

Project Design Phase Report

Project Title:

Streamlining Ticket Assignment for Efficient Support Operations

Category:

ServiceNow System Administrator

Skills Required:

- TensorFlow
- ServiceNow
- Python
- REST API
- Data Analysis

1. Introduction

The **Project Design Phase** focuses on transforming the analyzed requirements and planning outputs into a well-structured system design.

For the project “*Streamlining Ticket Assignment for Efficient Support Operations*”, this phase defines the **architecture, data flow, module design, user interfaces, and integration mechanisms** between **ServiceNow** and the **TensorFlow-based intelligent model**.

The design phase ensures the solution is technically feasible, scalable, secure, and aligns with ServiceNow’s best practices.

2. Objectives of the Design Phase

- Define the **overall system architecture** and component interactions.
 - Design **data flow and control flow diagrams**.
 - Establish **database schema and data model design**.
 - Design **module-level structure and integration workflow**.
 - Plan **user interface (UI)** and **dashboard design** for administrators.
 - Ensure that the design supports scalability, performance, and maintainability.
-

3. System Architecture

The architecture of the project follows a **three-layered design**:

A. Presentation Layer

- User interface through the **ServiceNow dashboard**.
- Displays ticket status, assigned agents, and analytics.

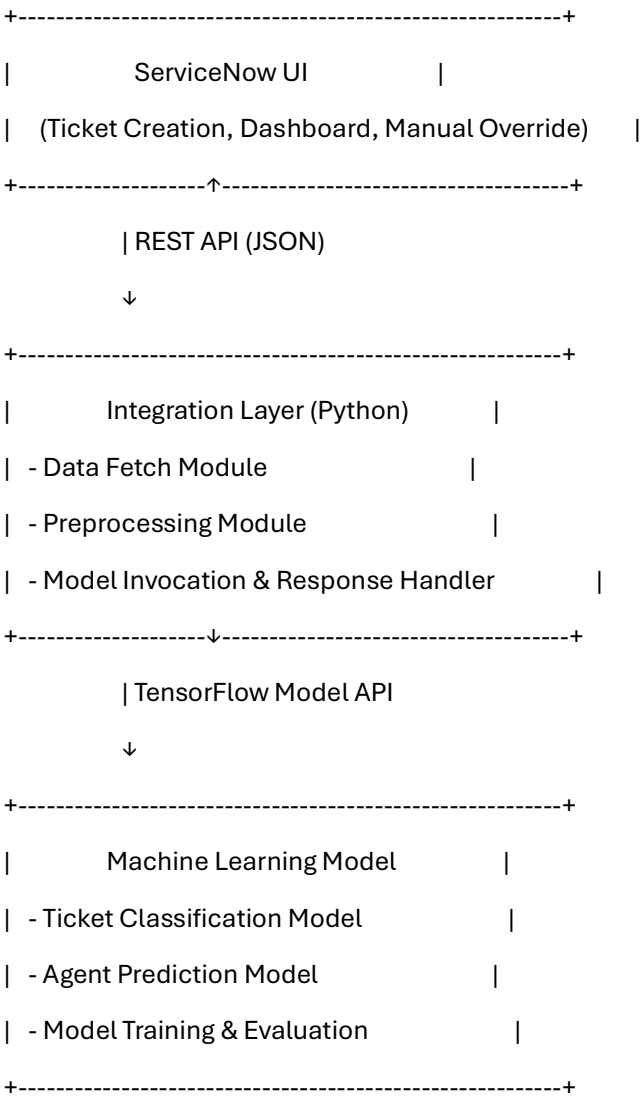
B. Application Layer

- **Integration Middleware**: Communicates between ServiceNow and the TensorFlow model via REST APIs.
- **Business Logic**: Handles data preprocessing, model invocation, and decision logic.

C. Machine Learning Layer

- **TensorFlow Model**: Predicts the best support agent for each ticket based on category, priority, and workload.
 - **Model Trainer**: Continuously improves accuracy by retraining with new data.
-

4. System Architecture Diagram



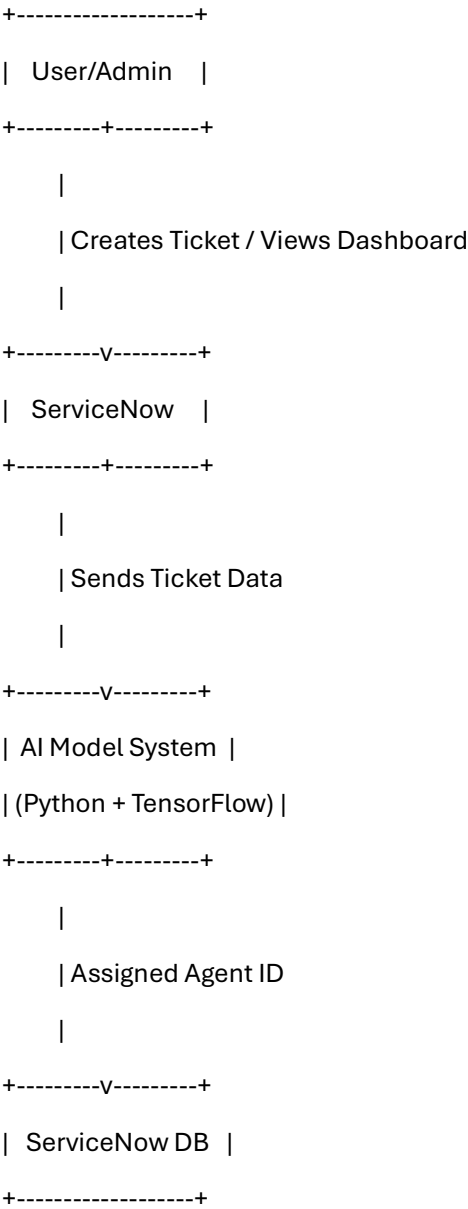
5. Module Design

Module Name	Description
Ticket Fetch Module	Connects to ServiceNow REST API and retrieves new or unassigned tickets.
Data Preprocessing Module	Cleans and formats ticket data (removes stopwords, encodes categories).
Model Training Module	Uses historical ticket data to train the TensorFlow model.
Prediction Module	Predicts the best agent based on trained model outputs.

Module Name	Description
Assignment Module	Updates ServiceNow with the assigned agent ID via REST API.
Monitoring & Dashboard Module	Displays metrics like model accuracy, ticket volume, and agent performance.

6. Data Flow Diagram (DFD)

Level 0 (Context Diagram)



7. Database Design

Tables and Fields

1. Ticket Table

Field Name	Data Type	Description
Ticket_ID	String	Unique ID for each ticket
Description	Text	Problem statement of ticket
Priority	String	Low / Medium / High
Category	String	Type of issue (Network, Software, Hardware)
Assigned_Agent	String	Agent ID predicted by ML model
Created_Date	DateTime	Date ticket was raised
Status	String	Open / Assigned / Resolved

2. Agent Table

Field Name	Data Type	Description
Agent_ID	String	Unique identifier for each support agent
Name	String	Agent's name
Skill_Set	String	Agent's area of expertise
Current_Workload	Integer	Number of active tickets
Availability	Boolean	Agent availability status

3. Model_Performance Table

Field Name	Data Type	Description
Model_ID	String	Version identifier of trained model
Accuracy	Float	Model's prediction accuracy
Last_Trained	DateTime	Date when last retraining occurred

8. Algorithm Design

A. Data Preprocessing Steps

1. Extract data from ServiceNow (JSON format).
2. Remove null or irrelevant fields.
3. Convert text to lowercase and remove stopwords.
4. Perform **tokenization** and **vectorization (TF-IDF)**.
5. Encode categorical fields (e.g., category, priority).

B. Model Training Algorithm (TensorFlow)

1. Input: Ticket description, category, priority, previous assignments.
2. Split data into training and testing sets (80:20).
3. Train a **Neural Network Classifier** to predict the agent.
4. Evaluate model accuracy and save the trained model.

C. Ticket Assignment Algorithm

1. Receive new ticket from ServiceNow.
2. Process description and extract key features.
3. Load trained TensorFlow model.
4. Predict the most suitable agent.
5. Send agent details back to ServiceNow via API.

9. Interface Design

A. ServiceNow UI

- Displays all incoming tickets and assigned agents.
- Includes “Override” and “Reassign” options for administrators.

B. Monitoring Dashboard

- Metrics:
 - Total tickets processed.
 - Model accuracy percentage.
 - Agent workload distribution.
 - Response time and SLA compliance.
-

10. Integration Design

The **integration** between ServiceNow and TensorFlow is achieved via **RESTful APIs**.

Action	API Endpoint	Method	Description
Fetch Tickets	/api/now/table/incident?state=new	GET	Fetch unassigned tickets
Assign Agent	/api/now/table/incident/{ticket_id}	PUT	Update ticket with agent ID
Monitor Tickets	/api/now/table/incident	GET	Fetch ticket metrics for dashboard

Authentication is done using **ServiceNow OAuth tokens**, ensuring secure communication.

11. System Security Design

- OAuth 2.0 authentication for API access.
- Encrypted data transmission using HTTPS.
- Access control based on ServiceNow roles (Admin, Support Agent, Viewer).
- Model files stored securely with restricted access.

12. Design Constraints

- Integration limited to ServiceNow developer instance.
- Dataset availability may impact model accuracy.
- Real-time response requirement (<2 seconds).

13. Design Evaluation

- **Accuracy:** Model evaluated with validation data to achieve >85% accuracy.
- **Performance:** API response time tested under load.
- **Scalability:** Modular architecture supports future model updates.

14. Expected Outcomes

- A well-structured AI-driven ticket assignment system integrated with ServiceNow.
 - Smooth data flow between ServiceNow and TensorFlow model.
 - Intuitive dashboard for administrators.
 - Secure and efficient automation improving support team productivity.
-

15. Conclusion

The **Design Phase** establishes the technical blueprint for the entire system.

It defines **how each module interacts**, **how data flows**, and **how the model integrates** within the ServiceNow environment.

This structured design ensures that the implementation phase can proceed efficiently, with minimal ambiguity and maximum system performance.