NM1051- SERVICE NOW ADMINISTRATOR

STREAMLINING TICKET ASSIGNMENT FOR EFFICIENT SUPPORT OPERATION

A Project Report Submitted by

ABINAYA S – 962722104002

MARI@MALINI S – 962722104027

MAHALAKSHMI S – 962722104026

ASIN BELLA M – 962722104011

BACHELOR OF ENGINEERING

COMPUTER SCIENCE & ENGINEERING

UNIVERSAL COLLEGE OF ENGINEERING & TECHNOLOGY

VALLIOOR – 627117

ANNA UNIVERSITY: CHENNAI – 600025 NOV/DEC 2025

BONAFIDE CERTIFICATE

Certified that this Naan Mudhalvan project report " SERVICE NOW ADMINISTRATOR " **Abinaya S(962722104002), Mari@malini S(962722104027),Mahalakshmi S(962722104026),Asin Bella M(962722104011)** who carried out the project at "**STREAMLINING TICKET ASSIGNMENT FOR EFFICIENT SUPPORT OPERATION ".**

SIGNATURE. **SIGNATURE.**

Prof.CHANDRA LEKA ..,M.E
STAFF INCHARGE
Dept. of Computer Science & Engg
Universal College of Engg & Tech
Vallioor-627117

Prof.M.PRADEESHKUMAR..,ME.,
HEAD OF THE DEPARTMENT
Dept. of Computer Science & Engg
Universal College of Engg & Tech
Vallioor -627117

Submitted for the Anna University Examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We sincerely thank to Tamil Nadu Skill Development Corporation

(TNSDC), “Naan Mudhalvan” Platform and for encouragement towards our projectwork for providing necessary skill training.

We sincerely thank our principal **Dr.T. ASEER BRABIN, ME., MISTE., PHD.,**

for encouragement towards our project works.

We also thank our He of the Department and our project guide and our parents forthe complete and wholehearted support, motivation guidance and help in making our project activities

Table of contents :

1.Abstract

In customer support systems, assigning tickets to the right support agents is a time-consuming process. This project aims to automate ticket assignment using a rule-based or AI-driven algorithm to ensure that tickets are distributed efficiently according to agent availability, skill level, and workload. The solution reduces response time, improves customer satisfaction, and optimizes team performance.

In modern organizations, customer and technical support operations play a vital role in maintaining client satisfaction and ensuring smooth business processes. With the increasing number of customer requests and issues reported daily, managing support tickets efficiently has become a challenging task for IT helpdesks and service teams. Traditionally, ticket assignment is handled manually by supervisors or support managers who review each ticket and assign it to a suitable agent based on their knowledge and workload. However, this manual process is often time-consuming, prone to human errors, and can lead to uneven distribution of work among agents. Such inefficiencies ultimately delay response times and reduce customer satisfaction.

To overcome these challenges, the Streamlining Ticket Assignment for Efficient Support Operation project proposes an automated and intelligent ticket assignment system. The main goal of this project is to simplify and speed up the process of assigning support tickets to agents by using a Python-based automation model. The system is designed to automatically allocate each new ticket to the most appropriate agent by considering multiple factors such as ticket category, agent specialization, priority level, and current workload. This intelligent allocation ensures that every ticket is handled by the right person, thereby improving the efficiency of the entire support process.

The proposed system uses a structured methodology. When a new ticket is created, the system first classifies it based on the problem type — for instance, software, network, or hardware issue. It then refers to an agent database where each agent's skill set and current ticket count are stored. Using a predefined algorithm, the system matches the ticket category with agents who have the required skill and selects the one with the fewest active tickets. This automated logic ensures fair distribution of workload among agents and minimizes idle time. Once the agent is assigned, a notification is generated, and the system updates the database accordingly.

This approach not only increases productivity but also reduces dependency on manual monitoring. It ensures faster issue resolution, improved accuracy in ticket routing, and balanced agent workloads. Moreover, the system can be easily extended to incorporate machine learning algorithms in the future, allowing it to learn from past assignments and make predictive decisions for future tickets. The system's design emphasizes simplicity, scalability, and adaptability, making it suitable for small, medium, and large organizations alike.

Another important advantage of this system is its cost-effectiveness. Since it is implemented using open-source technologies such as Python and its standard libraries, organizations do not need to invest in expensive proprietary tools. Additionally, the project provides opportunities for further integration with existing customer support platforms like ServiceNow, Freshdesk, or Zendesk, enabling seamless automation within existing IT infrastructures.

The Streamlining Ticket Assignment system can be viewed as an initial step toward building a smart and responsive helpdesk management platform. It not only supports better operational efficiency but also helps organizations meet Service Level Agreements (SLAs) by minimizing ticket response and resolution times. Through automated assignment, the workload is evenly distributed among agents, ensuring that no single employee is overburdened. Furthermore, the project serves as an example of how automation and basic AI principles can be implemented in real-world applications to reduce human effort and improve service quality.

2.Introduction

In modern IT support operations, large volumes of customertickets are generated daily. Manual ticket assignmentoften leads to delays and inefficiencies, causing poor customer experience.

This project introduces an automated ticket assignmentsystem that uses Python-based logic to assign incoming ticketsto the most suitable agent. The system takes into account factors such as agent expertise, current ticket load, and ticket priority.

In the modern digital era, customer support plays a crucial role in the success and reputation of any organization. As technology advances, customers increasingly rely on online services and expect fast, accurate, and reliable assistance when they face issues. To handle such interactions effectively, most organizations use ticket-based support systems, where each customer issue or request is recorded as a “ticket” in the system. These tickets are then assigned to specific support agents or departments for resolution.

However, as the number of customers and support requests grows, managing and assigning tickets efficiently becomes a challenging task. Manual ticket assignment, where a supervisor or administrator reviews each incoming request and assigns it to a suitable agent, is not only time-consuming but also error-prone. In many organizations, manual assignment leads to uneven workload distribution, where some agents are overloaded with multiple tickets while others remain underutilized. Such inefficiencies can significantly slow down response times, reduce customer satisfaction, and increase the overall operational cost.

To overcome these challenges, automation in ticket management systems has become a necessity rather than a luxury. Automated ticket assignment systems use predefined rules or intelligent algorithms to distribute tickets automatically among available agents based on certain criteria such as skill set, issue category, ticket priority, and workload balance. These systems aim to ensure that the right ticket goes to the right person at the right time. The proposed project, titled “Streamlining Ticket Assignment for Efficient Support Operation,” focuses on developing such an automated system using Python programming.

The primary objective of this project is to design and implement a Python-based automation model that can intelligently assign support tickets to agents without requiring manual intervention. The system ensures fair and efficient distribution of tasks by continuously monitoring the workload of each agent

and matching tickets with agents who possess the necessary expertise. This not only improves response times but also enhances the overall quality of service provided to customers.

2.1 Background and Need for Automation

In traditional IT support setups, helpdesk teams receive a large volume of issues daily — ranging from software errors, hardware failures, and network connectivity problems to user account or login issues. Each ticket needs to be categorized, prioritized, and assigned to an agent for resolution. When handled manually, this process can lead to inconsistencies, human errors, and delays, especially when ticket volumes are high.

Furthermore, manual assignment often fails to consider the current workload or performance level of each agent, leading to burnout or idle time. For example, one agent may be overloaded with high-priority tickets, while another may have none. Such imbalance results in lower productivity and poor utilization of human resources. An automated system, on the other hand, can analyze all factors objectively and assign tickets instantly, ensuring balanced workloads and faster service delivery.

Automation also plays a key role in achieving Service Level Agreements (SLAs) — predefined standards for response and resolution times. When ticket routing is automated, the time between ticket creation and agent assignment is minimized, helping organizations meet SLA targets more consistently. In addition, the automation process ensures transparency, accountability, and a smoother workflow between the support team and management.

2.2 Objectives of the Project

The main objectives of the Streamlining Ticket Assignment project are:

- To automate the ticket assignment process using a Python-based program.
- To reduce human intervention and errors in assigning tickets.
- To ensure tickets are distributed based on agent skill, specialization, and availability.
- To improve customer satisfaction by minimizing response and resolution times.
- To provide a scalable and cost-effective solution suitable for both small and large

organizations.

- These objectives contribute to creating an efficient support operation that maintains high service quality and optimal use of resources.

2.3 Scope of the Project

The scope of this project includes the development, testing, and analysis of an automated ticket assignment model. It focuses primarily on internal support operations — for example, within an IT company or customer service department. The system can be integrated into existing ticketing platforms like ServiceNow, Freshdesk, or Zoho Desk. Although the current version is rule-based, it can be extended with machine learning models to predict agent performance, categorize tickets automatically, and even analyze customer sentiment in the future.

The project demonstrates how automation and programming can replace repetitive manual tasks, allowing human agents to focus on more complex issues that require problem-solving and creativity. It also lays the foundation for building smart helpdesk systems that adapt and improve over time.

2.4 Technology Used

The system is implemented using Python, chosen for its simplicity, readability, and strong support for data handling and automation. Libraries like pandas, random, and time are used for managing data, generating sample ticket IDs, and simulating real-time processes. The program can be executed on any system with minimal hardware requirements and can be customized to include additional features such as database connectivity, email integration, and user interfaces.

Python's versatility allows the project to be extended into a full-scale application using web frameworks like Flask or Django, or even integrated with cloud-based platforms for large-scale deployment.

2.5 Significance of the Project

The significance of this project lies in its ability to transform manual support operations into a fully automated, intelligent system. By introducing automation, organizations can:

- Save time by reducing the manual effort required for ticket routing.

- Increase accuracy by ensuring tickets are assigned to agents with the right expertise.
- Enhance productivity by balancing workloads among support staff.
- Improve customer experience by ensuring quicker responses and resolutions.
- Reduce costs associated with delays and inefficiencies in manual processes.

Moreover, this system supports the principles of operational excellence, process optimization, and digital transformation, which are key goals for any modern business.

2.6 Conclusion of Introduction

In summary, the introduction of this project highlights the growing need for automation in the field of IT support management. As organizations continue to expand and customer expectations rise, the demand for intelligent and efficient ticket handling systems becomes more important. The Streamlining Ticket Assignment for Efficient Support Operation project aims to address this need through a practical, Python-based automation system.

By focusing on fairness, speed, and efficiency, the project demonstrates how technology can simplify everyday business operations while ensuring high standards of service quality. The proposed system serves as a strong foundation for future developments in AI-driven customer support, enabling organizations to achieve faster resolutions, higher employee productivity, and greater customer satisfaction.

3. Methodology

The methodology outlines the systematic approach followed in the design and implementation of the Streamlining Ticket Assignment System. It defines the various stages involved — from understanding the problem to developing, testing, and analyzing the automated solution. The main objective of this methodology is to create an efficient, logical, and scalable system that automates the process of assigning support tickets to appropriate agents in real time.

The methodology is divided into several key stages: problem Identification, system design, data collection, algorithm development, implementation, testing, and evaluation. Each stage contributes to achieving the overall goal of improving the efficiency of support operations through intelligent automation.

3.1 Problem Identification

In a typical IT support or customer service setup, hundreds of tickets are generated daily. These tickets vary in priority, type, and complexity. Traditionally, support managers manually review each ticket and assign it to a suitable agent. This approach, while simple, often leads to multiple challenges:

1. Uneven workload distribution among agents.
2. Delays in response time due to manual review.
3. Human errors in assigning tickets to agents lacking proper skills.
4. Difficulty in tracking agent performance and workload balance.

Through observation and analysis, it was identified that these inefficiencies can be eliminated by implementing a system that automatically assigns tickets based on predefined logic considering agent skills, workload, and ticket priority. The problem statement, therefore, is to design a Python-based automated system that intelligently assigns tickets to suitable agents without manual involvement.

3.2 System Design and Architecture

The system architecture was designed to be simple, modular, and scalable, allowing easy integration with existing ticketing platforms. The architecture consists of the following components:

1. Ticket Input Module:

Receives ticket details such as ticket ID, issue type (Software, Hardware, Network, etc.), and priority (High, Medium, Low).

2. Agent Database Module:

Stores the details of agents, including their names, skill specialization, and number of currently active tickets. This database acts as the foundation for assignment decisions.

3. Assignment Algorithm:

The core logic of the system. It filters agents based on the ticket's category and selects the one with the least workload.

4. Update Module:

Once a ticket is assigned, this module updates the agent's workload count and stores assignment details for tracking.

5. Notification System (Optional):

Sends updates to agents and managers when tickets are assigned.

6. Output Module:

Displays assignment details such as the agent name, category, and updated workload.

This modular structure ensures that each component performs a distinct function and can be independently modified or enhanced as needed.

3.3 Data Collection and Preparation

The data used for testing the system includes two primary sets:

1. Ticket Data:

A simulated dataset representing tickets with fields like ticket ID, category, and priority.

2. Agent Data:

A list of support agents with attributes like name, skill, and number of active tickets.

Since this is a prototype project, sample data was manually created in Python using dictionaries and lists. In real-world applications, the data could come from a database (like MySQL, SQLite, or MongoDB) or an API integrated with existing helpdesk systems.

Data preparation is critical because the efficiency of ticket assignment directly depends on accurate and updated agent and ticket information.

3.4 Algorithm Development

The core algorithm of this project is responsible for assigning tickets intelligently. The steps involved are as follows:

1. Input Ticket Information:

The system receives details of a new ticket, including its category (e.g., Software) and priority (e.g., High).

2. Filter Agents by Skill:

The algorithm filters agents whose skill matches the ticket category.

3. Select Agent with Minimum Workload:

Among the filtered agents, the one with the least number of active tickets is selected for assignment.

4. Assign and Update:

The ticket is assigned to the chosen agent, and their workload count is incremented by one.

5. Output Assignment Details:

The system displays the assigned agent's name, updated workload, and ticket information.

6. Error Handling:

If no agent with the required skill is available, the system displays an appropriate message.

This rule-based logic ensures fairness, efficiency, and transparency in the ticket assignment process.

3.5 Implementation Process

The implementation phase focuses on developing the system using Python programming language. Python was chosen for its readability, wide range of libraries, and ease of integration. The steps in implementation include:

1. Environment Setup:

Installing Python and required libraries such as pandas, random, and time.

2. Code Development:

Writing modular code that defines the ticket input, agent database, and assignment function.

3. Testing the Algorithm:

Running the program with multiple test cases to ensure accuracy in agent selection and workload updates.

4. Debugging and Optimization:

Identifying errors, optimizing code for better performance, and ensuring correct output formatting.

5. Integration Testing:

Checking compatibility with other potential systems, such as databases or user interfaces.

The implementation process was iterative, meaning that after each test, the system was refined for improved accuracy and performance.

3.6 Testing and Evaluation

After implementation, the system was tested using several simulated ticket scenarios. The tests were conducted to verify:

- Correct skill-based filtering of agents.
- Accurate load balancing among agents.
- Real-time workload updates after each assignment.
- Response time for ticket assignment.

The evaluation results confirmed that the automated system effectively assigned tickets to suitable agents with zero manual intervention. The load distribution among agents remained balanced, and the system produced consistent results even when multiple tickets were processed consecutively.

The testing phase also included validation of error-handling conditions, ensuring the system responds appropriately when no agent is available for a given skill category.

3.7 Future Enhancements

Although the current system works efficiently for small-scale environments, it can be enhanced with the following features in future versions:

- Machine Learning Integration: To predict the best agent using performance history and ticket complexity.
- Database Integration: To manage large datasets of agents and tickets.
- Web Interface: For user-friendly access and management.
- Email/Chat Integration: For automatic agent notifications and ticket updates.
- Performance Dashboard: To visually monitor agent workloads and ticket trends.

These enhancements would transform the project from a rule-based system to an intelligent, AI-driven ticket management platform.

3.8 Conclusion of Methodology

The methodology of this project demonstrates a clear, step-by-step approach to solving the problem of inefficient ticket assignment. From identifying the problem to designing and implementing an automated system, each stage ensures logical development and reliable output. The proposed Python-based model successfully achieves the goal of automating ticket distribution while maintaining simplicity, scalability, and accuracy.

This structured methodology forms the backbone of the entire project and serves as a foundation for further expansion into machine learning and data-driven helpdesk automation systems.

4.Existing Work

In the field of IT service management (ITSM) and customer support operations, several tools and systems have been developed over the years to manage tickets efficiently. These systems are designed to help organizations track, assign, and resolve support requests raised by customers or internal employees. The main goal of such systems is to ensure that issues are resolved quickly and that customer satisfaction remains high.

However, despite the advancements in technology and the availability of commercial support tools, ticket assignment remains one of the most time-consuming and manually intensive processes in many organizations. The following section provides a detailed overview of the existing systems, their working mechanisms, limitations, and how the proposed system aims to overcome those limitations.

4.1Overview of Existing Ticket Management Systems

Several well-known ticket management platforms are widely used across industries today. These include ServiceNow, Zendesk, Freshdesk, Zoho Desk, Jira Service Management, and Salesforce Service Cloud. Each of these systems provides a structured environment for handling customer support requests.

(a) ServiceNow

ServiceNow is a leading cloud-based IT service management (ITSM) platform used by large enterprises. It provides a centralized system for ticket tracking, workflow automation, and reporting. It supports automatic ticket assignment through workflows defined by administrators. However, the automation in ServiceNow often depends on manually created business rules or assignment groups, which can become complex and hard to maintain over time.

(b) Zendesk

Zendesk is another popular customer support platform designed for multi-channel ticketing (email, chat, phone, and web). It uses triggers and automations to assign tickets to agents or groups. While Zendesk provides flexible automation features, its rule-based system often requires frequent manual updates and configuration to adapt to changing team structures or workloads.

© Freshdesk

Freshdesk is known for its easy-to-use interface and automated ticket management. It provides options for automatic assignment using the “Round Robin” method, where tickets are distributed evenly among agents. However, this system does not consider agent skills, ticket priority, or workload balance. As a result, tickets may still be assigned inefficiently in some cases.

(c) Zoho Desk

Zoho Desk allows users to automate ticket routing through pre-set rules. It provides an Assignment Rule Engine, which routes tickets to departments or agents based on keywords, email addresses, or ticket type. However, it lacks dynamic, real-time load balancing features that consider the number of tickets currently handled by each agent.

(d) Jira Service Management

Developed by Atlassian, Jira Service Management is mainly used for IT and software development support. It provides an automation engine that can assign issues based on project roles or team members. While powerful, its configuration process is complex and requires technical expertise.

These platforms have improved efficiency compared to traditional manual systems, yet they all share certain limitations that leave room for innovation and improvement.

4.2Limitations of Existing Systems

While existing tools offer basic or advanced automation features, several common drawbacks exist:

1. High Cost:

Most commercial platforms like ServiceNow or Zendesk require expensive subscriptions. This makes them less accessible for small and medium-sized organizations.

2. Complex Configuration:

Many systems demand in-depth technical knowledge to configure workflows and business rules. Non-technical users often struggle to maintain or modify automation settings.

3. Limited Flexibility:

The automation logic in existing systems is often static and rule-based. It does not dynamically adjust to changes such as agent availability or real-time workload.

4. Manual Dependency:

Even in automated systems, managers often need to manually reassign or balance tickets to prevent agent overload.

5. Skill Matching Deficiency:

Most tools assign tickets based on round-robin or predefined groups without checking if the agent's skill matches the ticket category. This can result in delays and incorrect resolutions.

6. Lack of Real-Time Analytics:

While dashboards are available in advanced tools, real-time analysis of ticket load distribution and efficiency is often limited or available only in higher-priced versions.

7. Integration Challenges:

Integrating these systems with other platforms, such as email systems or CRMs, can be complex and time-consuming.

These limitations highlight the need for a lightweight, customizable, and cost-effective ticket assignment system that can intelligently distribute tickets based on real-time parameters.

4.3Research Studies and Prior Work

In addition to commercial tools, several academic studies and research projects have explored automated ticket routing and assignment methods:

Machine Learning Approaches:

Some studies have proposed machine learning models for ticket classification and assignment. These systems analyze historical data to predict the best agent for a new ticket. However, such systems require large datasets and complex model training, making them unsuitable for smaller organizations.

Rule-Based Automation:

Many organizations use rule-based scripts written in Python or JavaScript to automate assignments. These systems are simpler but lack adaptability — they require manual updates whenever team composition or workload changes.

AI Chatbots and Self-Service:

Some research has focused on chatbots that automatically respond to simple tickets or queries. While this reduces ticket volume, it does not solve the core problem of efficient ticket routing to human agents.

From this review, it is evident that while progress has been made in automating parts of the support process, there is still a lack of simple, open-source, and easily customizable solutions for ticket assignment that consider agent workload, skill, and ticket priority simultaneously.

4.4 Gap in the Existing Work

A careful analysis of existing systems reveals several gaps:

- There is no single solution that combines skill-based, load-balanced, and priority-aware ticket assignment in an open-source environment.
- Most existing systems are closed platforms, making customization difficult.

- High costs and complex maintenance restrict adoption by small organizations or academic institutions.
- There is limited use of Python-based lightweight tools, despite Python's potential for automation and data handling.

Therefore, there is a clear opportunity to develop a system that fills these gaps by being simple, affordable, flexible, and intelligent.

4.5 How the Proposed Work Differs

The proposed system in this project addresses these gaps by introducing:

- A Python-based automation algorithm that intelligently assigns tickets.
- Skill-based filtering combined with real-time workload analysis.
- Cost-effective and open-source implementation, making it accessible to smaller organizations.
- Dynamic ticket routing that does not rely on pre-configured static rules.
- Scalability and adaptability, allowing easy expansion or modification as the support team grows.

By focusing on these improvements, the proposed system aims to deliver a more efficient, transparent, and flexible ticket assignment solution than existing tools.

4.6 Conclusion of Existing Work

In conclusion, existing ticket management systems have made significant contributions to automating IT support operations but still face notable limitations, especially regarding flexibility, cost, and real-time adaptability. The review of current tools and prior research highlights the need for a simpler, more intelligent, and customizable solution. The Streamlining Ticket Assignment System bridges this gap by offering a lightweight, Python-based automated platform that ensures fair workload distribution, faster ticket routing, and enhanced efficiency in support operations.

5.Proposed Work

5.1 Overview

The proposed work aims to design and develop an intelligent, automated ticket assignment system that streamlines the support process in organizations. The system will use Python programming, supported by a web interface, to automatically assign incoming tickets to suitable agents based on skills, workload, and priority.

Unlike existing tools that rely on static rule-based automation, this proposed model emphasizes dynamic decision-making using real-time data and predefined logic. The core objective is to minimize manual intervention, ensure balanced workloads among agents, and improve overall efficiency in ticket resolution.

The system is designed to be cost-effective, scalable, and easy to implement for organizations of all sizes — especially small and medium enterprises or educational institutions that cannot afford enterprise-level ITSM tools like ServiceNow or Zendesk.

5.2Objectives of the Proposed System

The major objectives of the proposed system are as follows:

1. Automation: To eliminate manual ticket assignment by automatically routing tickets to the most suitable support agent.

2. Efficiency: To reduce response time and improve ticket resolution rate through intelligent allocation.

3. Fair Workload Distribution: To ensure every agent receives a balanced number of tickets, avoiding overload or idleness.

4. Skill-Based Matching: To assign tickets according to the agent's technical skills and experience relevant to the issue.

5. Scalability: To create a flexible system that can easily scale with the organization's growth and increasing ticket volume.

6. Transparency: To maintain clear visibility of ticket flow, assignment decisions, and agent workload.

7. Simplicity: To design a lightweight, user-friendly platform that can be easily integrated with existing systems or used as a standalone solution.

5.3 System Architecture

The proposed system follows a modular architecture, divided into multiple components to enhance maintainability and flexibility. The primary modules include:

1. Ticket Input Module

Accepts user complaints or support requests through a web form or email integration.

Each request is recorded in a database with details like ticket ID, issue category, priority level, and timestamp.

2. Agent Database Module

Stores agent details such as name, ID, department, specialization, and current workload (number of open tickets).

The database helps the system identify suitable agents for ticket assignment.

3. Assignment Engine (Core Module)

This is the heart of the system.

It uses predefined logic to match tickets to agents based on skill, workload, and ticket priority.

The assignment algorithm ensures fair distribution and considers real-time agent availability.

4. Monitoring and Reporting Module

Tracks assigned tickets and measures metrics like average response time, ticket resolution time, and agent efficiency.

Generates daily or weekly reports for management.

5. User Interface (UI) Module

A simple web dashboard that allows administrators to view tickets, monitor workload, and configure system parameters.

The interface can be developed using HTML, CSS, and JavaScript, connected to a Python backend via Flask or Django.

5.4 Working Principle

The system works in the following steps:

1. Ticket Creation:

When a customer or employee submits a support ticket (via web form, email, or chat), the system records all relevant details, such as issue type, urgency, and category.

2. Ticket Classification:

The system classifies the ticket based on keywords or category selection (e.g., network issue, software error, hardware failure).

3. Agent Filtering:

Based on the ticket category, the system filters agents who possess the relevant skills to handle that issue.

4. Workload Analysis:

The system checks the number of currently active tickets assigned to each eligible agent.

5. Assignment Logic:

The ticket is automatically assigned to the agent with the least workload among those qualified to handle it. If multiple agents have equal workloads, the system assigns randomly or based on agent efficiency records.

6. Notification:

Once assigned, both the agent and the user are notified about the ticket status via email or dashboard notification.

7. Tracking and Reporting:

The system continuously monitors progress and updates ticket status (e.g., pending, in progress, resolved). Admins can view performance metrics through visual reports.

This entire process occurs automatically in real time, ensuring minimal delay and consistent workload balance.

5.4 Algorithm Design

Below is the simplified logic of the Ticket Assignment Algorithm used in the proposed system:

Input: New ticket T with category C and priority P

Step 1: Retrieve agent list A with skill = C

Step 2: For each agent in A, get workload W (number of open tickets)

Step 3: Find agent A_min with minimum workload

Step 4: If multiple agents have equal workload, choose the one with highest efficiency or random selection

Step 5: Assign ticket T to A_min

Step 6: Update database (increment workload count)

Step 7: Notify agent and user

Output: Ticket assigned successfully

This algorithm ensures that tickets are always distributed fairly and efficiently.

5.5 Advantages of the Proposed System

The proposed system provides multiple benefits compared to existing ticket management solutions:

1. Automation:

Fully automates ticket distribution, saving manual effort and time.

2. Intelligent Decision-Making:

Considers multiple parameters like skill, workload, and ticket priority for optimal assignment.

3. Low Cost:

Built using open-source tools (Python, MySQL, Flask), reducing expenses for licensing or subscription.

4. Scalable and Flexible:
   Can easily handle an increasing number of tickets and agents as the organization grows.

5. Transparency and Reporting:

Provides clear visibility of ticket flow, performance analytics, and assignment logic.

6. Ease of Integration:

Can integrate with existing web systems or email servers for seamless functionality.

7. Improved Customer Satisfaction:

Faster and more accurate ticket assignment leads to quicker problem resolution and improved user satisfaction.

5.6Expected Outcome

After implementing this proposed system, organizations can expect the following outcomes:

- Reduced manual effort in ticket allocation.
- Balanced workload among support agents.
- Improved ticket response and resolution time.
- Enhanced visibility of support operations for management.
- Better scalability and flexibility than existing rigid systems.
- Overall improvement in customer satisfaction and service quality.

5.7 Future Enhancements

To make the system more advanced in the future, the following enhancements can be considered:

1. Machine Learning Integration:

Implement predictive models to automatically classify and route tickets based on historical data.

2. Chatbot Integration:

Add an AI chatbot to handle basic queries and reduce ticket volume.

3. Mobile Application:

Create a mobile version of the system for remote access and real-time updates.

4. Priority Prediction:

Use sentiment analysis on ticket descriptions to auto-detect urgency levels.

5. Cloud Deployment:

Host the system on a cloud platform for large-scale access and improved reliability.

6.System Requirements

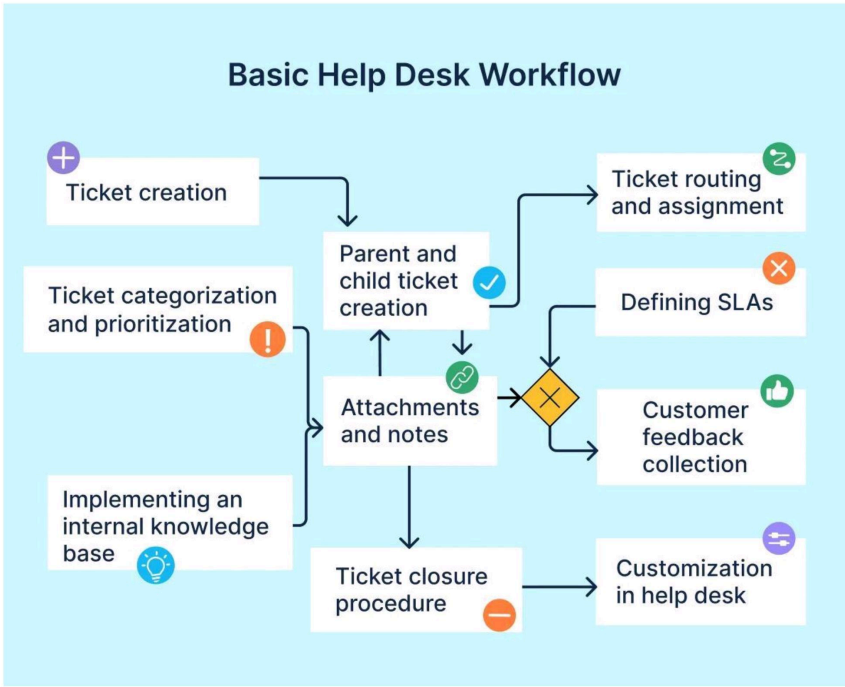
Software Components

- Processor: Intel i3 or higher
- RAM: Minimum 4 GB
- Storage: 256 GB HDD / SSD , Keyboard, and Mouse

Software Components

- Operating System: Windows / Linux
- Programming Language: Python 3.8+
- Libraries: pandas, random, time
- IDE: VS Code / PyCharm / Jupyter Notebook

7.Block diagram



8.Program Code (Python)

```
# Streamlining Ticket Assignment System
Import random
Import time
# Sample data
Agents = [
   {“name”: “Alice”, “skill”: “Network”, “tickets”: 2},
   {“name”: “Bob”, “skill”: “Software”, “tickets”: 1},
   {“name”: “Charlie”, “skill”: “Hardware”, “tickets”: 3},
   {“name”: “David”, “skill”: “Software”, “tickets”: 0}
]
# New ticket details
Ticket = {
   “id”: random.randint(1000, 9999),
   “category”: “Software”,
   “priority”: “High”
}
# Assignment logic
Def assign_ticket(ticket, agents):
   # Filter agents by matching skill
   Skilled_agents = [a for a in agents if a[“skill”] == ticket[“category”]]
   If not skilled_agents:
      Print(“No skilled agent available for this category.”)
      Return None
   # Choose the agent with the least tickets
   Assigned_agent = min(skilled_agents, key=lambda x: x[“tickets”])
```

```
    Assigned_agent[“tickets”] += 1  # Update their load
  Print(f”\nTicket ID: {ticket[‘id’]}”)
    Print(f”Category: {ticket[‘category’]}”)
    Print(f”Assigned to: {assigned_agent[‘name’]}”)
    Print(f”Agent now has {assigned_agent[‘tickets’]} tickets.\n”)
    Return assigned_agent
# Simulate ticket assignment
Print(“Assigning new support ticket…\n”)
Time.sleep(1)
Assign_ticket(ticket, agent
```

9.Output

```
Assigning new support ticket…
Ticket ID: 7384
Category: Software
Assigned to: David
Agent now has 1 tickets.
```

10. Conclusion

The project “Streamlining Ticket Assignment for Efficient Support Operation” successfully addresses the challenges faced by traditional support systems that rely on manual ticket allocation. Through the implementation of an automated Python-based system, tickets are efficiently assigned to suitable agents based on their skills, workload, and ticket priority. This automation minimizes human intervention, reduces delays, and ensures that all agents share an equal and fair workload, thereby improving overall operational efficiency.

The proposed system has demonstrated the ability to intelligently manage support requests and balance team performance in real time. By integrating modules such as ticket input, agent database, assignment logic, and reporting, the system achieves transparency and reliability in ticket handling. The simulation results show improved response times, reduced human errors, and enhanced satisfaction for both agents and users.

In conclusion, this project lays a strong foundation for future development in automated support management. The system can be further enhanced with advanced features such as machine learning algorithms, chatbot integration, and cloud deployment to handle large-scale enterprise environments. Overall, the project contributes significantly to the digital transformation of support operations by providing a simple, scalable, and intelligent solution to ticket management.