Project Design Phase – II

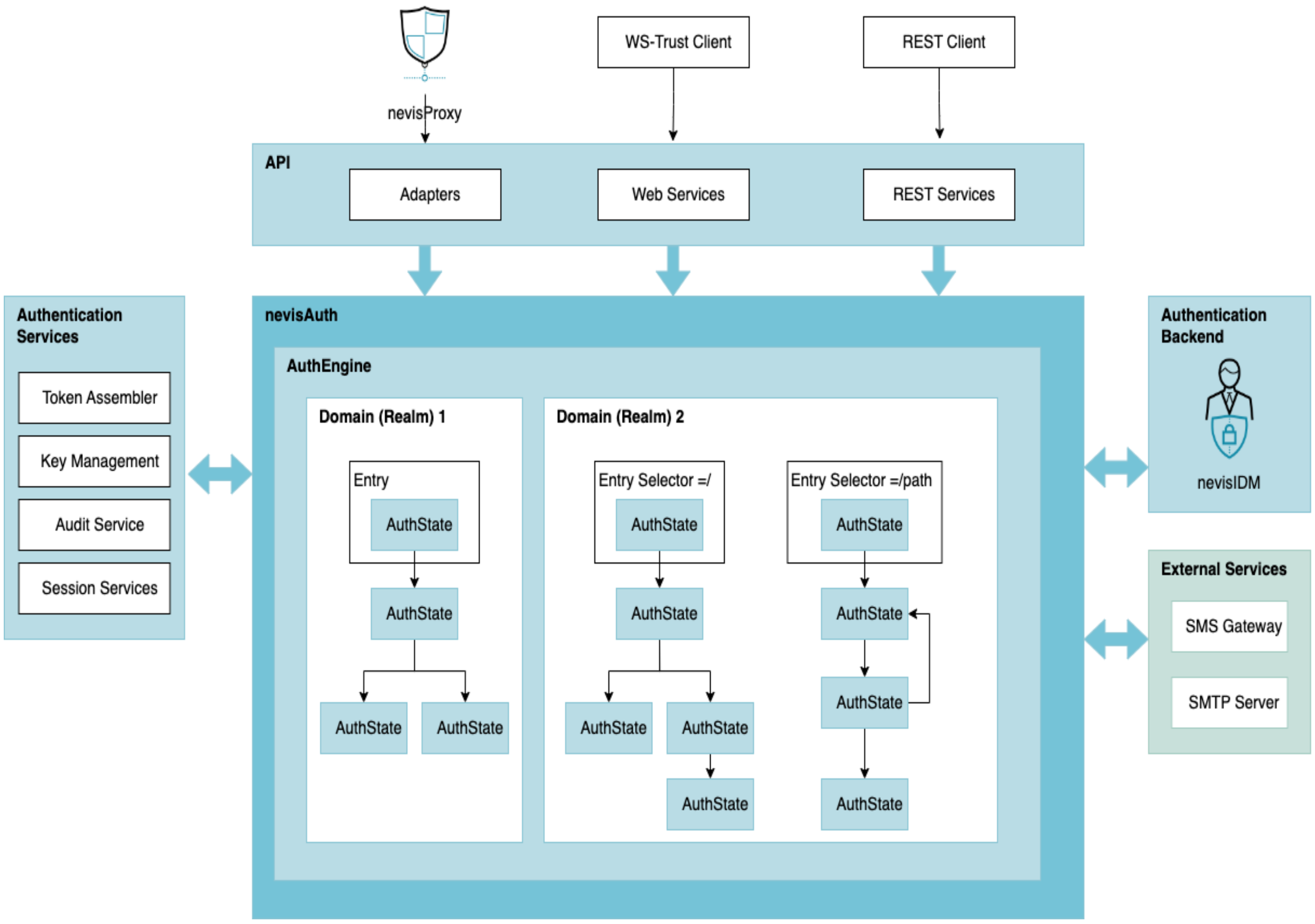
Technology Stack (Architecture & Stack)

Date	29 October 2025
Team ID	NM2025TMID08331
Project Name	Streamlining Ticket Assignment for Efficient Support Operations
Maximum Marks	4 Marks

Technical Architecture:

The deliverable includes an overview of the system architecture that supports automated ticket assignment for support operations.

Example: Order processing during pandemics for offline mode



Guidelines Followed:

- Includes all processes as application logic/technology blocks.
- Provides infrastructural separation between frontend, backend, and cloud components.
- Highlights integration with external APIs and cloud databases.
- Shows data storage, processing, and monitoring workflows.
- Indicates where automation and analytics occur in the stack.

Table 1: Components & Technologies

S.No	Component	Description	Technology
1.	User Interface	Users, agents, and admins interact through a responsive web dashboard.	HTML, CSS, React.js
2.	Application Logic – 1	Handles ticket creation, categorization, and priority assignment.	Java, Spring Boot
3.	Application Logic – 2	Automates ticket assignment using workload and skill-based logic.	Java, RESTful APIs
4.	Application Logic – 3	Generates notifications and updates ticket status in real-time.	Firebase Cloud Messaging
5.	Database	Stores user, agent, and ticket details.	MySQL
6.	Cloud Database	Cloud-hosted database for remote access and scalability.	AWS RDS
7.	File Storage	Stores reports, logs, and backups securely.	AWS S3
8.	External API – 1	Integrates with email or chat systems for ticket notifications.	Gmail API, Twilio API
9.	External API – 2	Optional integration with HRMS or CRM tools for user data.	REST API
10.	Machine Learning Model	Optional model for predicting ticket priority and resolution time.	Python, scikit-learn
11.	Infrastructure (Server / Cloud)	Application hosted and managed on cloud environment.	AWS EC2, Docker Containers

Table 2: Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Utilizes open-source frameworks for flexibility and customization.	React.js, Spring Boot
2.	Security Implementations	Implements authentication, role-based access, and encryption.	JWT, HTTPS, OAuth 2.0
3.	Scalable Architecture	Supports scaling of both the backend and database layers.	AWS Auto Scaling
4.	Availability	High availability through cloud infrastructure and backup systems.	Load Balancers, AWS EC2
5.	Performance	Optimized backend with asynchronous request handling and caching.	Redis, Java Threads