High-Level Design Document

Project: Segment-Driven Rule Generation System

Audience: Engineering & Product Teams

Author: Kunwar Vivek Pratap Singh

Date: April 2025

Version: 1.0

1. Introduction

This document outlines the high-level architecture of the Segment-Driven Rule Generation System. The system aims to dynamically generate human-readable rules based on model-generated segment definitions. It supports efficient serving and internal caching of these rules, driven by a backend service and continuously monitored by a scheduler. The design caters to both engineering implementation and product-level clarity.

2. Scope

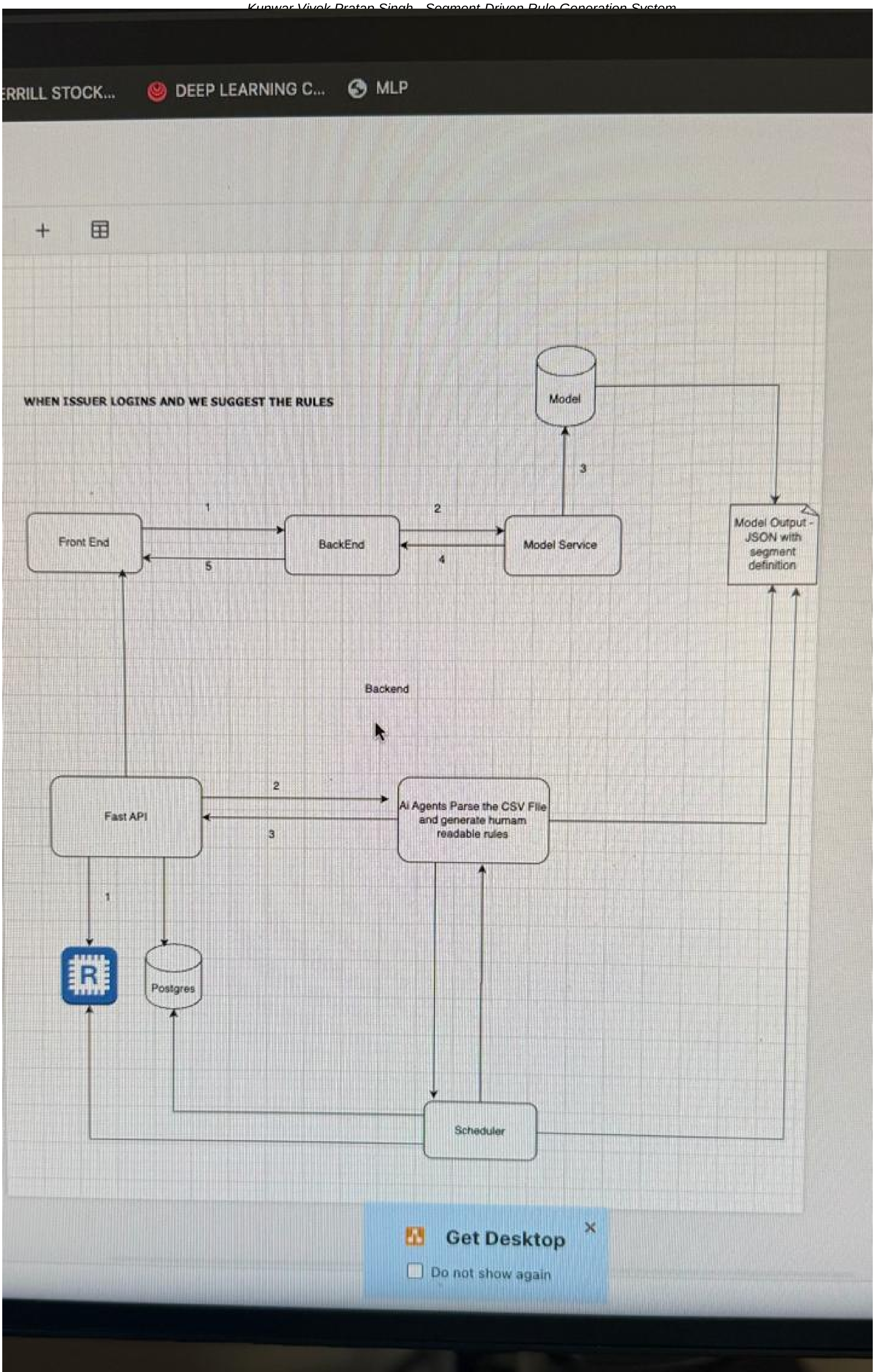
This HLD covers the architecture, components, and interactions involved in generating and serving rules based on segment definitions. It includes both user-facing flows and internal background processes like cache updating, without separating them structurally. The target users of this document are engineers building the backend and frontend, and product managers defining use cases around segment rules.

3. Big Picture

The system's primary objective is to deliver user-specific fraud detection rules based on underlying behavioral segments. Segment definitions are generated by an ML model running in the MLP (Machine Learning Platform) using Hadoop. These are stored in Model Storage and then parsed into readable rules by AI agents (Crew Agents).

The backend orchestrates communication between the Frontend, Model Service, Cache, and AI Agents. A scheduler monitors changes in segment definitions and triggers rule updates automatically.

4. Architecture Diagram


ERRILL STOCK...
DEEP LEARNING C...
MLP
WHEN ISSUER LOGINS AND WE SUGGEST THE RULES
Model
3
1
2
Front End
BackEnd
Model Service
5
4
Model Output - JSON with segment definition
Backend
2
Fast API
Ai Agents Parse the CSV File and generate humam readable rules
3
1
R
Postgres
Scheduler
Get Desktop
Do not show again

5. Assumptions

- Segment definitions are always stored in a well-structured JSON format.
- MLP continuously feeds updated segments into Model Storage.
- The AI Agents can parse any segment definition without manual intervention.
- Redis or equivalent caching layer will be available for low-latency rule access.
- The Scheduler will either be a Cron Job or Airflow DAG capable of polling Model Storage efficiently.

6. Component Descriptions

- **Frontend (React)**: Interface for issuers to log in and view their personalized rules.
- **Backend (FastAPI)**: Orchestrates all flows; interfaces with model service, AI agents, and cache.
- **Model Service (Python Flask/FastAPI)**: Fetches segment JSONs from Model Storage.
- **Model Storage**: Holds segment definitions in a structured format, updated by the MLP.
- **MLP (Hadoop + ML Models)**: Background engine that identifies new user segments based on behavior.
- **AI Agents (Crew Agents)**: Parses raw JSON segments into human-readable fraud rules.
- **Cache**: Stores parsed rules for quick access and UI delivery (likely Redis or equivalent).
- **Scheduler (Cron/Airflow)**: Periodically polls Model Storage, detects new segments, and invokes AI agents to refresh cache.

7. System Workflow

End-to-End Unified Flow:
1. Issuer logs into the system via Frontend.
2. Backend receives request and checks the Cache.
3. If rule is not cached, Backend calls Model Service.
4. Model Service fetches the corresponding segment JSON from Model Storage.
5. Segment is passed to AI Agents.
6. AI Agents return human-readable rules.
7. Backend caches the rules and returns them to the Frontend.

Internal Cache Update Flow (within Serving Flow):
1. Scheduler runs periodically (via Airflow or Cron).
2. It queries Model Storage for new/updated segments.
3. If found, Scheduler triggers AI Agents with the segment JSON.
4. Parsed rules are saved into Cache for future serving.

This ensures rules are fresh and available without delay when requested by issuers.

8. Glossary

- **Segment**: A group of behaviors derived from user transaction data using ML.
- **Segment Definition**: A JSON representation of thresholds, tags, and behaviors related to fraud patterns.
- **MLP**: Machine Learning Platform that analyzes Hadoop data and updates segments.
- **Model Storage**: Persistent store holding all segment definitions.
- **AI Agents**: Services (Crew Agents) that parse segment definitions to natural language.
- **Cache**: Fast-access in-memory store used to reduce processing latency.
- **Scheduler**: Background polling job that updates cache when new segments are detected.
- **Frontend**: User-facing layer built in React.
- **Backend**: Python-based orchestrator using FastAPI.