

# Smart Clinic Management System (SCMS)

This document provides an in-depth architectural and design documentation for the Smart Clinic Management System (SCMS). It focuses on the Class Diagram and the Outbox Pattern used to support a modular monolith architecture and its evolution toward a microservices based system. The document is written to meet academic and production-level standards and it is intended to be used directly as part of the final project report.

## 1. Introduction

Modern health care systems require reliability, scalability, and maintainability. SCMS is designed to digitize clinic operations such as patient registration, appointment scheduling, prescription management, billing, and notifications. The system adopts Domain-Driven Design (DDD) principles and an event-driven architecture to ensure loose coupling and future scalability. This document focuses on two critical architectural artifacts: the Class Diagram and the Outbox Pattern. Together, these artifacts ensure a clear domain model and reliable asynchronous communication between bounded contexts.

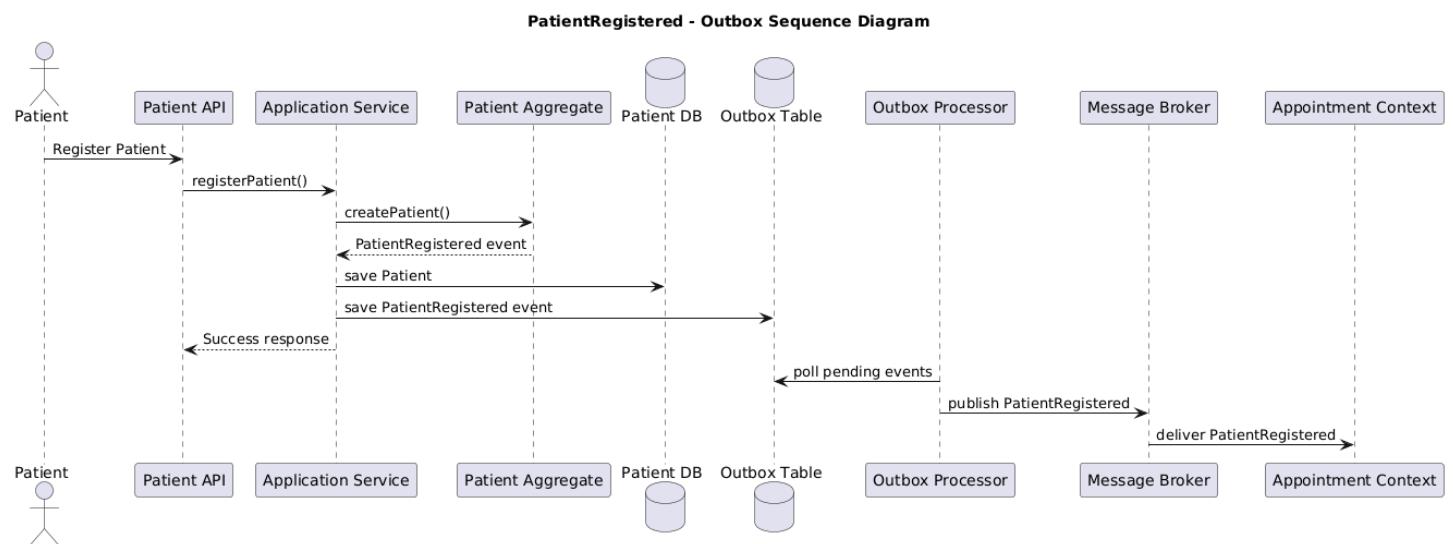
## 2. Architectural Overview

SCMS is implemented as a modular monolith. Each bounded context is logically separated into modules with clear responsibilities. Over time, selected modules such as the AI Assistant or Notification service can be extracted into independent microservices without major refactoring. The architecture emphasizes separation of concerns, observability, security, and eventual consistency. Communication between contexts is achieved through domain events published via a message broker such as RabbitMQ or Kafka.

### 2.1 Patient Bounded Context

The Patient context is the core repository of clinical truth within the system. It handles identity, medical history, and clinical observations.

#### 2.1 Behavior & Sequence Flow

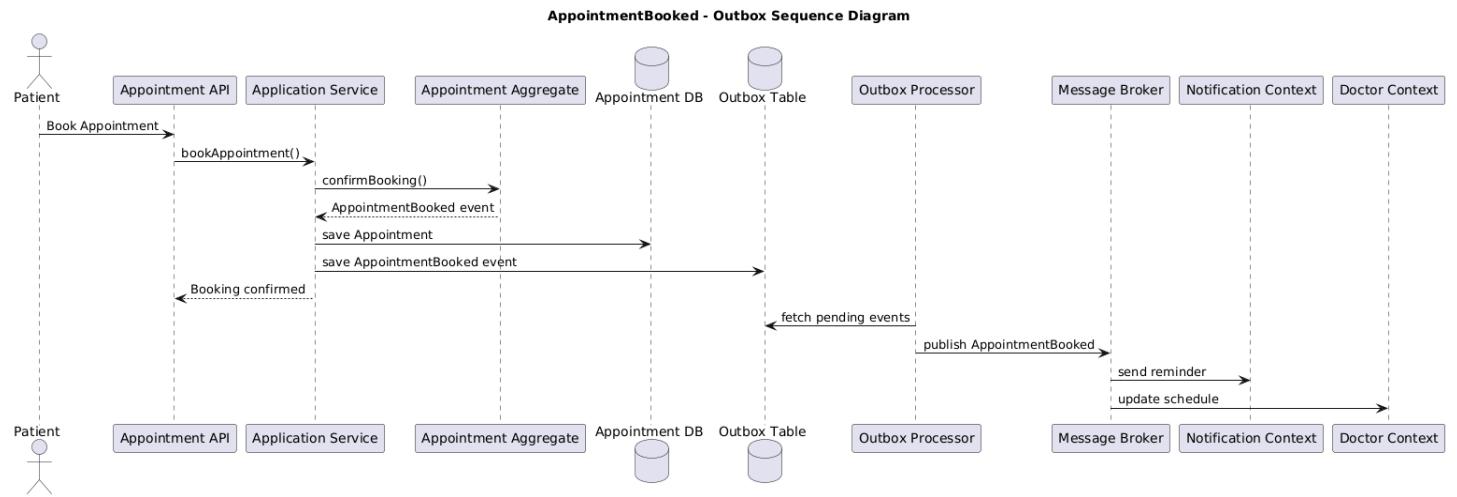


The sequence diagram illustrates the "Register Patient" flow. When a request hits the Minimal API.

## 2.2. Appointment Bounded Context

The Appointment context manages the orchestration of time, resources, and clinical staff availability.

### 2.2.1 Behavior & Sequence Flow

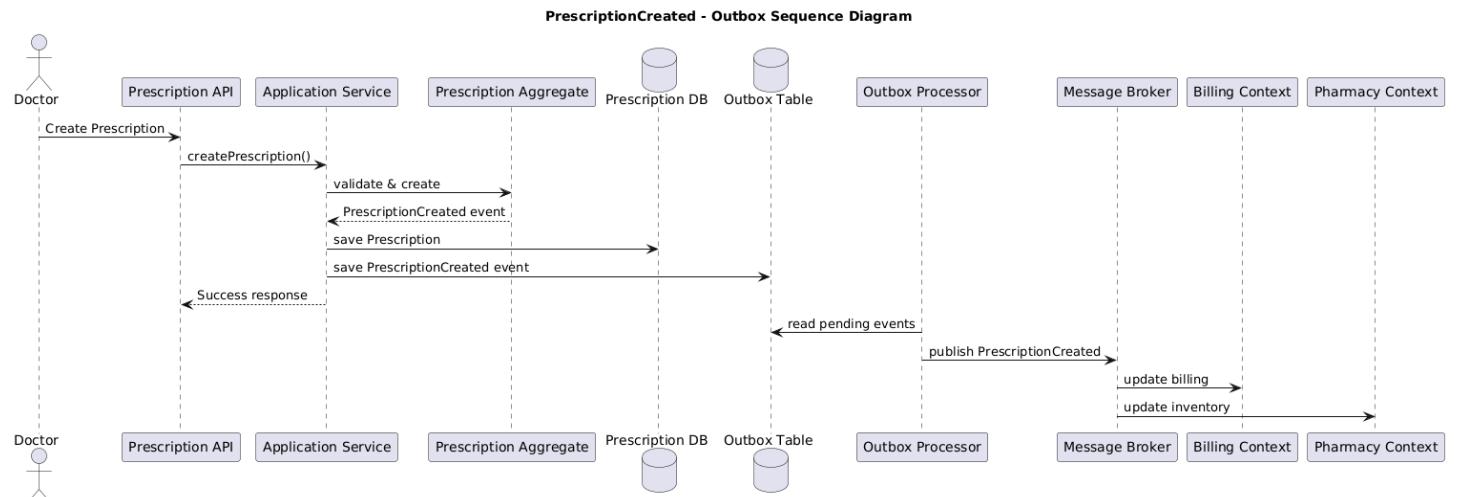


In this flow, when an appointment is "Confirmed," the system triggers an **AppointmentBooked Domain Event**. This event is critical because it signals the transition of responsibility from the administrative staff to the clinical staff.

## 2.3. Prescription Bounded Context

This context handles the lifecycle of medication orders, ensuring patient safety and regulatory compliance

### 2.3.1 Behavior & Sequence Flow

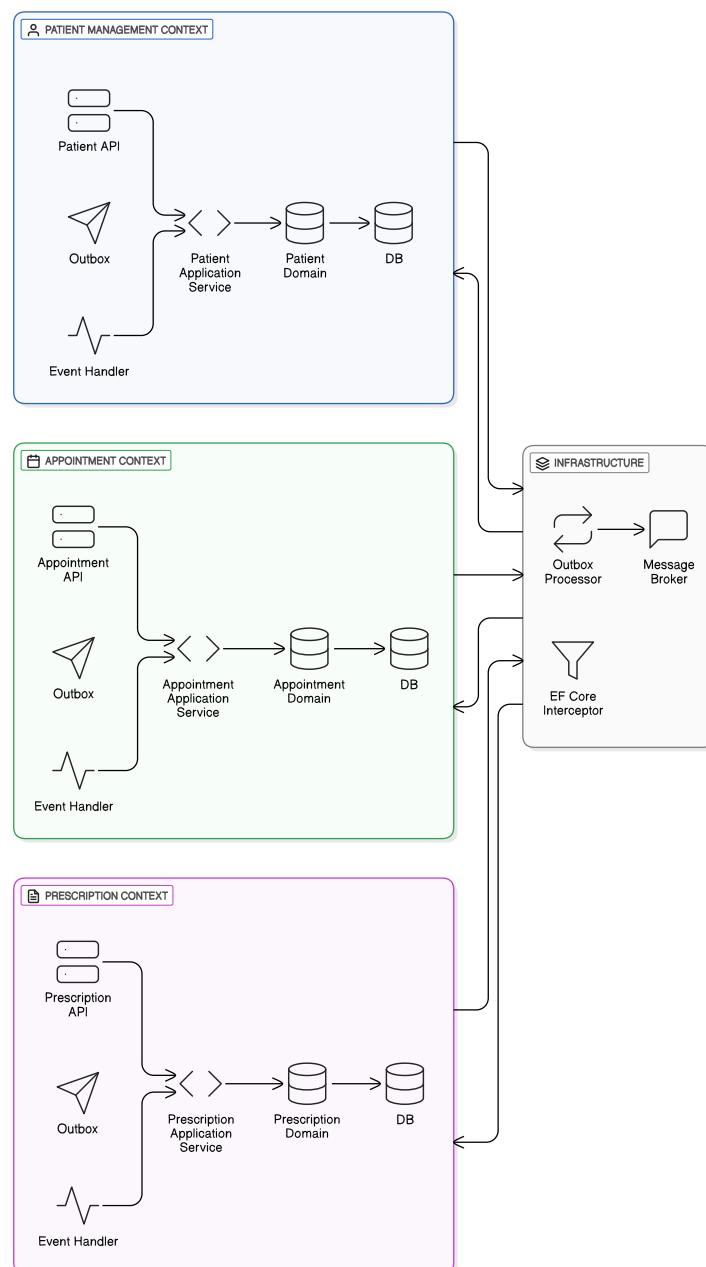


The sequence shows the "Issue Prescription" process. Note that once the prescription is saved to the database, a `PrescriptionCreated` event is generated. This event is what the **Notification Module** listens for to send a copy to the patient's mobile app.

### 3. Transactional Outbox Implementation

Reliability is a core requirement of SCMS. To ensure that we never lose a message when communicating between modules (or future microservices), we have implemented the **Transactional Outbox Pattern**.

#### 3.1 The Mechanism



Outbox Pattern is used to ensure reliable communication between bounded contexts in an event-driven architecture. It prevents inconsistencies caused by partial failures. In distributed systems, updating the database and publishing events to a message broker are separate operations. Failures between these operations can result in lost or duplicated events