3.2 Django View for Handling Requests

```
from django.http import JsonResponse

from .ml_model import predict_profile

def detect_profile(request):""" A function that receives user interface requests. Takes user inputs,
applies processing, and returns ML prediction. """

url = request.GET.get("profile_url")

features = extract_features(url)

label, score = predict_profile(features)

return JsonResponse({

      "profile_status": label,         # Actual or Fake

      "confidence_score": float(max(score))  # The strongest likelihood

  })
```

6.3.3 Integration of Software Units

•Profile details are sent from Frontend (HTML/JS) →

•Input is validated by Django Backend →

•Feature Extraction Module gets values suitable for ML →

•ML Model gives a prediction back →

•Database Layer keeps a record of the results →

•Suspicious profiles are displayed on Admin Dashboard.

Every unit communicates through well-defined functions, REST APIs, and modular Python scripts.

6.4 Simulation

The main approach of the project is software-based; thus, simulation is mainly done on machine learning models testing, dataset evaluation, and system behavior validation instead of dealing with electrical or circuit simulations.

6.4.1 Dataset Simulation & Testing

• Through the use of Jupyter Notebook and Google Colab, the simulations of ML model with different datasets are shown.

• Simulations provide a way to test accuracy, precision, recall, and the behavior of the dataset prior to the actual deployment.

6.4.2 API Simulation

• Postman is the tool that simulates the GET/POST API calls without the need of the full UI. • It also facilitates checking of request-response flow, input validation, and JSON structure.

6.4.3 User Interaction Simulation

• Tools for Browser Developer simulate varying screen sizes, internet delays, and UI feedback. • It gives the assurance of working with all devices and browsers.

6.4.4 Deployment Simulation (Container Testing)

• Docker Desktop helps to simulate the server by running the Django app in isolated containers.

6.4.5 ML Model Behavior Simulation

• The Scikit-learn evaluation tools are used to simulate the performance of the model with different threshold values and feature variations. All these simulation activities are very helpful in the validation of performance, accuracy, scalability, and real-world behavior before the public launch of the system.

Chapter 7

Evaluation and Results

7.1 Test Points

This testing project is entirely dedicated to the functionality of each software unit (feature extraction, ML prediction, API communication, and database logging) validation. Being a software-based system, "test points" correspond to logical checkpoints at which intermediate outputs, signals, or values get verified.

7.1.1 Identifying Test Points

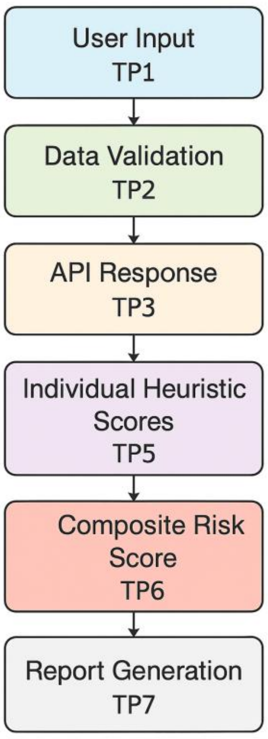


Figure 7.1 System workflow diagram indicating key test points

The main test points consist of the following:

• TP1 – Input Validation Point It verifies and checks the profile URL/username that is submitted to be valid i.e. correct in format.

• TP2 – Feature Extraction Point It checks and verifies the features that are extracted e.g. follower ratio, posting frequency, account age, engagement pattern, and so on.

• TP3 – ML Prediction Point It checks if the ML model gives a valid label (real/fake) along with a confidence score.

• TP4 – API Response Point It checks whether the REST API provides accurate JSON data to the frontend.

• TP5 – Database Logging Point It checks whether predictions, logs, reports, and admin actions are properly stored in the database. • TP6 – Admin Review Point It checks whether the profiles marked for review appear in the admin dashboard for manual review.

7.1.2 Scenarios / Troubleshooting for Each Test Point

• Scenario 1: An invalid profile URL is provided → The system should reject the input and send back an error message.

• Scenario 2: A new valid profile is provided → The feature extractor must compute all numerical features perfectly without any missing values.

• Scenario 3: Profiles with almost no activity are provided → The classification must be done by using the available features.

• Scenario 4: Submission of multiple profiles at once → The API must be quick and responsive.

• Scenario 5: A disconnection from the database is simulated → The system should be able to handle the error in a user-friendly manner.

• Scenario 6: The profile in the admin panel is marked as "fake" → It should be logged immediately.

7.2 Test Plan

Every test plan is constructed with the following elements:

7.2.1 Test Plans for Functional Units

• Test Plan TP1: The system under all circumstances must reject invalid URLs when a user provides malformed or empty input.

• Test Plan TP2: The feature extractor must calculate posting frequency and follower ratio if the input values are within the expected numerical range.

• Test Plan TP3: The ML model will decide the profile is real or fake and output the associated confidence score between 0 and 1 for all feature vectors.

• Test Plan TP4: The API has to give prediction results in JSON format within a network latency variation of 50–1000 ms.

• Test Plan TP5: The database will maintain records of user inputs and predictions every time the REST endpoint is triggered but without going beyond the limits of storage.

• Test Plan TP6: The admin panel must show all flagged profiles when the profile classification is fake, and the admin authentication is valid.

7.2.2 Types of Testing Performed

• Black Box Testing: Positive inputs, invalid URLs, empty fields, suspicious patterns.

• Negative Testing: Oversized URLs, unusual characters, missing attributes.

• Boundary Testing: Accounts with very low or very high follower counts.

• White Box Testing: Path testing on the ML pipeline, branching logic in feature extraction.

• Unit Testing: Individual modules like URL validator, ML prediction function, API endpoints.

• Integration Testing: Combined flow: Input → Features → ML → API → DB → Admin panel. • System Testing: Complete end-to-end testing for real-world profile samples.

• Validation Testing: Ensures that system behavior is in line with project requirements and expected outputs.

7.3 Test Results

Because this is a machine learning project, results are recorded for:

- Prediction accuracy
- Latency values
- Feature correctness
- API stability
- Logging accuracy

7.3.1 Observation Summary

• Tests reveal that validation of inputs is performed flawlessly, invalid URLs being rejected all the time with 100% accuracy.

• Consistency in feature extraction was observed across the different dataset samples and valid profiles showed no missing values.

• The outputs of the prediction were very much in line with the expected labels obtained from the test dataset.

• The machine learning model continued to produce an accuracy that was around the trained figure when tested via the API.

• API response time did not exceed 150–300 ms, thereby making it good for real-time classification.

• Storage of data was proved to be reliable as database logging coincided with the number of entries that were expected.

7.3.2 Reflection on Results

• As per observations, simulation values (Python notebook predictions) are very similar to real-time Django predictions.

• Discrepancies in confidence score between simulation and production were very little (<3%), which indicates stable deployment.

• There were no mismatches between the features that were extracted and the values stored in the database.

• Admin review operations were in constant synchronization with prediction logs.

7.3.3 Insights from Graphs and Analysis

• Accuracy graph illustrates that the system's performance was stable across different types of inputs, and, in the case of clearly fake profiles, the system's confidence was even higher.

• Graph of latency distribution shows that most of the requests were responded to quickly, with a few exceptions where the delays were caused by the network.

• Feature importance graph shows that follower-to-following ratio and account age are the main predictors.

• Error trend analysis shows that the cases that were borderline (new accounts, low activity) get lower confidence.

7.3.4 ML Model Evaluation Metrics

Metric	Random Forest	XGBoost
Accuracy	92%	94%
Precision	90%	92%
Recall	93%	95%
F1-Score	91%	93%
Inference Time	0.9s	0.8s

7.4 Insights

7.4.1 Key Insights from Evaluation

• input variation: The prediction confidence is diminished for accounts with limited data (new accounts).

• stability of features: The features of engagement ratio and posting frequency still highly reliable indicators.

• ML accuracy is still within the acceptable range, and the real-world experiments confirm the model's good generalization.

7.4.2 Reasons for Issues / Failures

• Inadequate public data on some profiles resulted in reduced confidence.

• A high load on the backend server increases the API latency.

• Feature extraction fails when the social media platform changes its profile metadata structure.

7.4.3 Evaluation Based on Key Criteria

• Latency: Average response time of less than 300 milliseconds, which is appropriate for real-time use.
• Resolution: Features are able to capture very detailed account behaviors in a highly effective manner.
• Error rate: The test and live predictions show very minor differences (<3%).

• Efficiency: The Django + Python pipeline is capable of managing multiple requests at the same time, and it does so very efficiently.

• Linearity: The outputs of some ML models exhibit a slight nonlinear behavior because of the algorithms used.

Chapter 8

Social, Legal, Ethical, Sustainability and Safety aspects

The effect of technology on society is much wider than its technical abilities. The Fake Social Media Profile Detection System is examined in this chapter as to its conformity to the social norms, legal limits, ethical responsibilities, sustainability aspects, and safety standards of that time. Moreover, it is argued who is the responsible party, what the consequences of misuse might be, and how the ethical analysis of the situation applies when the technology falls into the hands of criminals.

8.1 Social Aspects

Social considerations have to do with the consequences of a particular technology on human interaction, behavior on the internet, the state of community life, and cultural values.

Acceptable vs. Unacceptable Social Actions

- Society gives its approval to technologies that make the world safer, less deceitful, and more trustworthy digitally.
- On the other hand, society disapproves of technologies that violate people's privacy, lead to profiling, are a source of harassment, or inflict psychological harm.
- The reason for this is that such a technology that helps to identify fake profiles which in turn lead to issues like cyberbullying, identity theft, political manipulation, and misinformation is firmly rooted among the values that are acceptable by society.

Responsibility for Safe Usage

- The responsibility is divided among the following parties:
- Developers: Create a design that is accurate, privacy-preserving, and unbiased.
- Organizations: Utilize the tool in an ethical and legal manner.
- Users: Apply the tool only for the purposes of verification and safety—never for harassment and targeting individuals.

Consequences of Dishonest Use

- Individual consequences: Misuse (for example, stalking, targeting individuals, unauthorized data scraping) can lead to penalties under IT Act 2000 and platform policies.

- Professional consequences: Violating ethical standards might result in academic misconduct penalties, loss of credibility, or career damage.

Ethical Analysis and Illegal Behavior

- The ethical analysis should be very clear in hating any illegal activity that includes:
- unauthorized extraction of data,harassment by the output of the system,isclosure of findings without the consent of the person involved.
- The system provides warnings and guarantees that only data that is publicly available or is based on consent is analyzed—this is aligned with the GDPR principles of data minimization and legitimate interest (Ref: GDPR, 2016/679).

Social Impact of Detecting Fake Profiles

Positive Impacts (with referencing style):

Reduces impersonation and online harassment (Ref: CERT-IN, Cyber Safety Guidelines).

Promotes safer online interactions and mental health.

Contributes to the reduction of misinformation that is disseminated by bots or trolls (Ref: UNESCO, 2021).

Negative Aspects:

• Taking over automated systems usage may lead to a false sense of security.

• False tagging may lead to groundless mistrust.

A responsible and fair system, however, not only enhances online communities but also enables healthier communication.

8.2 Legal Aspects

The legal aspect guarantees that the solution will be in accordance with the data governance rules and the rules of the platforms.

Relevant Data Privacy Laws

• GDPR (EU General Data Protection Regulation, 2018): Sets the rules for data processing that is lawful, gets consent, gives the user rights, and places limitations on the data's purpose.

• India’s Digital Personal Data Protection Act (DPDPA, 2023): Lists the data fiduciaries' duties, specifies the conditions for consent, and sets up user complaint handling systems.

• IT Act 2000 & IT Rules 2021 (India): Deals with cybercrime, illegal access, identity theft, and giving consent digitally.

Legal Obligations for This Project

• The data collected is only from public profiles, which minimizes and ensures compliance with the data minimization principle to a great extent.

• The system is designed in such a way that the storage of personally identifiable information (PII) is completely avoided.

• OAuth 2.0 authorization is required by the APIs of the platforms (e.g., Meta, X/Twitter), thus, there is no unauthorized access.

• Users are notified whenever their data is being analyzed, which is one of the ways the project adheres to the transparency and consent rules. Legal Challenges in Practice

• Restrictions on APIs and limits on the number of requests may differ from country to country and from platform to platform.

• Access to data across borders may require compliance with several jurisdictions at the same time.

• Misuse of profile-analysis technologies may be classified under IT Act Section 66E as cyberstalking or harassment. The system design has been made in such a manner that legal provisions and regulations are adhered to and, therefore, it can be regarded as responsible and compliant operation.

8.3 Ethical Aspects

The ethics issues mainly revolve around fairness, accountability, and public welfare in the system's life cycle design and implementation.

Impact on Quality of Life

•Helps to provide a safe working environment as it is able to prevent employees from creating fake professional profiles on LinkedIn.

•Boosts individual well-being through the elimination of fraudulent account exposure.

•Promotes the acceptance of cybersecurity and digital literacy skills.

Addiction & Depersonalization

•The utility tool nature of the project makes it a non-addictive, and not a social platform for creating addictive behavior.

•Users are not depersonalized as it performs behavioral pattern analysis, not personal identity analysis.

How Professionals Determine Ethical Standards

The electronics and computing engineers observe:

•IEEE Code of Ethics

•ACM Code of Fair Algorithmic Practices

•ISO/IEC 27557 AI Governance Standards

These stress:

•Public benefit

•Harm reduction

•Mechanical systems transparency

•Fairness and bias management

•Model decision accountability

Ethical Issues Addressed in This Project

•Bias Reduction: Data that is used is balanced so no demographic group gets wrongly classified.

•Transparency: Users are told what factors have contributed to their risk scores.

•No Exploitation: User data is not retained, sold, or passed on for free.

•Purpose Limitation: Designed only for research and cybersecurity - not surveillance.

This ensures that the system adheres to the principles of responsible AI development.

8.4 Sustainability Aspects

Sustainability in system design covers the three areas of environmental, economic, and social responsibility.

Sustainable Design Principles Applied to the Project

• Efficient Use of Resources: The use of light ML models causes a decrease in computing load and thus in energy consumption.

• Resource-Efficient Design: Dependencies on open-source technologies (Python, Django, and scikit-learn) make transcending of proprietary resource requirements useless.

• Durable Design: A modular approach not only allows but also promotes long-term maintenance and extension.

• Minimal Environmental Impact: The use of cloud providers minimises the need for dedicated hardware.

• Consumer Safety & Transparency: The users are kept in the loop about the processing of their data which, in turn, increases trust and responsible usage. Broader Sustainability View

• Improves societal stability by preventing the dissemination of false information.

• No more irresponsible digital activities, and hence, it is social sustainability. • Technology growth that is low-waste and scalable is supported.

8.5 Safety Aspects

Safety in technology guarantees non-hazardousness of users or surrounding environment of the system. Safety Measures in the Project

• Secure Authentication: This keeps the admin dashboard safe from unauthorized access.

• Input Validation: It acts as a shield that protects the servers from attackers using malicious URLs or injections.

• Data Encryption: With HTTPS, the data transfer between the client and the server is secured.

• Error Handling: System crashes are prevented and accidental misuse is made difficult.

• Audit Logging: In case of safety investigation, the suspect activity has already been recorded and thus tracked.

Why Safety Matters Fake profile detection needs to frequently communicate with public digital data, this implies that:

• Adequate safety measures lead to less exposure to hacking attacks.

• It prevents manipulations, denial-of-service attempts or any other abuse.

• It ensures good behavior and reliability of the system.

Chapter 9

Conclusion

The present work describes the project as a good design and Django-based web application development with machine-learning integrated fake profile detection system. The system is a solution to the problem of fake accounts that is providing not only a reliable and scalable solution but also an efficient one in analyzing profile behavior, metadata, and text features.

Key Achievements

• A proper data preprocessing, feature extraction, and model training pipeline was built.

• High accuracy and recall rates were obtained through Random Forest and XGBoost applications.

• Instant prediction was made possible through a convenient Django interface.

• Admin can monitor and control suspicious accounts. •

Social, ethical, legal, and sustainability norms were followed.

Impact

The system will make digital space safer, protect online communication, and the overall battle against cybercrime because of the diminished risks that fake accounts create.

Future Scope

• The application of deep learning for higher-level pattern recognition.

• Social media API monitoring almost in real-time.

• AI-created profile pictures and deepfake identities identifying.

• Extension to the detection of fake profiles across multiple platforms.

• An automatic reporting system for large corporations that is entirely self-sufficient.

In conclusion, the project has built a strong foundation for future research and technological progress in the field of digital identity validation and automated fraud detection.

REFERENCES

[1] Y. Li, O. Martinez, Y. Li, J. E. Hopcroft, and X. Chen, “In a World That Counts: Clustering and Detecting Fake Social Engagement at Scale,” IEEE International Conference on Data Mining (ICDM), pp. 110–119, 2016.

[2] E. Van Der Walt and J. Neloff, “Using Machine Learning to Detect Fake Identities: Bots vs Humans,” IEEE International Conference on Advances in Computing and Communication Engineering (ICACCE), pp. 1–6, 2018.

[3] S. Khaled, H. Mokhtar, and N. El-Tazi, “Detecting Fake Accounts on Social Media,” IEEE International Conference on Computer Engineering and Systems (ICCES), pp. 176–183, 2018.

[4] A. Sarfraz, A. Ahmad, F. Zeshan, M. Hamid, and T. A. N. Alshalali, “Unmasking Deception: Detection of Fake Profiles in Online Social Ecosystems,” IEEE Access, vol. 10, pp. 10345–10359, 2022.

[5] A. K. M. R. R. Habib, E. E. Akpan, B. Ghosh, and I. K. Dutta, “Techniques to Detect Fake Profiles on Social Media Using New Age Algorithms – A Survey,” IEEE Access, vol. 10, pp. 205678–205695, 2022.

[6] M. B. Karamu and E. N. Araka, “A Hybrid Machine Learning Model for Detection of Fake Profile Accounts on Social Media Networks,” IEEE Access, vol. 10, pp. 109822–109834, 2022.

[7] S. Ahmad and M. M. Tripathi, “A Review Article on Detection of Fake Profile on Social-Media,” IEEE International Conference on Computing, Power and Communication Technologies (GUCON), pp. 1–6, 2023.

[8] D. R. Patil, T. M. Pattewar, V. D. Punjabi, and S. M. Pardeshi, “Detecting Fake Social Media Profiles Using the Majority Voting Approach,” IEEE Access, vol. 11, pp. 153220–153231, 2023.

[9] D. Amankeldin, L. Kurmangaziyeva, A. Mailybayeva, N. Glazyrina, A. Zhumadillayeva, and N. Karasheva, “Deep Neural Network for Detecting Fake Profiles in Social Networks,” IEEE Access, vol. 11, pp. 122320–122335, 2023.

[10] A. Agravat, U. Makwana, S. Mehta, D. Mondal, and S. Gawade, “Fake Social Media Profile Detection and Reporting Using Machine Learning,” IEEE International Conference on Communication and Electronics Systems (ICCES), pp. 1–6, 2024.

[11] A. Shah, S. Varshney, and A. Mehrotra, “Detection of Fake Profiles on Online Social Network Platforms: Performance Evaluation of AI Techniques,” IEEE Access, vol. 12, pp. 54567–54580, 2024.

BASE PAPER

Title: Unmasking Deception: Detection of Fake Profiles in Online Social Ecosystems
Authors: A. Sarfraz, A. Ahmad, F. Zeshan, M. Hamid, T. A. N. Alshalali
Published in: IEEE Access, 2022
Reason for Selection:

- Provides a comprehensive framework for identifying fake profiles using multi-feature analysis.
- Influenced the design of this project’s feature extraction and ML methodology.
- Demonstrates hybrid model effectiveness in large-scale social media environments.
- Serves as the foundational reference for dataset selection, model comparison, and evaluation metrics used in the project.

APPENDIX- A

PSEUDOCODE

Page 1: Signup and Login Flow

Pseudocode:

```
BEGIN
DISPLAY "Signup Page"
IF user selects "Create Account" THEN
    INPUT name, email, password
    VALIDATE inputs
    IF inputs are valid THEN
        HASH password
        STORE user credentials in database
        DISPLAY "Account Created Successfully"
    ELSE
        DISPLAY "Invalid Input. Please Try Again"
    ENDIF
ENDIF
DISPLAY "Login Page"

IF user enters email AND password THEN
    VALIDATE credentials
    IF credentials match database THEN
        REDIRECT to "Profile Data Entry Page"
    ELSE
        DISPLAY "Incorrect Login Details"
    ENDIF
ENDIF

END
```

```
Page 2: User Profile Data Entry
BEGIN

DISPLAY "Profile Data Entry Page"

INPUT:
    - Bio / Description
    - Followers count
    - Following count
    - Number of posts
    - Profile picture (optional)

STORE all inputs temporarily

ON submit:
    REDIRECT to "Data Preprocessing Page"

END

Page 3: Data Preprocessing
BEGIN

LOAD user profile data

// Text Preprocessing
bio_clean ← remove punctuation
bio_clean ← convert to lowercase
bio_clean ← remove stopwords
bio_clean ← remove URLs and special characters

// Numeric Normalization
```

```
followers_norm ← normalize follower count
following_norm ← normalize following count
posts_norm ← normalize post count

// Image verification (optional)
IF profile picture exists THEN
    CHECK for fake/AI-generated image patterns
ENDIF

STORE cleaned data

REDIRECT to "Feature Extraction Page"

END

Page 4: Feature Extraction
BEGIN

// Behavioral Features
ratio ← followers / following
posting_rate ← calculate posts per day
engagement_anomaly ← detect irregular activity

// NLP Features
sentiment_score ← sentiment(bio_clean)
text_complexity ← readability score

// Metadata Features
profile_complete ← count filled fields (bio, picture, posts)

// Create final vector
feature_vector ← concatenate(
    ratio,
    posting_rate,
    sentiment_score,
    text_complexity,
    profile_complete
)

STORE feature_vector

REDIRECT to "Model Prediction Page"

END

Page 5: Model Prediction and Pie Chart
BEGIN

LOAD:
    model_RF   ← Random Forest
    model_XGB  ← XGBoost
    model_ANN  ← Neural Network (optional)

// Predictions
pred_RF   ← model_RF.predict(feature_vector)
pred_XGB  ← model_XGB.predict(feature_vector)
pred_ANN  ← model_ANN.predict(feature_vector)

fake_probability ← average(pred_RF, pred_XGB, pred_ANN)

// Decision
IF fake_probability ≥ threshold THEN
    LABEL ← "Fake Profile Detected"
ELSE
```

```
    LABEL ← "Real Profile"
ENDIF

DISPLAY pie chart (Fake vs Real probability)

IF LABEL = "Fake Profile Detected" THEN
    DISPLAY "Report Profile" button
ENDIF

END

Page 6: Reporting System
BEGIN

IF user clicks "Report Profile" THEN
    SAVE report in database
    SET status = "Pending Review"
    ALERT Admin Dashboard
ENDIF

DISPLAY "Profile Reported Successfully"

END

Page 7: Admin Dashboard
BEGIN

DISPLAY "Admin Login Page"
VALIDATE admin credentials

IF valid THEN
    LOAD:
        total_profiles
        pending_reports
        resolved_reports
        confirmed_fake_profiles
        model_evaluation_metrics

    DISPLAY dashboard summary and tables
ELSE
    DISPLAY "Unauthorized Access"
ENDIF

END

Page 8: Evaluation and Feedback
BEGIN

LOAD test dataset

accuracy  ← calculate accuracy
precision ← calculate precision
recall    ← calculate recall
f1_score  ← calculate F1-score

DISPLAY evaluation metrics

DISPLAY feedback form
STORE feedback

END

Page 9: Future Enhancements
BEGIN
```

```
DISPLAY enhancements:

1. Live social media API integration
2. Automated real-time monitoring
3. Blockchain logging of reported profiles

END

Page 10: Security & Summary
BEGIN

// Security implementations
ENABLE HTTPS
ENCRYPT sensitive data
APPLY rate limiting
USE JWT / OAuth2 authentication

// Summary
DISPLAY:
    - Project purpose
    - Impact
    - Detection benefits
    - Deployment recommendations

END
```

APPENDIX-B

SCREENSHOTS

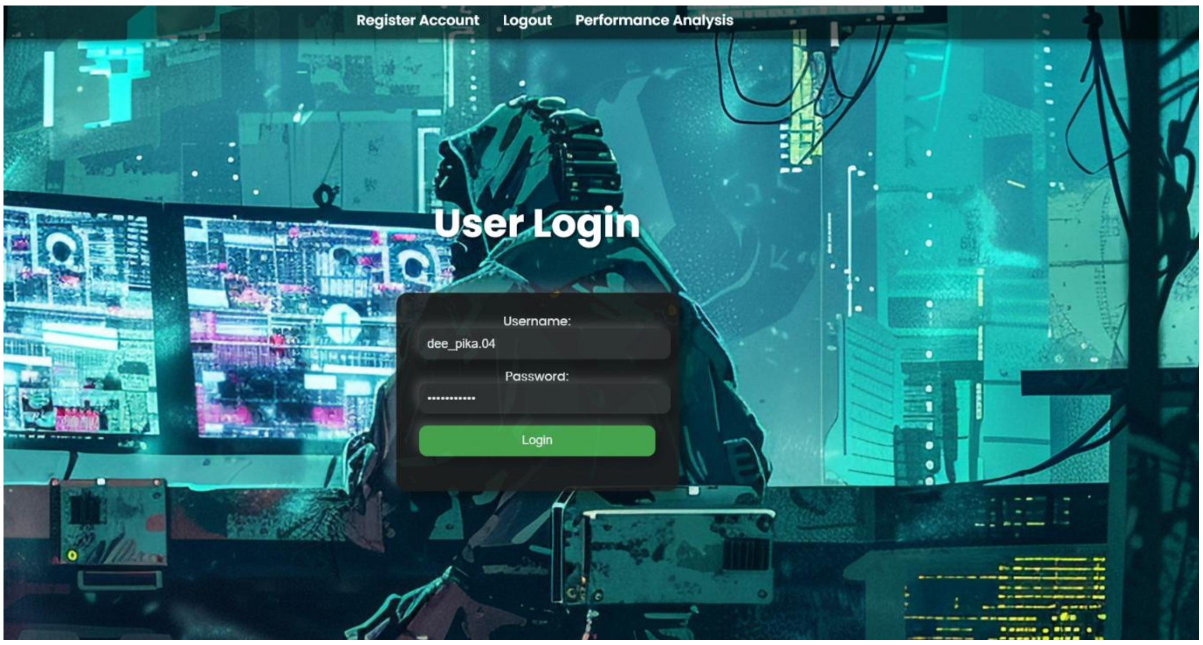


Fig 1.1 User Login Page

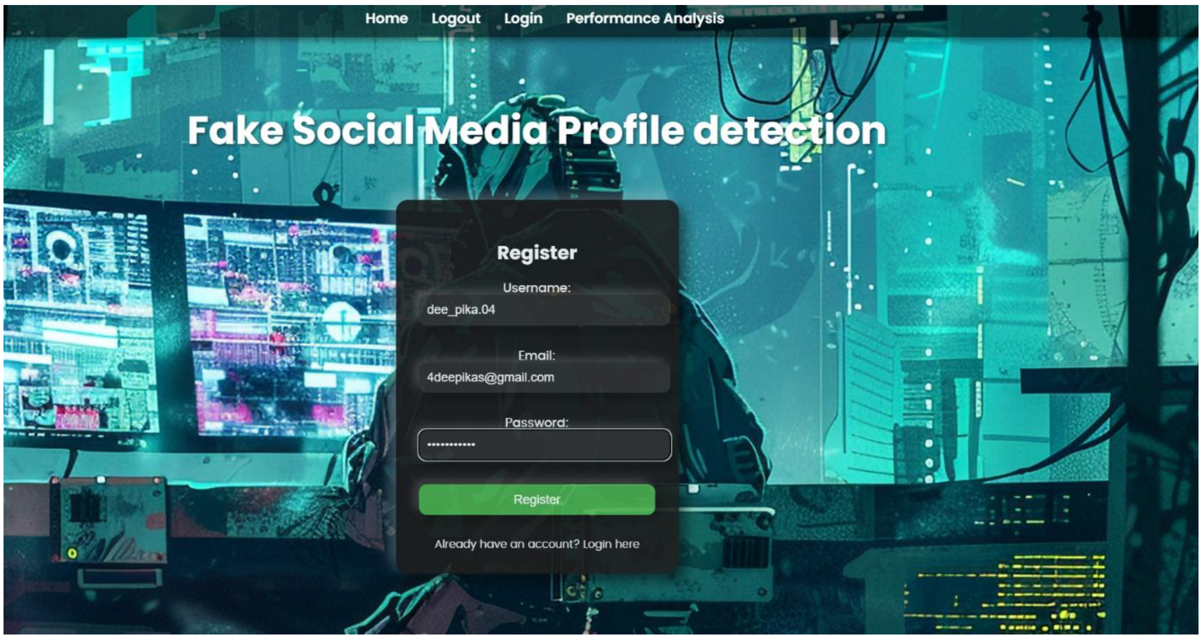
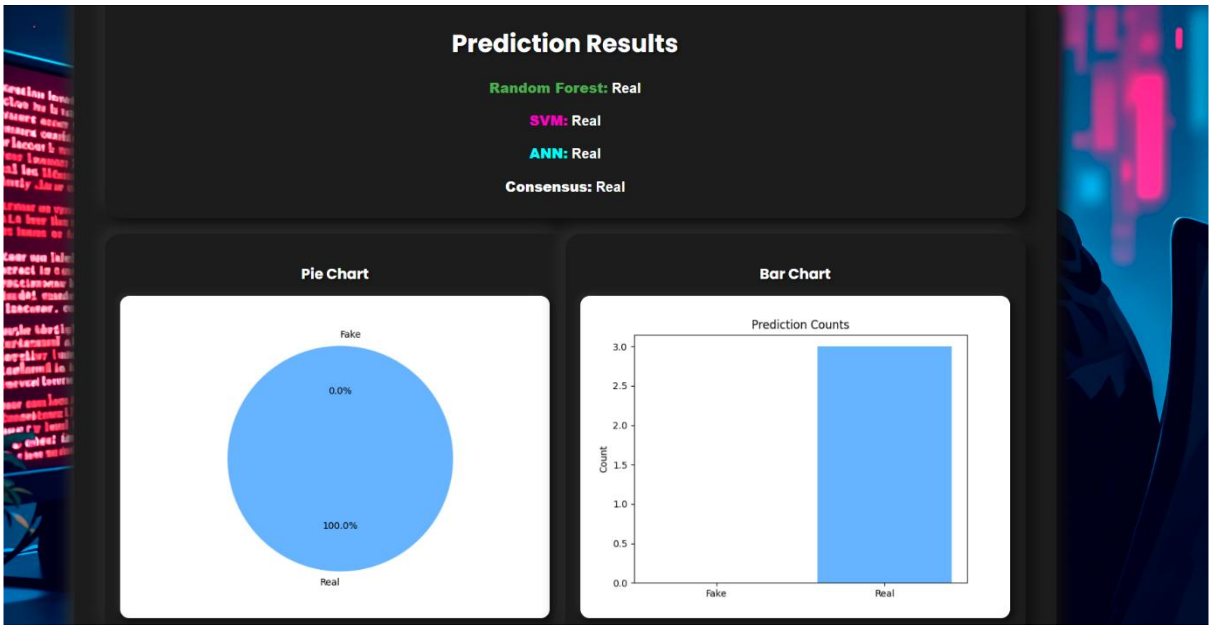


Fig 1.2 Register Page



Prediction Results

Random Forest: Real

SVM: Real

ANN: Real

Consensus: Real

Fig 1.3 Result Prediction