Gold Card Module Design Document

1. Introduction

The Gold Carding Module aims to streamline the prior authorization (PA) process within the healthcare sector by automating approval for high-performing providers based on predefined criteria. This document outlines the software design for the Gold Carding Module, including system architecture, database schema, API endpoints, and functionality details.

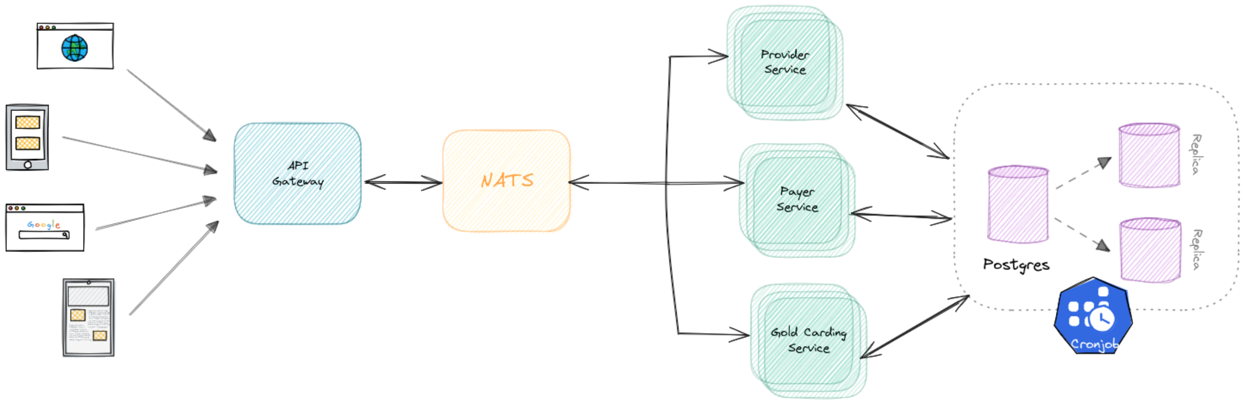
2. Problem Statement

The problem at hand is to streamline the prior authorization process in healthcare by developing a Gold Carding Module. This module aims to automate the identification of high-performing healthcare providers who consistently meet predefined criteria, such as approval rates, submission volumes and adherence to treatment guidelines. By granting expedited approval for certain procedures to these providers, the module seeks to improve efficiency, reduce administrative burdens, and enhance patient access to necessary treatments, ultimately optimizing the healthcare system.

3. System Overview

Gold Card Module Design Document

The Gold Carding Module is designed to be scalable, resilient, and efficient following a microservices architecture pattern. This section outlines the key components and their interactions within the system.



Components

1. **API Gateway**:
   - Serves as the entry point for client requests, routing requests to appropriate microservices over NATS Server.
   - Handles API authentication, request validation, rate limiting, and request transformation.
2. **Message Broker (NATS)**:
   - Facilitates asynchronous communication between microservices.
   - Provides publish-subscribe messaging model for decoupled interactions.
   - Ensures reliable message delivery and fault tolerance.
3. **Provider Service**:
   - Responsible for managing provider data, including creation, retrieval, update, and deletion of provider records.
   - Publishes provider data updates to the message broker for other services to consume.
4. **Payer Service**:
   - Manages payer data, gold carding rules, and eligibility criteria.
   - Subscribes to provider data updates from the message broker to perform eligibility checks and rule evaluations.
5. **Gold Carding Service**:
   - Evaluates providers against gold carding criteria defined by payers.

   - Subscribes to provider and payer data updates to perform evaluations and assign gold carding levels.
   - Publishes evaluation results and gold carding status updates to the message broker for other services to consume.

6. **Job Scheduler**: Executes scheduled tasks for auto-approved request checks.

Overall, the API Gateway could publish messages to NATS topics or queues, allowing other microservices to consume them asynchronously.

Data Flow

The data flow between the API Gateway and other microservices over the NATS server typically involves a combination of HTTP requests and asynchronous messaging. Here's how the data flow might occur:

1. **Client Request Handling**:
   - Client requests are routed through the API Gateway, which performs authentication, validation, rate limiting and request transformation.
   - The API Gateway forwards valid requests to the appropriate microservice over **NATS** server.
2. **Provider Data Management**:
   - The Provider Service receives requests to create, update, retrieve, or delete provider records.
   - Upon data modification, the Provider Service publishes provider data updates to the message broker.
3. **Payer Data Management and Rule Evaluation**:
   - The Payer Service manages payer data and gold carding rules.
   - It subscribes to provider data updates from the message broker to perform eligibility checks and rule evaluations.
   - Eligibility results and rule evaluations are stored within the Payer Service.
4. **Gold Carding Evaluation**:
   - The Gold Carding Service subscribes to provider and payer data updates from the message broker.
   - It evaluates providers against gold carding criteria based on data received.
   - Evaluation results, including gold carding status and remarks, are published to the message broker for other services to consume.

Gold Card Module Design Document

4. Design Considerations

Assumptions

- Data provided for providers, payers, and gold carding criteria is accurate, validated and properly formatted.
- The system architecture incorporates load balancing, caching mechanisms, cluster setup and efficient database indexing for scalability and performance.
- The system implements encryption, access controls, audit trails, and regular security assessments to maintain data security and compliance.
- Dashboard interfaces, analytics tools, and reporting mechanisms are available for stakeholders to monitor system performance and outcomes.

These assumptions provide a foundation for designing and implementing the Gold Carding Module, ensuring that the system meets the requirements effectively and delivers value to stakeholders in the healthcare domain.

Constraints for Gold Card and Auto Approval

In addition to the given gold carding criteria, here are some additional conditions or metrics that could be considered for evaluating providers and determining gold carding eligibility highlighted in yellow:

Metric	**Threshold** (vary based on rules and eligibility)	**Measurement Period**
approval_rate	0.95	12 months
submission_volume	100	12 months
readmission_rate	0.03	12 months
treatment_guideline_adherence	0.95	12 months
patient_satisfaction	0.8	12 months
recovery_rate	0.9	12 months
cost_efficiency	0.7	12 months

Gold Card Module Design Document

1. **Gold Card Levels**:

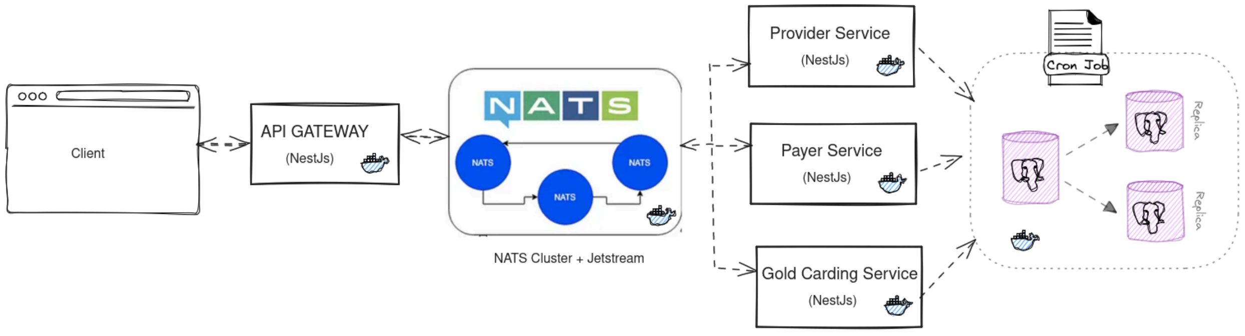
Level	**Meets Criteria**
Premier	All
Advanced	Most
Basic	Least

2. **Auto-Approval Future Requests**:

MRI scan prior authorization requests will automatically receive approval if they maintain a 100% approval rate over a specified measurement period, such as 12 months.

5. Architecture Design

Design the architecture for the Gold Carding Module using three microservices (Provider Service, Payer Service, and Gold Carding Service) with Nest.js and PostgreSQL communicating via an API Gateway and NATS, we'll outline the components, interactions, and technologies involved:



In this diagram:

- The client interacts with the system through the API Gateway.
- The API Gateway forwards requests to the appropriate microservices.
- Microservices (Provider Service, Payer Service, and Gold Carding Service) communicate asynchronously via NATS for data exchange and event notification.
- The PostgreSQL Database stores data used by the microservices.

- All microservices are developed using Nest.js framework and event-driven messaging method using Client Nest Classes (**ClientModule** for registering microservices & **ClientProxy** to get instance of the service)

6. Database Design

The database design section outlines the structure and functionality of the database components used in the Gold Carding Module.

Tables and Schema

- **Providers Table**:
  - Stores information about healthcare providers, including provider ID, name, specialty, and contact details.
- **CPT Code Table**:
  - Contains Current Procedural Terminology (CPT) codes relevant to gold carding criteria, along with their descriptions.
- **Provider CPT Approval Table**:
  - Records approval status for specific CPT codes submitted by providers, including denial reasons if applicable.
- **Payers Table**:
  - Stores information about insurance payers, including payer ID, name, and description.
- **Gold Carding Criteria Table**:
  - Defines gold carding criteria set by payers, specifying metrics, thresholds, and measurement periods for evaluation.
- **Provider Gold Carding Status Table**:
  - Tracks gold carding status for individual providers, including criteria met, gold carding level, and validity periods.

Integration with pg_cron Extension

Utilize the pg_cron extension in PostgreSQL to automate the periodic checking of auto-approved requests. This process ensures timely evaluation of requests meeting predefined criteria, contributing to the efficiency and accuracy of the gold carding system.

- **Cron Job Creation**:
  - The pg_cron extension enables the creation of cron jobs directly within the PostgreSQL database.
  - A cron job is scheduled to run periodically, querying the database for auto-approved requests based on predefined conditions.
  - The SQL Function executed by the cron job identifies auto-approved requests and logs them in file accordingly.

By automating the process of checking auto-approved requests via Job Scheduler, the Gold Carding Module improves operational efficiency and reduces manual intervention.

7. API Design

API endpoints are designed to provide functionality for managing provider and payer data, defining gold carding criteria, evaluating providers, and handling auto-approved requests. The API endpoints include:

- Provider CRUD endpoints for managing provider data.
- Payer CRUD endpoints for managing payer data and gold carding rules.
- Gold carding criteria endpoints for defining and managing gold carding criteria.
- Evaluation endpoints for assessing providers against gold carding criteria.

8. Acceptance Testing

- **Functional Testing**:
  - Validate provider and payer data management functionality.
  - Verify correct evaluation of providers against gold carding criteria.
- **Integration Testing**:
  - Ensure seamless communication between API Gateway and microservices.
  - Validate reliability of message exchange via NATS.
- **Performance Testing**:
  - Measure database and microservice performance under varying loads.
  - Assess scalability and responsiveness of the system.
- **End-to-End Testing**:
  - Simulate auto-approval workflow and cron job functionality.
  - Validate accuracy of auto-approval evaluations.

- **User Acceptance Testing (UAT)**:
  - Engage stakeholders in the testing process.
  - Validate system usability and effectiveness in real-world scenarios.

Comprehensive acceptance testing ensures the Gold Carding Module meets functional requirements, integrates seamlessly, performs well under load, functions end-to-end and satisfies user needs.