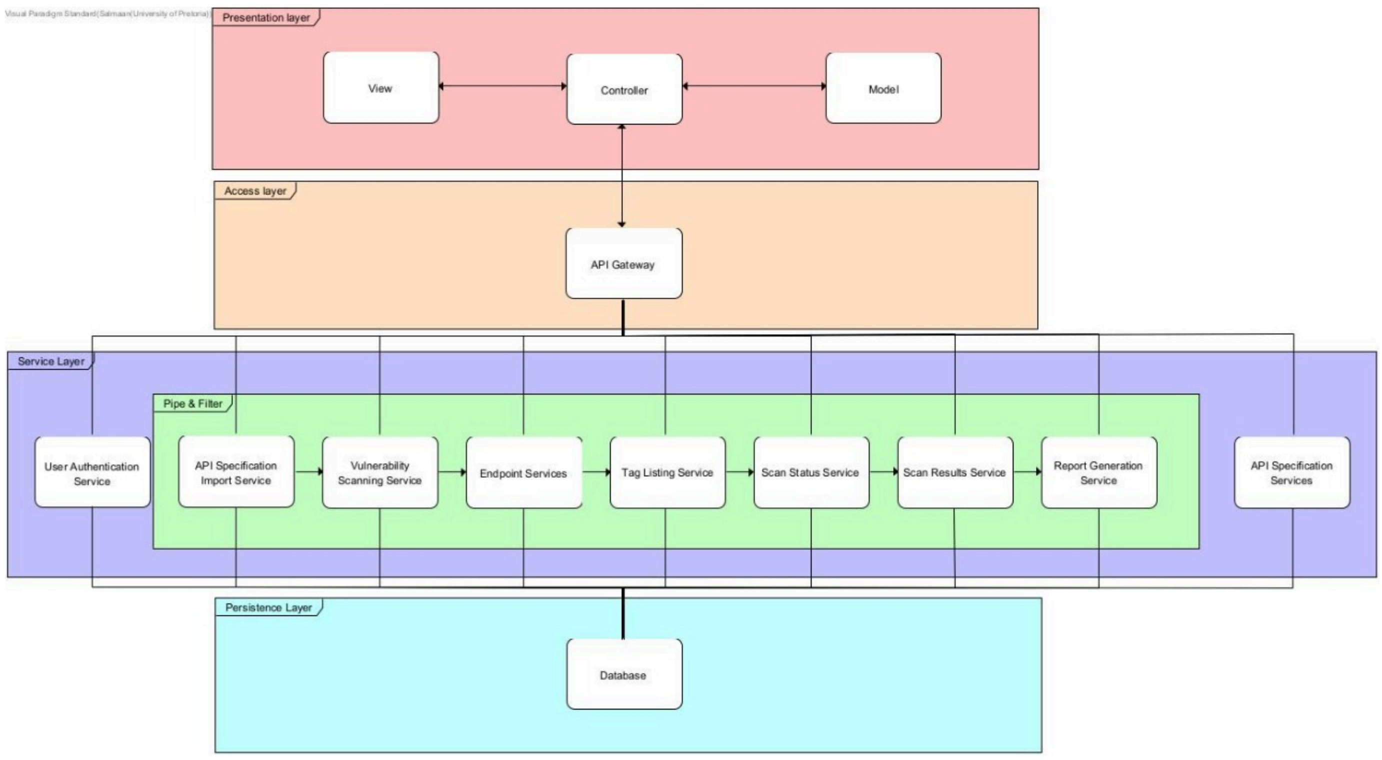
Software Architecture

System Architecture Overview

The AT-AT system adopts a **microservices-inspired modular architecture** designed to support independent development, scalability, and maintainability. The system consists of clearly separated logical components, each responsible for a specific function.

Diagram


Visual Paradigm Standard(Salmaan(University of Pretoria))
Presentation layer
View
Controller
Model
Access layer
API Gateway
Service Layer
Pipe & Filter
User Authentication Service
API Specification Import Service
Vulnerability Scanning Service
Endpoint Services
Tag Listing Service
Scan Status Service
Scan Results Service
Report Generation Service
API Specification Services
Persistence Layer
Database

Layers

1. Presentation Layer (Frontend)

- Built with React.js.
- Responsible for rendering the user interface.
- Sends requests to the API for scans, report generation, and user authentication.

2. API Layer (JS Express API)

- Acts as a middleware between frontend and backend processing.
- Performs user session validation, RESTful request parsing, and dispatches tasks to Python backend.

3. Processing Layer (Python Backend)

- Executes scanning logic, report generation, and vulnerability detection.
- Communicates with the API layer through internal HTTP calls.
- Reads environment variables for secure credentials.

Architectural Justification

The separation into distinct layers follows microservices principles to support:

- **Scalability**: Each component (Frontend, API, Backend) can be deployed or scaled separately.
- **Maintainability**: Isolated codebases reduce complexity when updating or testing.
- **Fault Isolation**: Issues in one layer (e.g., backend processing) do not bring down the UI or authentication.

> Note: No direct communication occurs between the frontend and the backend processor — all traffic is routed through the API layer.

Non-Technical Notes

- This architecture is **technology-independent** in concept.
- It may be deployed to Docker, virtual machines, or dedicated servers, without altering the structural design.
- Quality Attributes (QAs) such as **performance**, **modularity**, and **reliability** are enhanced by this separation of concerns.