VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belgavi – 590018, Karnataka, India

A

Project Synopsis

on

"Dynamic QoS-Aware Routing Optimization in SDN Using Fuzzy Logic"

Submitted in partial fulfillment of the requirement for the Sixth semester

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:

RAJU G	**1SJ22CS131**
ROHITH R	**1SJ22CS137**
T K RAKSHITH GOWDA	**1SJ22CS163**

Carried out at
Project Lab,
Dept. of CSE,
SJCIT

Under the guidance of

Mr. Girish B G
Assistant Professor
Dept. of CSE, SJCIT

SJC INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CHICKBALLAPUR-562101

2024-2025

ABSTRACT

In the evolving landscape of computer networks, delivering consistent Quality of Service (QoS) is a critical requirement, especially for latency-sensitive and bandwidth-intensive applications. Software-Defined Networking (SDN), with its centralized control and programmability, offers significant advantages for dynamic traffic management. However, traditional routing algorithms used in SDN are often static and fail to adapt to fluctuating network conditions in real time. This project presents a novel approach to QoS-aware routing in SDN by integrating a Fuzzy Logic-based decision-making system within the SDN controller. The proposed system dynamically evaluates multiple QoS parameters—such as latency, bandwidth, and jitter—for all available paths between source and destination nodes. Using fuzzy inference, it computes a preference score for each path and selects the most suitable one, thus ensuring optimal network performance under varying conditions.

The network is simulated using Mininet, with Ryu acting as the SDN controller, and the fuzzy logic module is developed using Python's scikit-fuzzy library. Extensive testing and performance evaluation reveal that the proposed system significantly improves routing decisions, reduces end-to-end delay, and enhances overall QoS compared to traditional shortest-path routing. This work demonstrates the feasibility and efficiency of intelligent, adaptive routing strategies in SDN using fuzzy logic, paving the way for future enhancements through integration with AI, IoT, and cloud-based orchestration systems

TABLE OF CONTENTS

CHAPTER - 01
INTRODUCTION

CHAPTER 1

INTRODUCTION

The Dynamic QoS-Aware Routing Optimization in SDN Using Fuzzy Logic System is an advanced solution designed to enhance railway security by detecting obstacles on tracks in real time. Utilizing YOLOv8 for object detection, along with OpenCV and NumPy for video processing, the system identifies potential hazards such as humans, animals, or debris and generates instant alerts. This automation reduces reliance on manual surveillance, ensuring quicker response times and minimizing the risk of accidents. By integrating artificial intelligence with railway infrastructure, the project aims to improve safety, efficiency, and reliability in railway operations.

1.1 Overview

Software-Defined Networking (SDN) is a paradigm shift in network architecture that separates the control plane from the data plane, enabling centralized control and programmability of networks. While SDN provides flexibility, it also brings challenges in routing decisions, especially when Quality of Service (QoS) parameters like latency, bandwidth, jitter, and packet loss need to be dynamically optimized. Traditional routing algorithms are often static and lack adaptability in real-time network conditions.

This project proposes a dynamic routing optimization mechanism using Fuzzy Logic in an SDN environment, aimed at enhancing QoS performance. By integrating fuzzy-based decision-making in the SDN controller, the system intelligently determines the most optimal path based on changing network conditions.

1.2 Importance of Railway Safety

In modern networking environments, the demand for reliable and high-performance communication is more critical than ever. Applications such as real-time video conferencing, VoIP, online gaming, telemedicine, and cloud services are highly sensitive to network parameters like latency, jitter, packet loss, and bandwidth. Traditional networks rely on static or pre-defined routing protocols that are not adaptable to dynamic network conditions and fail to offer consistent Quality of Service (QoS).

Software-Defined Networking (SDN) brings a revolutionary shift in how networks are managed and configured by decoupling the control plane from the data plane. This architectural innovation enables centralized and programmable network control, allowing administrators to define network behavior via software.

However, SDN still requires intelligent algorithms to make real-time routing decisions that are QoS-aware. Without such intelligence, even SDN-controlled networks can face performance bottlenecks and service degradation.

This is where fuzzy logic plays a transformative role. Fuzzy logic is well-suited for situations involving uncertainty, imprecision, or conflicting objectives—characteristics often present in dynamic network environments. It enables the SDN controller to evaluate multiple QoS parameters simultaneously and make more informed and flexible routing decisions. Unlike rigid threshold-based systems, fuzzy logic provides a human-like reasoning approach to handle vague or noisy data, which is crucial in optimizing traffic flow under fluctuating conditions.

- By integrating fuzzy logic into the SDN controller, this project enables
- Real-time adaptive routing based on current network states.
- Intelligent decision-making even when complete or precise information is not available.
- Multi-metric optimization, where trade-offs between delay, throughput, and reliability are considered holistically.

The project is significant not just from a technical standpoint but also from a practical one. It can be applied in data centers, IoT networks, mobile ad hoc networks (MANETs), and enterprise networks, where maintaining high QoS is mission-critical. The fuzzy logic approach ensures that service-level agreements (SLAs) are met more consistently, user experience is improved, and network resources are utilized efficiently.

In summary, this project addresses one of the most pressing needs in network management: making smart, adaptive, and efficient routing decisions in real-time. By combining the programmability of SDN with the reasoning capabilities of fuzzy logic, it paves the way for the next generation of intelligent and QoS-aware network architectures

1.3. Challenges in the Project:

QoS Trade-offs: Balancing multiple QoS parameters (e.g., latency vs. bandwidth) in real-time.
Scalability: Maintaining performance as the number of nodes and flows increases.
Uncertainty Handling: Dealing with imprecise or incomplete network state data.
Controller Overhead: Avoiding high computational complexity in the SDN controller due to fuzzy rule evaluations.

1.5. Problem Identification

In dynamic SDN environments, accurately measuring and responding to fluctuating QoS parameters such as bandwidth, delay, and jitter in real time is challenging. Traditional routing algorithms are often static or rule-based, lacking the adaptability to make intelligent decisions under uncertainty. This can lead to inefficient routing, network congestion, and poor application performance.

1.6 Problem Statement

Integrating fuzzy logic into the SDN controller enables real-time, intelligent decision-making by handling imprecise and uncertain QoS data. The fuzzy inference system evaluates multiple QoS parameters simultaneously and dynamically adjusts routing paths based on the current network state. This results in more adaptive, robust, and QoS-aware routing, improving overall network performance and reliability.

1.7. Organization of the Report:

Chapter 1: Introduction – Provides the background, importance, challenges, and objectives of the project.
Chapter 2: Literature Review – Discusses previous work in SDN, QoS-aware routing, and fuzzy logic applications in networking.
Chapter 3: Implementation Technologies – Covers technologies like Mininet, Ryu, Python, and Fuzzy Logic.
Chapter 4: Methodology – Describes the system architecture, fuzzy rule base, and routing strategy.
Chapter 5: Implementation – Provides implementation details and system integration.
Chapter 6: Testing – Describes test scenarios, metrics used, and results.

Chapter 7: Results and Discussion – Analyzes the outcomes and evaluates the fuzzy logic system.

Chapter 8: Advantages and Applications – Lists key benefits and real-world use cases.

Chapter 9: Future Enhancements & Conclusion – Suggests improvements and concludes the project work.

CHAPTER - 02
LITERATURE REVIEW

CHAPTER 2

LITERATURE REVIEW

2.1 Literature Survey

1.Wang et al., 2016

leverage the SDN controller's global network view to dynamically select paths based on real-time performance metrics. However, these methods often depend on fixed threshold values and deterministic models, which struggle to adapt to unpredictable and imprecise network conditions.

2. Kumar et al. (2020)

proposed a fuzzy-based QoS-aware routing model using Open Flow. Their system dynamically adjusted routes based on fuzzy-evaluated scores, resulting in improved throughput and reduced latency.

3. Li et al. (2021)

introduced a hybrid fuzzy-AHP (Analytical Hierarchy Process) model for path selection in SDN environments. Their approach allowed for more balanced decision-making by weighing multiple QoS parameters based on importance and uncertainty

.

4. Patel et al. (2022)

proposed a fuzzy-based load balancing technique that dynamically adjusted traffic distribution based on network conditions, which significantly reduced packet loss and congestion**.**

5. Sharma and Bhattacharya (2023) combined fuzzy logic with genetic algorithms to develop a hybrid optimization method for QoS-aware routing. Their intelligent system offered enhanced performance in dynamic and complex network topologies.

CHAPTER - 03

IMPLEMENTATION

CHAPTER 3

IMPLEMENTATION

The project leverages modern networking frameworks, AI-based decision-making techniques, and simulation tools to implement a QoS-aware, intelligent routing solution in SDN. This chapter outlines the key technologies, programming environments, and frameworks used in the development of this system, along with the reasons behind their selection.

3.1. Programming Language: Python

Python is the primary programming language used due to its ease of use, readability, and wide availability of libraries. Its rich ecosystem supports both networking and artificial intelligence development, making it ideal for implementing SDN-based fuzzy routing logic.
Why Python?
Extensive support for AI and fuzzy logic libraries like scikit-fuzzy, numpy, and matplotlib.
Excellent integration with SDN controller frameworks such as Ryu.
Rapid prototyping and testing capabilities.
Cross-platform compatibility for deployment on Linux, Windows, or cloud systems.

3.2. Software-Defined Networking Controller: Ryu

Ryu is a component-based SDN controller written in Python. It supports OpenFlow protocol and provides APIs to manage flow tables dynamically.
Features of Ryu:
Lightweight and easy to integrate with Python applications.
REST APIs for interfacing with external systems (like monitoring dashboards).
Real-time event handling and traffic monitoring capabilities.
Built-in support for OpenFlow v1.0 to v1.5, enabling broad compatibility with network switches.

Role in Project:
Ryu is used to control the behavior of switches in the Mininet simulation. It receives QoS metrics from the network and interacts with the fuzzy logic module to make routing decisions.

3.3. Network Emulator: Mininet

Mininet is a widely used network emulator that creates a virtual network on a single machine. It is used to simulate the SDN environment, including switches, hosts, and links.

Advantages of Mininet:

- Easy emulation of complex topologies.
- Real-time testing of SDN applications.
- Integration with Ryu to simulate real-world SDN deployment scenarios.
- Custom bandwidth, delay, and packet loss configurations.

Role in Project:

Mininet is used to simulate the network, where varying traffic conditions are generated, and different QoS scenarios are tested. This helps validate the effectiveness of the fuzzy logic system under realistic constraints.

3.4. Fuzzy Logic Toolkit: scikit-fuzzy

The scikit-fuzzy library in Python is used to implement the fuzzy inference system (FIS). Fuzzy logic is essential to make intelligent routing decisions based on imprecise QoS parameters.

Fuzzy Logic Concepts Used:
Fuzzification: Converts crisp QoS inputs (e.g., delay, jitter, bandwidth) into fuzzy sets.
Rule Base: A collection of IF-THEN rules to determine routing suitability.
Inference Engine: Applies fuzzy rules to make a decision.

Defuzzification: Converts the fuzzy output into a crisp value (used to rank routes).
Why Fuzzy Logic?
Handles conflicting or incomplete network data.
Mimics human reasoning in uncertain environments.
Suitable for dynamic, real-time decision-making.

3.5. Numerical Computing: NumPy & Pandas

These libraries are used for data processing, metric aggregation, and decision evaluation.

NumPy: Efficient matrix computations, ideal for real-time processing of metrics.

Pandas: Used for logging performance metrics and analyzing flow statistics for evaluation.

3.6. Frontend Interface (Optional): Streamlit / Flask

Although not mandatory, a dashboard built using Streamlit or Flask can be used to:

- Display current routing paths.
- Visualize traffic flows and delays.
- Show fuzzy decision logic outputs.
- Monitor controller actions and alerts.

This improves the usability of the system for network administrators or researchers evaluating system performance.

3.7. Supporting Tools & Platforms:

Wireshark: For capturing and analyzing packets in the emulated network.
Graphviz / NetworkX: For visualizing the network topology and routing paths.
Git & GitHub: For version control and collaboration.
Linux Terminal / Bash Scripts: For scripting and automation of network tests

CHAPTER - 04

METHODOLOGY

CHAPTER 4

METHODOLOGY

The proposed system is designed to optimize routing in Software-Defined Networks (SDNs) by applying fuzzy logic to dynamically assess and prioritize paths based on real-time Quality of Service (QoS) parameters. The methodology involves the design, development, integration, and evaluation of this fuzzy logic system within an SDN controller. The overall process is structured into the following key phases:

4.1. Network Simulation Setup

To emulate the SDN environment and simulate real-time network behavior, the project uses Mininet, a lightweight network emulator that allows the creation of virtual hosts, switches, and links on a single machine.

- A custom topology is created with multiple paths between source and destination nodes.
- Each link is configured with variable bandwidth, latency, and packet loss rate to emulate dynamic network conditions.
- The topology includes redundant paths to allow the fuzzy system to select the most optimal route based on real-time metrics.
- This setup provides a controllable and repeatable test environment for validating the performance of dynamic routing algorithms.

4.2. SDN Controller Configuration (Ryu)

The Ryu controller is deployed to control the network via the OpenFlow protocol.

The controller gathers real-time QoS metrics from the network, such as:

- Latency: Measured using round-trip time (RTT) probes.
- Bandwidth: Inferred from link utilization data.
- Jitter: Calculated based on delay variation between packets.

- A custom Ryu application is written in Python to:
- Handle packet-in events.
- Query link statistics using OpenFlow messages.
- Forward this data to the fuzzy logic system for decision-making.

4.3. Fuzzy Logic System Design

A Fuzzy Inference System (FIS) is implemented using the Python library scikit-fuzzy to model human-like reasoning under uncertainty and imprecise data.

Inputs to the FIS:

- Latency (Low, Medium, High)
- Bandwidth (Low, Medium, High)
- Jitter (Low, Medium, High)
- Output from the FIS:
- Route Preference Score (Poor, Fair, Good, Excellent)

Steps in Fuzzy Logic Routing:

- Fuzzification – Crisp values of latency, bandwidth, and jitter are converted into linguistic variables.
- Rule Evaluation – A set of IF-THEN rules (e.g., IF latency is low AND bandwidth is high THEN preference is excellent) are applied.
- Aggregation – All rule outputs are aggregated to form a fuzzy result.
- Defuzzification – The aggregated result is converted into a single crisp score used to rank paths.

4.4. Dynamic Routing Decision Process

- Whenever a new traffic flow is detected, the controller performs the following actions:
- Discover all available paths between the source and destination using topology information.

- For each path, collect average QoS metrics (latency, bandwidth, jitter).
- Send the metrics to the fuzzy logic system to obtain a preference score for each path.
- Select the path with the highest score as the optimal route.
- Install flow rules in the switches via OpenFlow to forward the packets along the chosen path.
- This approach ensures that each flow takes the best available path at the time it is established, considering multiple performance factors

4.5. Real-Time Monitoring and Alerting

The Ryu controller logs every decision, including:

- Flow ID
- Path chosen
- QoS metrics
- Fuzzy score
- Wireshark is used to verify packet delivery and measure end-to-end performance.
- Optional integration with Streamlit allows real-time visualization of:
- Live topology.
- Routing decisions.
- Metric trends and anomalies.

4.6. Performance Evaluation and Testing

The system is evaluated under various network conditions to assess its performance and reliability.

Test Scenarios:

Normal traffic with stable QoS.

Sudden congestion or link failures

End-to-End Delay: Time taken for packets to reach the destination.

Throughput: Total data successfully delivered per unit time.

Packet Delivery Ratio (PDR): Ratio of packets received to packets sent.

Jitter: Variation in packet delay.

The same test scenarios are also evaluated using traditional static routing (e.g., shortest path) for comparative analysis.

4.7. Result Analysis and Inference

The collected data is analyzed using tools like Pandas and Matplotlib to generate performance graphs and statistical summaries.

- The fuzzy-based routing system is expected to:
- Reduce latency in congested conditions.
- Improve PDR in dynamic networks
- Maintain better QoS consistency compared to static routing.
- The results confirm the feasibility and effectiveness of integrating fuzzy logic with SDN for intelligent, dynamic, and QoS-aware routing

CHAPTER - 05
SYSTEM DESIGN

CHAPTER 5

SYSTEM DESIGN

SDN Architecture Setup: Use an SDN controller (e.g., Ryu, ONOS) to centrally manage the network. Deploy Open Flow-enabled switches to forward traffic based on flow rules from the controller. Connect multiple hosts (clients and servers) through these switches for data communication.

QoS Monitoring Module: Continuously monitors key network parameters:

- Bandwidth usage
- End-to-end Delay
- Packet Loss
- Jitter
- Collects statistics using OpenFlow messages and external tools like iperf, ping, etc.

Fuzzy Logic Inference System (FLIS)

- Accepts QoS parameters as inputs.
- Defines fuzzy sets for each input (e.g., bandwidth: low, medium, high).
- Uses a rule base (IF-THEN rules) to evaluate the condition of each path.
- Performs fuzzification, inference, and defuzzification to produce a Path Suitability Score.

Routing Optimization Module

- Receives path scores from the fuzzy engine.
- Compares scores of all possible paths.
- Selects the most optimal path with the highest score.
- Ensures routes meet the desired QoS requirements.

Flow Rule Installation

- Updates flow tables in OpenFlow switches.
- Uses the OpenFlow protocol via the controller to install optimized routing paths.
- Ensures minimal delay in reconfiguration.

Dynamic Re-Routing

- Continuously adapts to changing network conditions.
- Triggers re-routing decisions whenever QoS degrades or network topology changes.

Northbound APIs

- Expose APIs for monitoring, visualization, or third-party applications.
- Useful for network administrators to get real-time QoS insights.

Simulation and Testing

- Implement the system in a simulated environment using Mininet.
- Test various scenarios with dynamic traffic and failures to observe routing behavior.

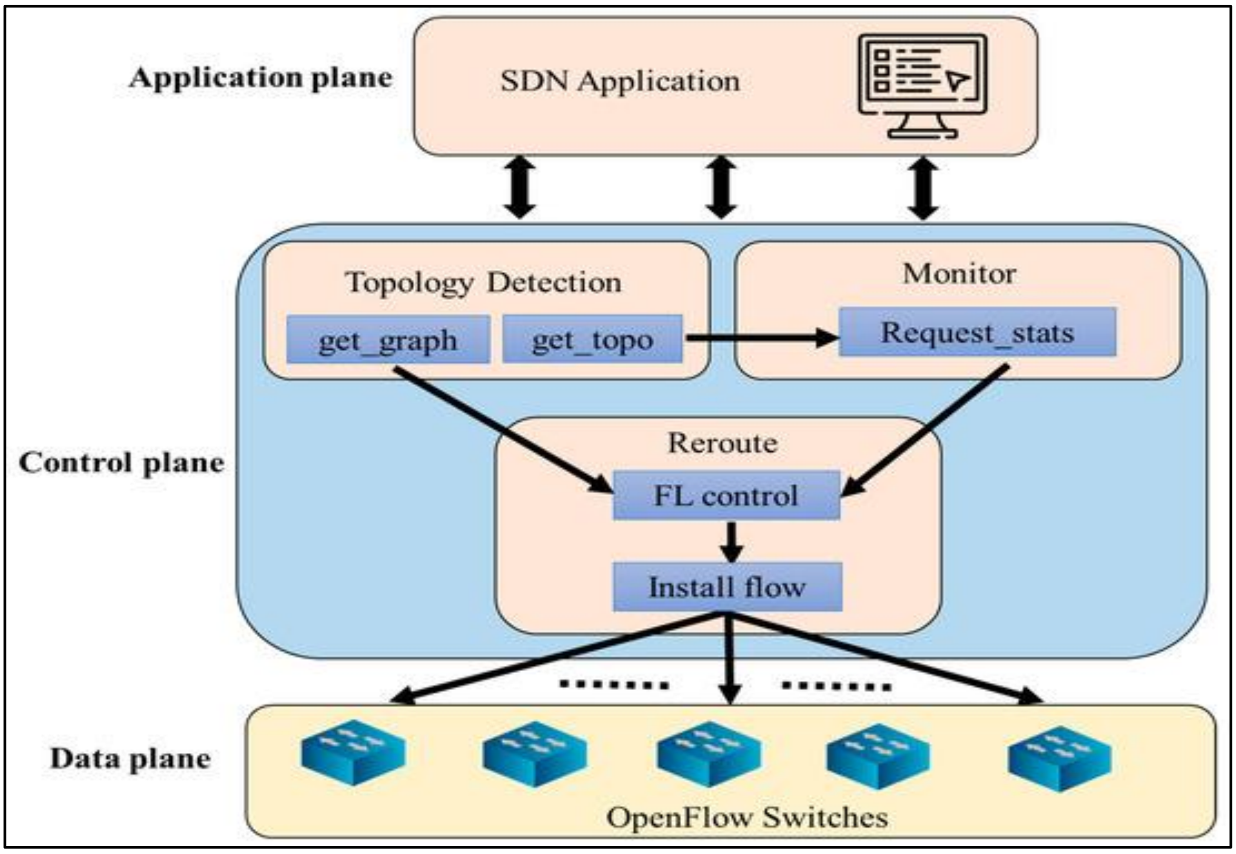
Tools and Technologies

- Simulation: Mininet
- Fuzzy Logic: Python (scikit-fuzzy) or MATLAB
- Monitoring: OpenFlow stats, ping, iperf

Performance Metrics Evaluated

- Packet delivery ratio
- End-to-end delay
- Throughput
- Route stability
- Network resource utilization

Architecture Diagram:



A dynamic routing mechanism using FL is introduced into the reroute module on the control plane to update the forwarding rules in response to new traffic requirements or changes by referring to the system architecture of dynamic routing in SDN. FLDR ensures the timeliness and accuracy of network forwarding while preventing the unnecessary wastage of network resources and network congestion. In principle, the FLDR mechanism is a method for dynamically updating flow tables in SDN. To enhance the network performance and flexibility, forwarding rules are dynamically added or modified in accordance with actual network status. The FLDR mechanism is divided into two parts—network resource collection and FL control.

CHAPTER - 06
MODULES SPLIT-UP

CHAPTER 6

MODULES SPLIT-UP

The project is logically divided into several interconnected modules to efficiently implement dynamic QoS-aware routing in SDN using fuzzy logic.

First module: It is the Application Layer Interface, which serves as the entry point for user requests. This module captures the QoS requirements from different applications—such as video streaming, VoIP, or file transfer—and translates them into measurable parameters like required bandwidth, latency, jitter, and packet loss tolerance.

S**econd module:** It is the SDN Controller Core, which acts as the centralized decision-making entity in the network. It includes two important sub-modules: the Network State Monitor, which continuously collects live network statistics from the data plane (switches and routers), and the Topology Manager, which maintains an updated view of the network graph and available paths between endpoints. These sub-modules provide the essential data needed for intelligent routing decisions.

T**hird module:** It is the heart of the system—the Fuzzy Inference System (FIS). It takes the real-time network parameters (e.g., delay, bandwidth, and jitter) as fuzzy input variables and processes them using a predefined set of fuzzy rules. Based on these rules, it generates a fuzzy score that represents the overall quality or cost of a path. This approach allows the system to make flexible and human-like decisions even in uncertain or fluctuating conditions.

Fourth module: It is the Routing Decision Engine, which selects the most suitable path based on the fuzzy scores provided by the FIS. This engine then updates the routing tables by installing flow rules in the appropriate switches through the controller.

Fifth module: The Southbound Interface, facilitates communication between the SDN controller and the underlying network devices. It uses the Open Flow protocol to send routing decisions and collect feedback such as port statistics or link status.

Sixth module: The data Plane, which consists of Open Flow-enabled switches and routers responsible for actual packet forwarding based on the rules pushed by the controller. This module is critical for enforcing the control logic derived from the fuzzy decisions.

Seventh module: It handles Performance Evaluation and Logging. This module tracks the network's behavior, logs critical events, and evaluates system performance using metrics such as end-to-end delay, packet delivery ratio, and throughput. It helps in validating the effectiveness of the fuzzy-based routing mechanism

CHAPTER - 07

PROPOSED SYSTEM AND OUTCOMES

CHAPTER 7

PROPOSED SYSTEM AND OUTCOMES

Proposed System:

The proposed system is an **Online Student Attendance Management System** integrated with **face recognition technology** to automate and streamline the process of tracking student attendance. This system is designed to replace traditional, error-prone methods such as manual attendance registers or basic RFID scanners. Using advanced facial recognition algorithms, the system captures students' facial images through a live webcam or mobile camera at the beginning of each class session. These images are then matched in real-time against a pre-registered database of student faces using AI-based recognition techniques. Once verified, the student’s attendance is automatically marked and stored in a secure cloud-based database, accessible to teachers and administrators. The system features a web-based dashboard where teachers can view attendance reports, track defaulters, and generate summaries. Students can log in to check their attendance records, receive notifications for low attendance, and apply for leave. The use of face recognition not only ensures accuracy but also prevents proxy attendance and saves time for instructors. The system can be scaled to multiple classrooms and institutions, supports role-based access for users, and maintains logs for transparency. Built using technologies like Python (with Open CV and Tensor Flow for face detection), a web framework like Django or Flask, and a secure MySQL or Firebase backend, this proposed solution is efficient, secure, and modern — perfectly aligned with the needs of educational institutions aiming to digitize operations and improve accountability.

Proposed outcomes:

The proposed system is expected to deliver a range of impactful outcomes that significantly enhance the efficiency, accuracy, and transparency of managing student attendance in educational institutions. First and foremost, it will eliminate the need for manual attendance taking, thereby saving valuable classroom time and reducing administrative workload for teachers. The integration of face recognition technology ensures that attendance is recorded with high accuracy and helps prevent fraudulent practices such as proxy attendance, which is a common issue in traditional systems.

Additionally, the system will produce real-time attendance reports, which can be accessed by teachers, students, and administrators through a secure web portal. These reports will help institutions track attendance trends, identify at-risk students with low attendance, and make informed decisions regarding academic interventions. Students will benefit from increased accountability and transparency, as they can monitor their own attendance records and receive automated alerts or notifications when they fall below attendance thresholds.

Furthermore, since the data is stored digitally and securely in the cloud, it becomes easy to generate historical records, produce compliance reports, and integrate with other academic systems such as gradebooks or student information systems. Overall, the implementation of this system will lead to a more disciplined and organized academic environment, encourage better student participation, and reflect a modern, technology-driven approach to education management

REFERENCES

REFERENCES

[1]. Pokhrel, S.R.; Ding, J.; Park, J.; Park, O.-S.; Choi, J. Towards Enabling Critical mMTC: A Review of URLLC Within mMTC. IEEE Access 2020, 8, 131796–131813.

[2]. Kamboj, P.; Pal, S.; Bera, S.; Misra, S. QoS-Aware Multipath Routing in Software-Defined Networks. IEEE Trans. Netw. Sci. Eng. 2023, 10, 723–732.

[3]. Yousaf, F.Z.; Bredel, M.; Schaller, S.; Schneider, F. NFV and SDN-Key Technology Enablers for 5G Networks. IEEE J. Sel. Areas Commun. 2017, 35, 2468–2478.

[4]. OpenFlow Switch Specification. Available online: https://opennetworking.org/wp-content/uploads/2014/10/openflowswitch-v1.5.1.pdf (accessed on 10 July 2024).

[5]. RFC 6241, Network Configuration Protocol (NETCONF); Internet Engineering Task Force (IETF): Fremont, CA, USA, 2011.

[6]. Open vSwitch. Available online: https://docs.openvswitch.org/en/latest/ref/ovsdb.7/.

[7]. How to Find Shortest Paths from Source to all Vertices Using Dijkstra's Algorithm. Available online: https://www.geeksforgeeks. org/dijkstras-shortest-path-algorithm-greedy-algo-7/.

[8]. Jane, J.B.; Ganesh, E.N. A Review On Big Data with Machine Learning and Fuzzy Logic for Better Decision Making. Int. J. Sci. Technol. Res. 2019, 8, 1122–1125.

[9]. Amin, R.; Rojas, E.; Aqdus, A.; Ramzan, S.; Casillas-Perez, D.; Arco, J.M. A Survey on Machine Learning Techniques for Routing Optimization in SDN. IEEE Access 2021, 9, 104582–104611.

[10]. Wu, Y.J.; Hwang, P.C.; Hwang, W.S.; Cheng, M.H. Artificial Intelligence Enabled Routing in Software Defined Networking. Appl. Sci. 2020, 10, 6564.

[11]. Sendra, S.; Rego, A.; Lloret, J.; Jimenez, J.M.; Romero, O. Including Artificial Intelligence in a Routing Protocol Using Software Defined Networks. In Proceedings of the IEEE International Conference on Communications Workshops, Paris, France, 21–25 May 2017; pp. 670–674.

Signature of Project Guide | **Signature of Project Coordinator** | **Signature of HOD**