Movie Ticket Booking System

Welcome to our end-to-end solution for browsing movies and booking tickets.

Our microservices architecture ensures scalability and flexibility for theaters and customers alike.

S **by Saratha Natarajan**

Overview of the Movie Ticket Booking System

Online Platform

Seamless experience for browsing, selecting, and booking tickets from any device.

Core Entities

Movie, Theater, Customer, and Booking form the foundation of our system.

Modern Architecture

Microservices approach enables robust performance and easy scaling.

Core Entities Explained

Movie

Comprehensive details for all films in the system.

Theater

Information about cinema locations and available screens.

Customer

Registered users who purchase movie tickets.

Booking

Records of all ticket purchases with essential details.

Entity: Movie

Key Attributes

- Title and unique ID
- Genre and duration
- Rating and language
- Cast members
- Synopsis

Functionality

Supports both current releases and upcoming movies in the system.

Each movie has a unique identifier for accurate reference throughout the platform.

Entity: Theater

Location

Geographic position with address details for customer navigation.

Screens

Number and types of screens available at each location.

Facilities

Amenities like concessions, parking, and accessibility features.

Identity

Unique Theater ID for system reference and management.

Entity: Customer

Profile Information

Name, email, phone, and personal preferences stored securely.

Booking History

Complete record of past purchases and preferred genres.

Loyalty Status

Rewards program participation and accumulated benefits.

Multiple Bookings

One customer can create many bookings over time.

Entity: Booking

Booking ID

Unique identifier for each transaction

Showtime & Seats

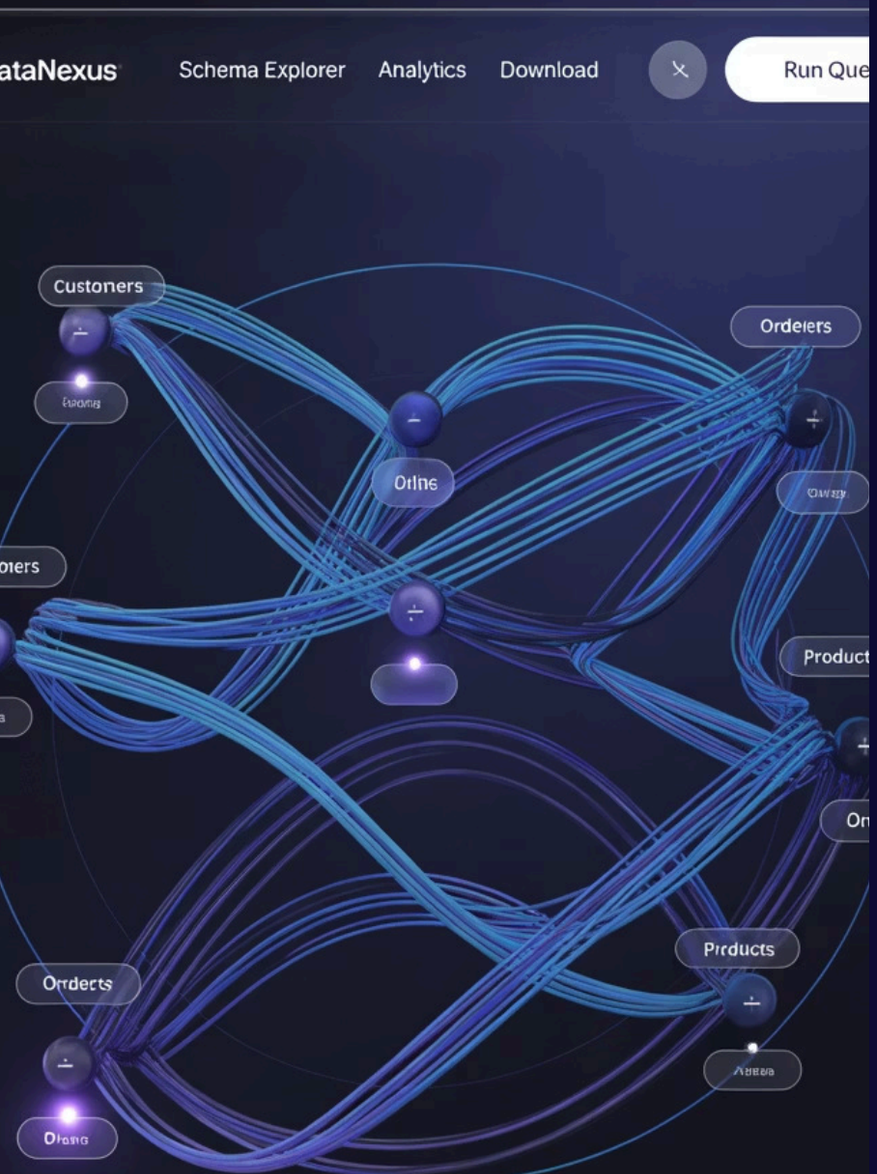
Selected time slot and specific seats

Price & Payment

Cost details and payment status

Entity References

Links to Movie, Theater, and Customer



Entity Relationships Overview

1

Movie ↔ Theater

Many-to-Many relationship

- One movie in many theaters
- Theaters show multiple movies

2

Customer → Booking

One-to-Many relationship

- Customer creates multiple bookings

3

Booking → Entities

Many-to-One references

- Each booking links to one movie
- Each booking links to one theater
- Each booking links to one customer

ER Diagram: Visualizing the Relationships

Movies

Connected to Theaters and Bookings

Theaters

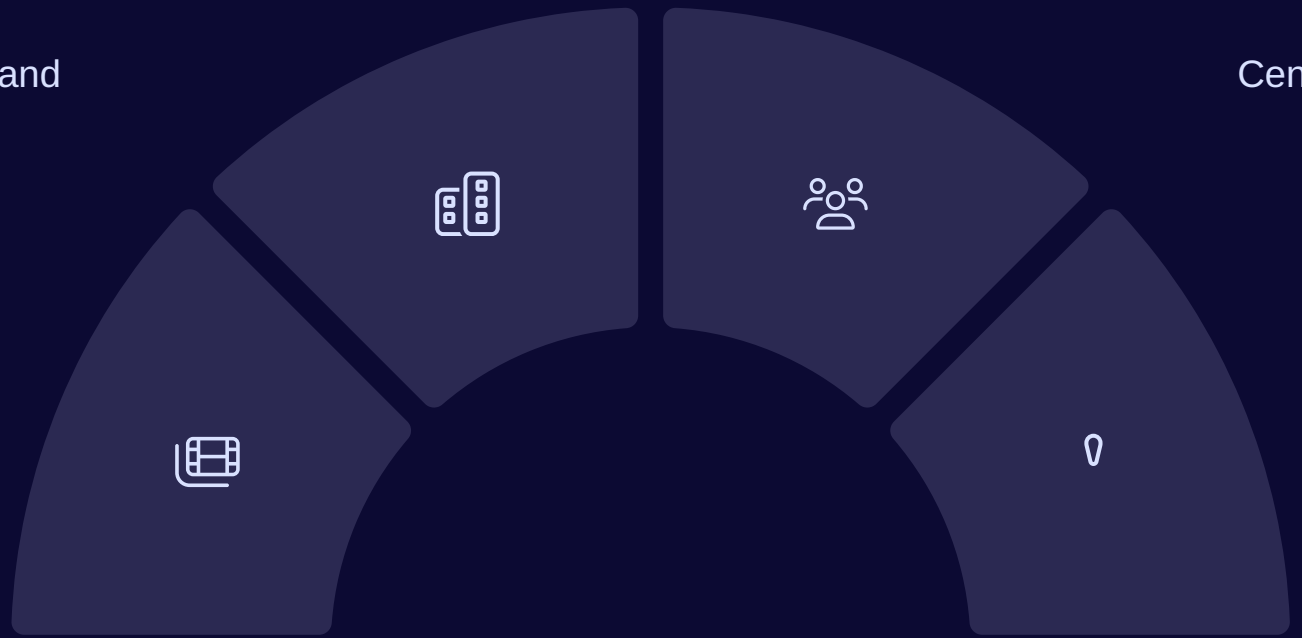
Connected to Movies and Bookings

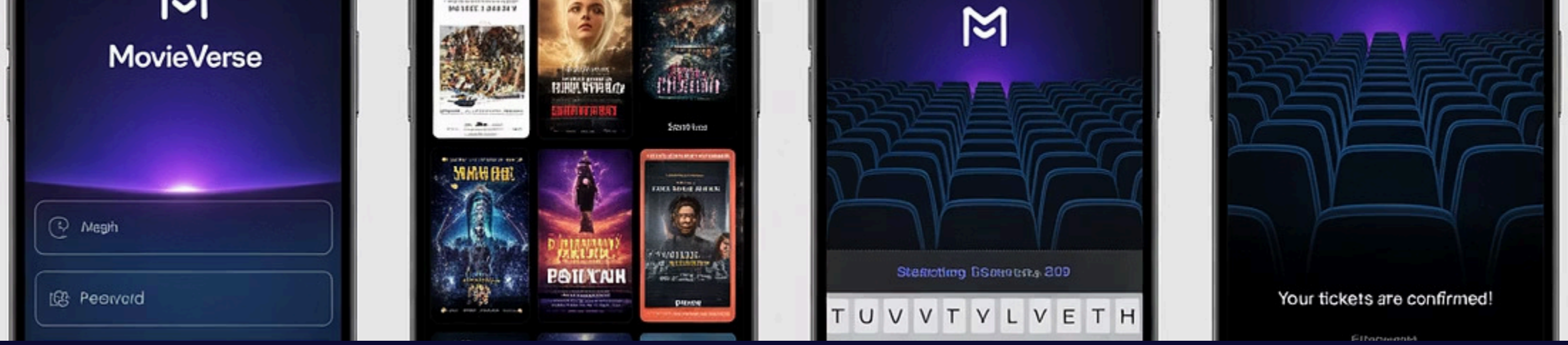
Customers

Connected directly to Bookings

Bookings

Central entity connecting all others





Use Case Flow: Booking a Ticket

Browse & Select

Customer logs in and browses available movies and theaters.

User selects preferred movie and convenient showtime.

Book & Pay

Customer selects available seats for chosen showtime.

System calculates total cost and processes payment.

Confirm & Use

System generates and delivers e-ticket with QR code.

Customer presents e-ticket at theater for entry.

Typical Booking Scenario (Example)

Customer Login

Ana logs into her account

Selection

"Inception" at City Plex, Friday 7pm

Seat Choice

Selects 3 premium seats

Payment

Pays $42 total

Confirmation

Receives QR code e-ticket

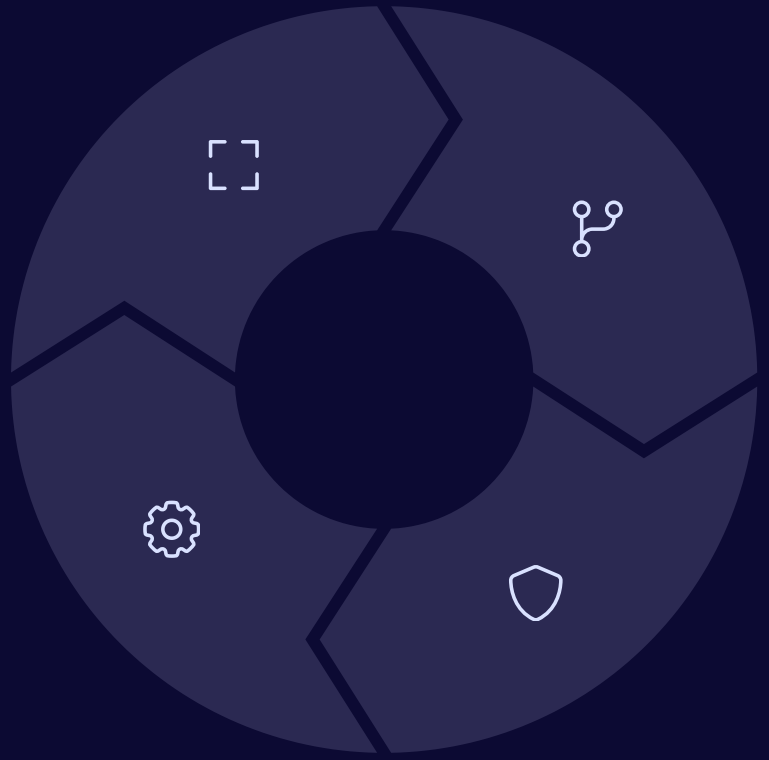
Microservices Architecture: Why and How

Scalability

Services scale independently based on demand

Maintainability

Smaller codebases are easier to maintain



Independence

Teams deploy and update services separately

Resilience

Failures isolated to specific services

Booking as a Microservice: Key Features

4+

API Endpoints

Create, update, cancel, and view booking operations

3+

Integrations

Payment, Notification, and Inventory services

0

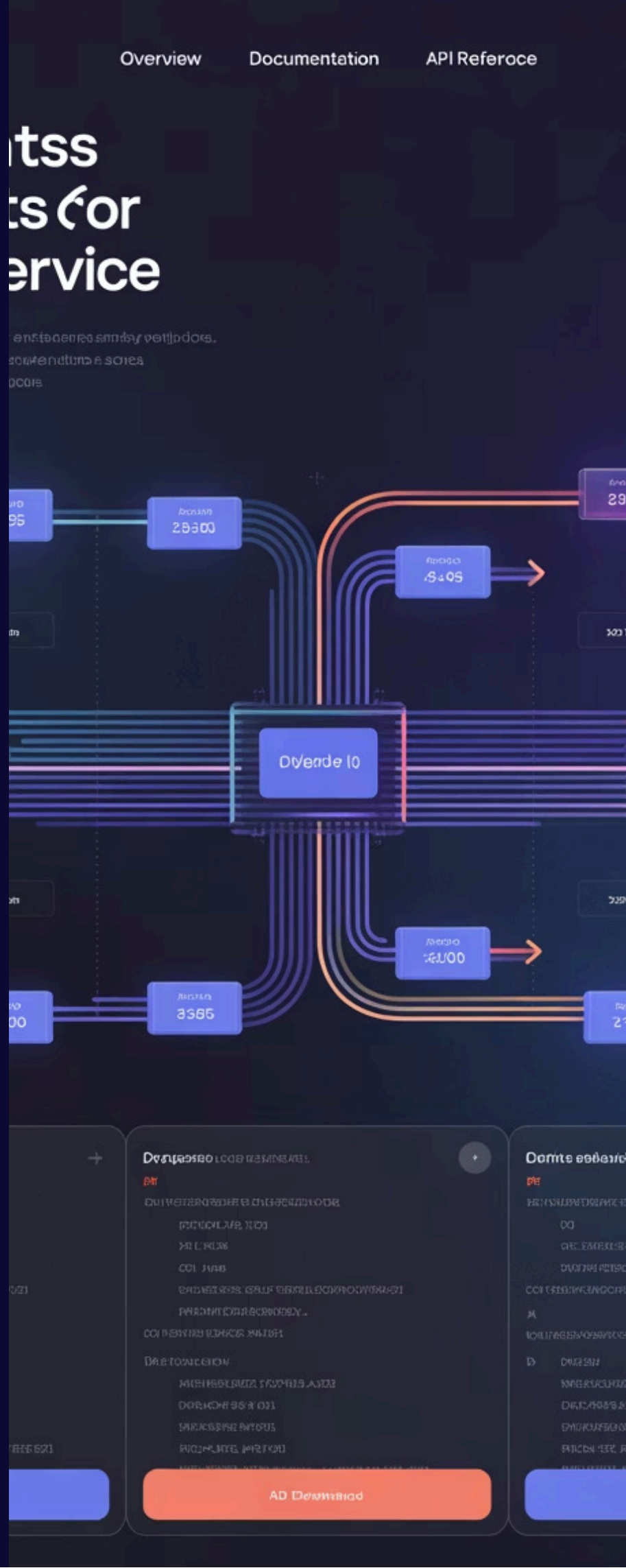
State

Stateless design for horizontal scaling

24/7

Availability

Continuous operation using message queues

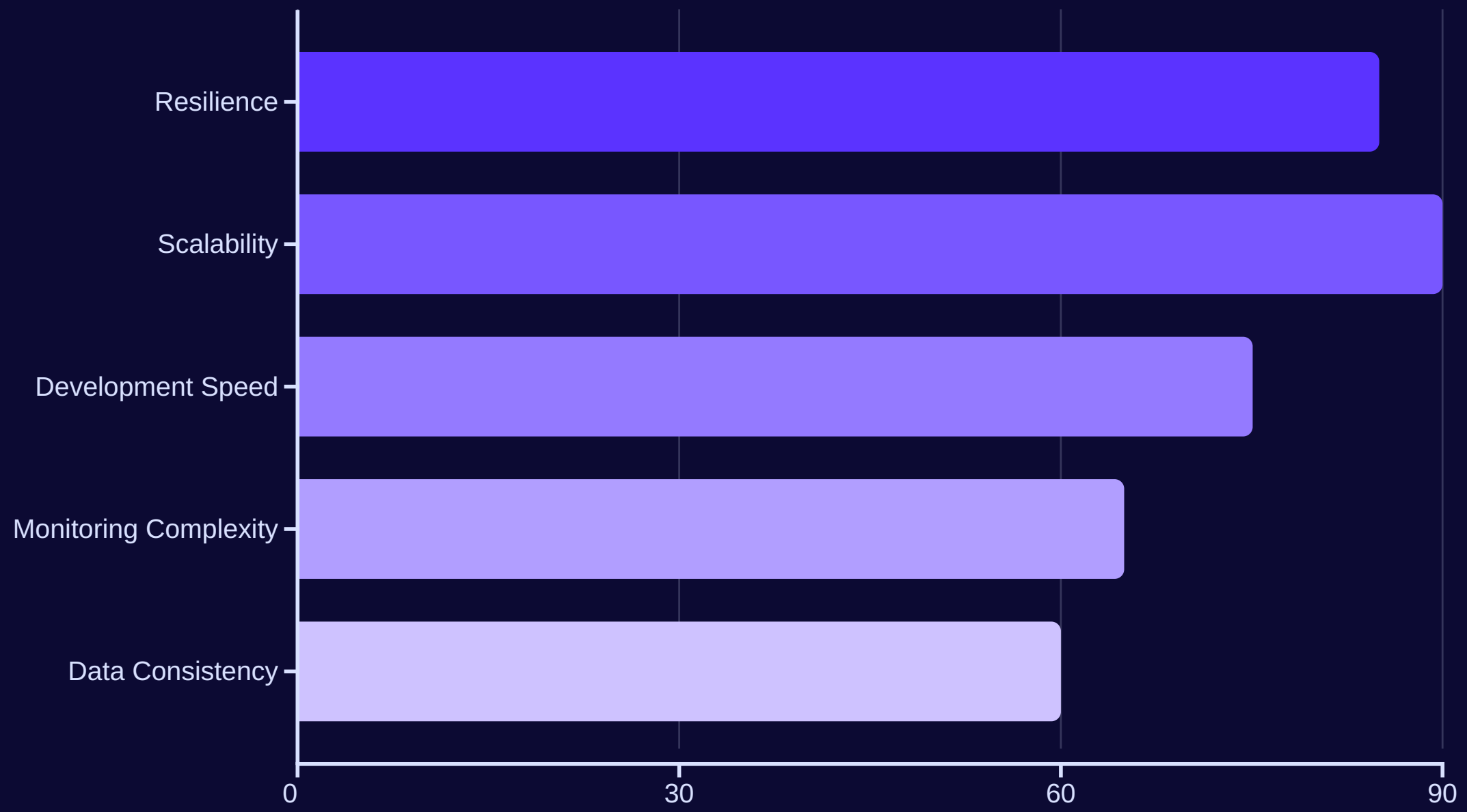


Data Consistency and Transaction Handling

Our system uses distributed transactions to manage seat locks, payments, and confirmations.

The saga pattern handles multi-step workflows with compensating actions for failures.

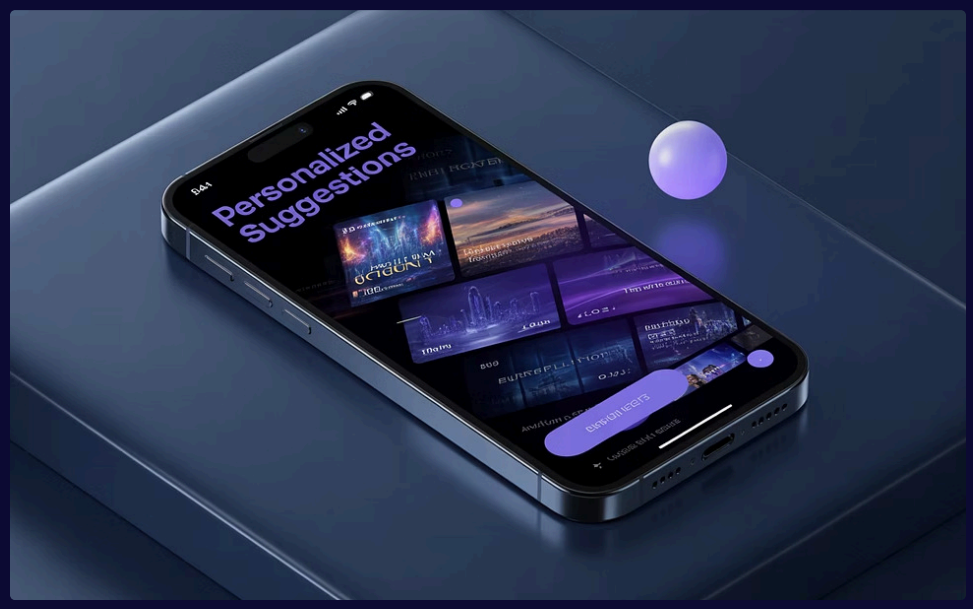
Benefits & Challenges of Microservices in Booking



While microservices excel in resilience and scalability, they present challenges in monitoring and data consistency.

Containerization and orchestration with Kubernetes help manage these complexities.

Conclusion: Future-Ready Ticketing Systems



Innovation

Microservices enable rapid feature deployment and continuous innovation.

Availability

High-availability architecture ensures uninterrupted service for users.

Integration

Ready for smart recommendations, analytics, and partner integrations.