

Solution Document for Online Ticket Booking Platform

Prepared by: Prijesh M

Table of Contents

Solution Document for Online Ticket Booking Platform.....	1
Table of Contents.....	2
Context Diagram.....	4
Overview.....	4
Context Diagram.....	4
Actors.....	5
System.....	5
External Services.....	5
Relationships.....	5
Containers Diagram.....	6
Overview.....	6
Containers Diagram.....	6
Containers.....	7
Relationships.....	7
Major Containers.....	8
Theater Service.....	8
Key Components.....	10
Interactions and Relationships.....	10
Theater Partner Interaction.....	10
Management of Theaters.....	10
User Session and Preferences.....	10
Event-Driven Synchronization.....	11
Movie Catalog Service.....	12
Key Components.....	12
Movie Catalog Service (Node.js).....	12
Movie Database (MongoDB).....	13
External Services.....	13
Interactions.....	13
End Customers.....	13
Administrators.....	13
Theater Partners.....	13
External Services.....	13
Booking Service.....	14
Key Components.....	14
Booking Service (Node.js).....	14
Booking Database (PostgreSQL).....	14
Payment Service.....	15
Notification Service.....	15
Interactions.....	15

End Customers.....	15
Theater Partners.....	15
Administrator.....	15
User Interface (Web & Mobile).....	16
Key Components.....	17
Web Application.....	17
Mobile Application.....	17
API Gateway.....	17
Interactions.....	17
End Customers.....	17
Theater Partners.....	17
Central Entry Point.....	17
Functional features to implement Good to have - Code Implementation (Read scenario)):	18
1. Anyone of the following read scenarios: (Only Service Implementation needed/ No UI required).....	18
Component Diagram:	
support-files/c4model-diagrams/component_movie_catalog_service.puml.....	19
Data Model: support-files/data-models/movie_catalog_service_datamodel.dm.....	20
Code Path: services/movie-catalog-service.....	21
Technologies Used.....	21
2. Anyone of the following write scenarios: Good to have - Code Implementation (write scenario):.....	22
Repo: https://github.com/prijeshm/bct-assesment	23
Component Diagram:	
support-files/c4model-diagrams\component_booking_service.puml.....	23
Data Model: support-files/data-models/movie_catalog_service_datamodel.dm.....	24
Code Path: services/booking-service.....	25
Technologies Used.....	25

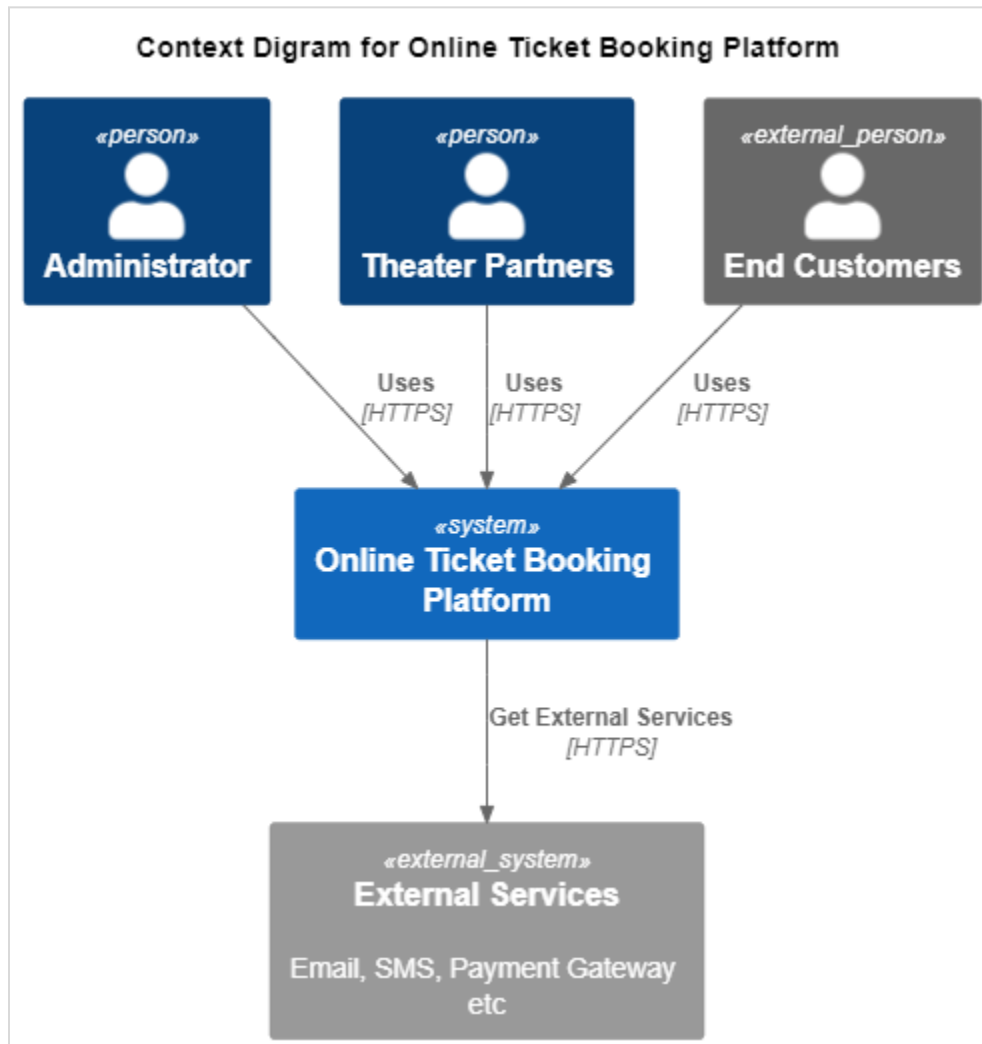
Context Diagram

Overview

The context diagram provides a high-level view of the Online Ticket Booking Platform, showcasing its interactions with key actors and external services. This diagram serves as the foundation for understanding the system's boundaries and its relationships with external entities.

Context Diagram

The context diagram below depicts the core actors and external services connected to the Online Ticket Booking Platform.



Actors

1. **Administrator:** Administrators are responsible for managing and overseeing the platform. They have the authority to configure and maintain the system.
2. **Theater Partners:** Theater partners are entities that collaborate with the platform to onboard theaters, update showtimes, and manage theater-related information.
3. **End Customers:** End customers are the primary users of the platform who browse movies, book tickets, and interact with the system to access movie-related services.

System

Online Ticket Booking Platform (systemTicketBooking): This is the central component of the system, facilitating the entire movie ticket booking process, including user registration, movie catalog management, theater partner interactions, user booking, and payment processing.

External Services

External Services (systemExternal): These services include email, SMS notifications, and payment gateways. They are used by the system to enhance user experiences and provide critical functionality.

Relationships

1. **Administrator:** Administrators use the Online Ticket Booking Platform to configure and manage the system over HTTPS.
2. **Theater Partners:** Theater partners utilize the platform to update showtimes and theater information via HTTPS.
3. **End Customers:** End customers interact with the platform for movie browsing and ticket booking over HTTPS.
4. **Online Ticket Booking Platform:** The platform accesses External Services over HTTPS to provide additional services to end customers and theater partners.

This context diagram offers an initial understanding of the platform's interactions and sets the stage for delving into the system's internal components and interactions.

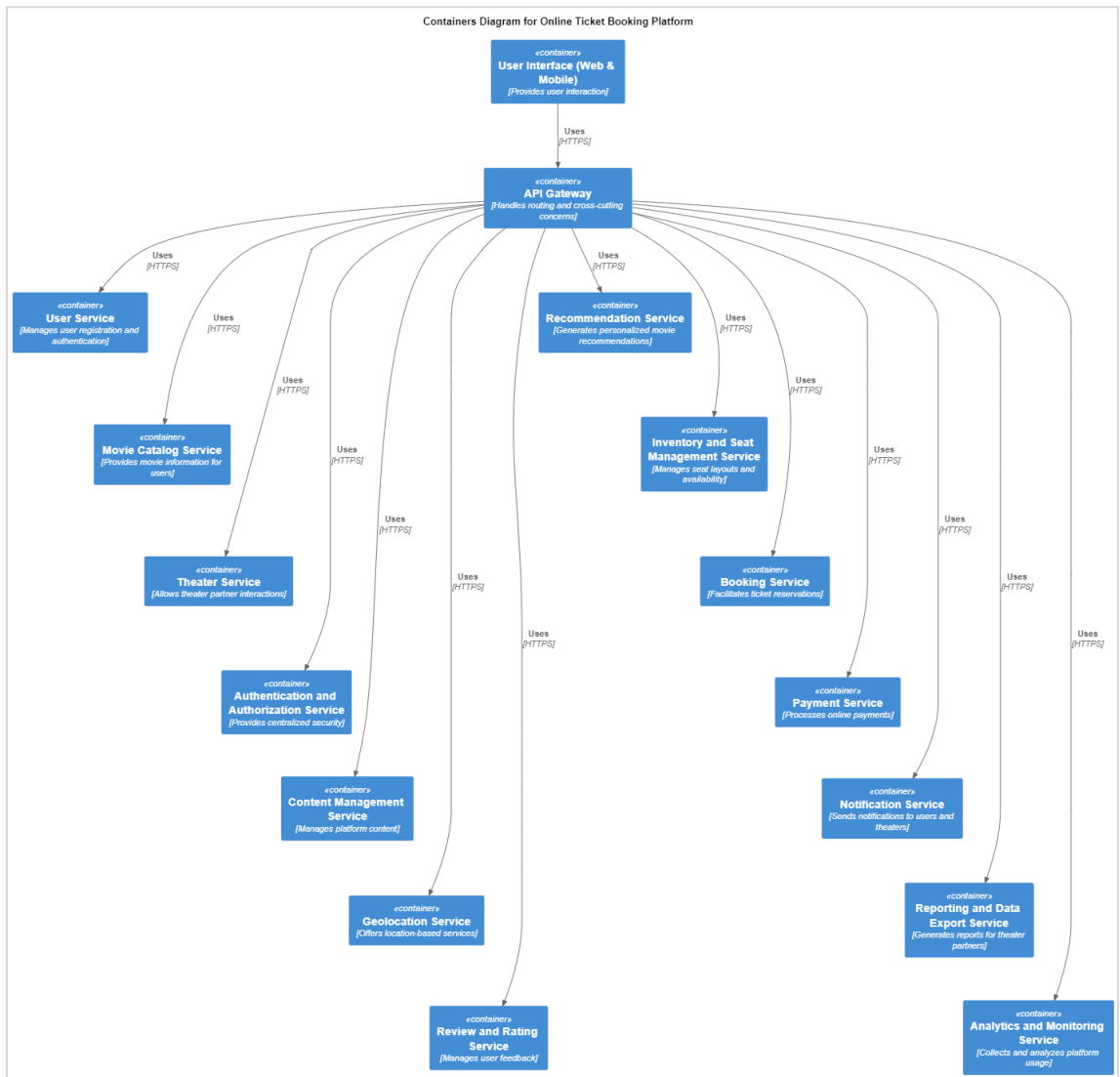
Containers Diagram

Overview

The Containers Diagram provides a more detailed view of the Online Ticket Booking Platform by breaking down the system into containers and depicting how they interact. These containers represent major components or services within the platform, and this diagram helps us understand their relationships and dependencies.

Containers Diagram

The Containers Diagram below illustrates the major components and services that make up the Online Ticket Booking Platform.



Containers

1. User Interface (Web & Mobile): This container represents the front-end components of the platform that provide user interaction through web and mobile interfaces.
2. API Gateway: The API Gateway container handles request routing and manages cross-cutting concerns such as authentication, logging, and monitoring.
3. User Service: Responsible for user registration and authentication, ensuring secure access to the platform.
4. Movie Catalog Service: Manages the catalog of movies, providing details to users.
5. Theater Service: Allows theater partners to onboard theaters and manage showtimes.
6. Authentication and Authorization Service: Centralized security service responsible for user access control and token management.
7. Content Management Service: Manages platform content, including images and promotional materials.
8. Geolocation Service: Offers location-based services to help users find nearby theaters and showtimes.
9. Review and Rating Service: Collects user reviews and ratings for movies, calculating and displaying average ratings.
10. Recommendation Service: Provides personalized movie recommendations based on user preferences.
11. Inventory and Seat Management Service: Manages seat layouts and real-time seat availability information.
12. Booking Service: Facilitates the booking of movie tickets, including seat reservations.
13. Payment Service: Processes online payments securely and integrates with various payment gateways.
14. Notification Service: Sends notifications to users and theaters, including booking confirmations and reminders.
15. Reporting and Data Export Service: Generates reports for theater partners, summarizing their performance.
16. Analytics and Monitoring Service: Collects and analyzes usage data, providing insights for system optimization and decision-making.

Relationships

The diagram illustrates how the various containers interact with one another through the API Gateway. Each container is responsible for specific functionality, and these relationships collectively enable the seamless operation of the Online Ticket Booking Platform.

This Containers Diagram offers an in-depth understanding of the platform's internal components and their interactions, paving the way for further exploration into the components' detailed implementations.

Major Containers

To achieve the key goals of enabling theater partners to onboard their theaters and providing a seamless experience for end customers, we have to consider the below containers in detail with component diagrams.

1. Theater Service:

This container is crucial for enabling theater partners to onboard their theaters. It should include components related to theater partner management, theater information, showtime scheduling, and any other features necessary to support theater operations.

2. Movie Catalog Service:

This container is essential for end customers to browse movies. It should contain components responsible for managing movie information, including titles, descriptions, genres, languages, cast, and crew details.

3. Booking Service:

To allow end customers to book tickets in advance, you'll need to explain the components within the Booking Service. This should include components for booking management, seat reservations, and booking confirmations.

4. User Interface (Web & Mobile):

The user interface container is critical for providing a seamless experience to end customers. Explain the components that make up the user interface, including those responsible for user interactions, movie browsing, ticket booking, and a user-friendly experience.

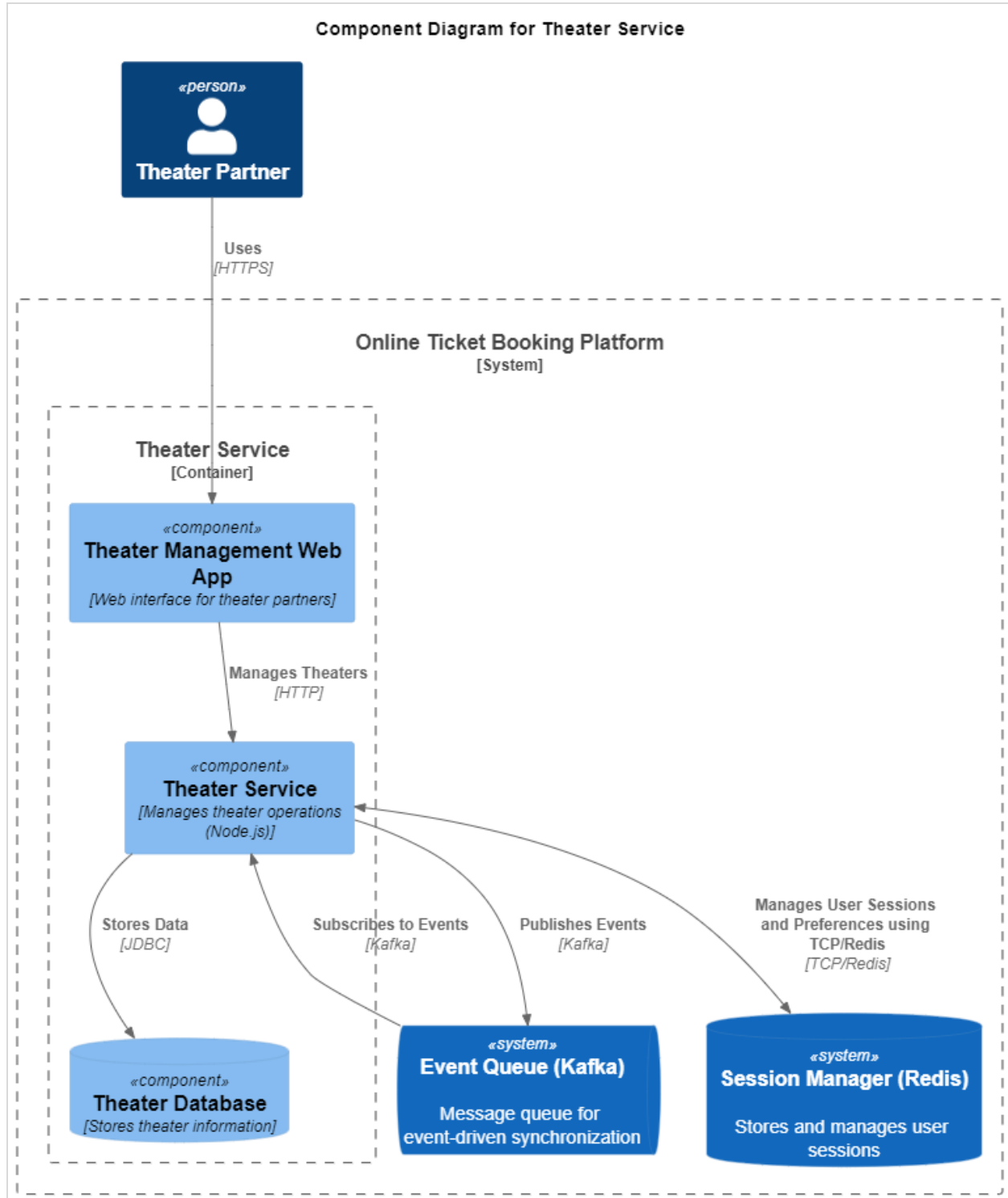
5. API Gateway (If used for external interactions):

If you have an API Gateway, you should detail the components related to routing requests and handling cross-cutting concerns, such as authentication and security.

These containers play a significant role in achieving the specified goals. By explaining their components in detail with component diagrams, can provide a comprehensive understanding of how the platform enables theater partners to onboard theaters and delivers a seamless experience to end customers. Component diagrams will help illustrate the internal structure and interactions of each container, allowing stakeholders to visualize the system's architecture more effectively.

Theater Service

The Theater Service plays a pivotal role in the Online Ticket Booking Platform. It manages the core operations related to theaters, ensuring that theater partners and end customers have a seamless and delightful experience. The Component Diagram below provides an architectural overview of the Theater Service within the broader system.



Key Components

Theater Service (Node.js): At the heart of the Theater Service is a Node.js-based component responsible for the management of theater operations. This component handles various tasks, including theater partner interactions, movie scheduling, and seat availability.

Theater Database: The Theater Database stores crucial information about theaters, enabling easy access to theater-related data. This database is vital for managing theater listings, location information, showtimes, and seat availability.

Theater Management Web App: The Theater Management Web App is the web interface through which theater partners interact with the platform. It empowers theater partners to manage their theater listings, update showtimes, and access reports.

Session Manager (Redis): Redis serves as the dedicated session manager, responsible for storing and managing user sessions and preferences. This ensures that user interactions are personalized and seamless across the platform. It's a critical component for maintaining stateful user experiences, including seat selections and user preferences.

Event Queue (Kafka): The Event Queue, powered by Kafka, facilitates real-time, event-driven synchronization between microservices. It allows for the seamless exchange of information between components, ensuring that data is consistently up to date.

Interactions and Relationships

Theater Partner Interaction

The Theater Management Web App is the primary channel through which theater partners interact with the platform. They can manage theater listings, view bookings, and access essential reports. User interactions with this web app occur over HTTPS.

Management of Theaters

The Theater Service communicates with the Theater Database via JDBC to store and retrieve data related to theaters, showtimes, and seat availability. It enables real-time updates and access to accurate theater information.

User Session and Preferences

The Session Manager (Redis), configured for efficient TCP/Redis communication, plays a crucial role in managing user sessions and preferences. It ensures that user interactions are personalized, and their preferences are stored and maintained.

Event-Driven Synchronization

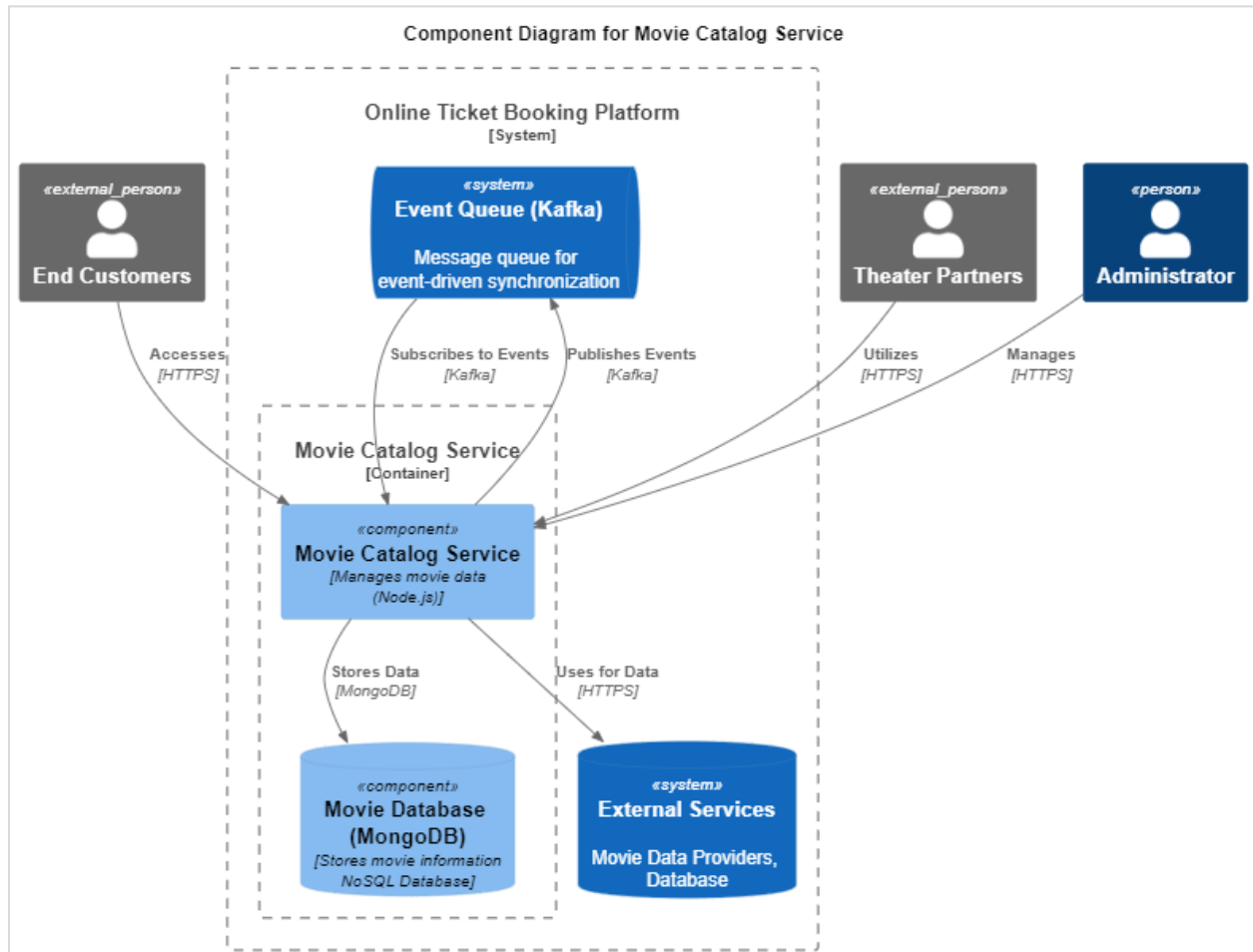
Kafka's Event Queue facilitates event-driven synchronization among components. The Theater Service publishes events related to theaters, showtimes, and bookings, which other microservices subscribe to. This real-time communication is vital for ensuring that all parts of the platform are informed and up to date.

The Component Diagram showcases the relationship between components and their interactions, highlighting how each element contributes to the seamless operation of the Theater Service within the Online Ticket Booking Platform. It illustrates how the service manages theaters, user interactions, and data synchronization, ensuring a delightful experience for theater partners and end customers.

This Component Diagram serves as a fundamental reference for understanding the architecture of the Theater Service and its role in delivering a robust, user-friendly, and efficient online ticket booking platform.

Movie Catalog Service

The Movie Catalog Service is a cornerstone of the Online Ticket Booking Platform, enriching the user experience by providing a comprehensive catalog of movies across diverse genres, languages, and cities. The Component Diagram below presents a detailed architectural overview of the Movie Catalog Service.



Key Components

Movie Catalog Service (Node.js)

At the core of the Movie Catalog Service is a Node.js-based component dedicated to managing and presenting movie data. This component ensures that movie information is always up to date, empowering end customers, administrators, and theater partners with the latest and most accurate data.

Movie Database (MongoDB)

The Movie Database, built on MongoDB, acts as the repository for all movie-related information. This NoSQL database excels in managing unstructured data, making it the ideal choice for storing and querying movie details. It provides the necessary flexibility to handle diverse data points, such as movie descriptions, cast, genres, and more.

External Services

External Services encompass a network of data providers and databases that feed the Movie Catalog Service with a continuous stream of movie data. These services play a pivotal role in enriching the catalog and ensuring that users have access to an extensive and updated movie selection.

Interactions

End Customers

End customers actively engage with the Movie Catalog Service to explore movies, access showtimes, and make informed decisions about their entertainment choices. They experience a rich and user-friendly interface that enables seamless interaction.

Administrators

Administrators leverage the Movie Catalog Service to curate and manage the movie catalog. They can update movie listings, make scheduling decisions, and ensure the platform's movie information remains current and accurate.

Theater Partners

Theater partners utilize the Movie Catalog Service to enhance their theater listings, make informed scheduling decisions, and offer an extensive array of movie choices to their customers.

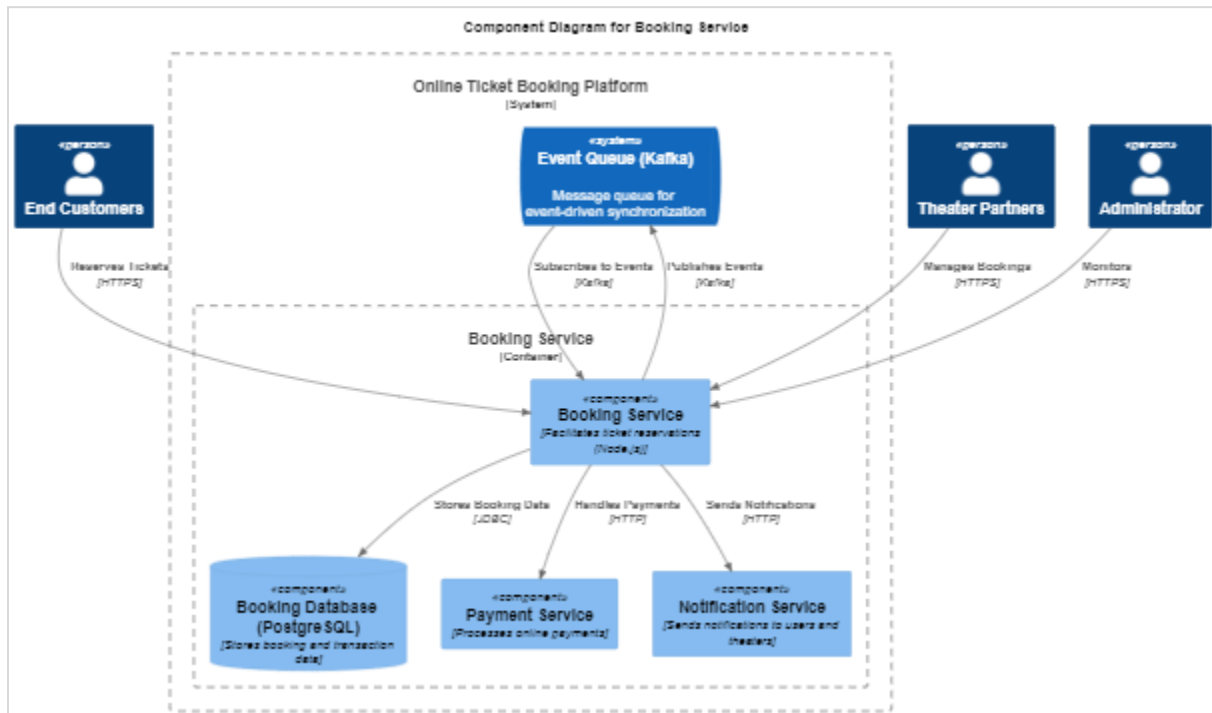
External Services

The Movie Catalog Service relies on external data providers and databases to continuously enrich its catalog. These services contribute to the platform's extensive movie selection.

The Component Diagram illuminates the intricate relationships and dependencies of the Movie Catalog Service within our Online Ticket Booking Platform. It highlights the service's role in providing a diverse and captivating movie catalog to end customers, administrators, and theater partners.

Booking Service

The Booking Service is a pivotal component within the Online Ticket Booking Platform, serving as the backbone for facilitating ticket reservations and managing transactions. This Component Diagram provides an architectural view of the Booking Service and its integration with key platform components.



Key Components

Booking Service (Node.js)

The Booking Service, developed on Node.js, plays a central role in enabling users to reserve tickets effortlessly. This component offers a seamless and user-friendly interface that empowers end customers, administrators, and theater partners to make reservations with ease.

Booking Database (PostgreSQL)

The Booking Database, based on PostgreSQL, serves as the repository for all booking and transaction data. This relational database is designed to ensure data integrity and provide robust data management capabilities.

Payment Service

The Payment Service is responsible for processing online payments securely. It handles financial transactions, ensuring a smooth payment experience for users while maintaining the highest standards of security.

Notification Service

The Notification Service is instrumental in delivering timely notifications to users and theaters. It plays a crucial role in keeping users informed about reservation confirmations, ticket details, and other important updates.

Interactions

End Customers

End customers utilize the Booking Service to reserve tickets for their chosen movies and showtimes. The service offers a streamlined and intuitive booking experience.

Theater Partners

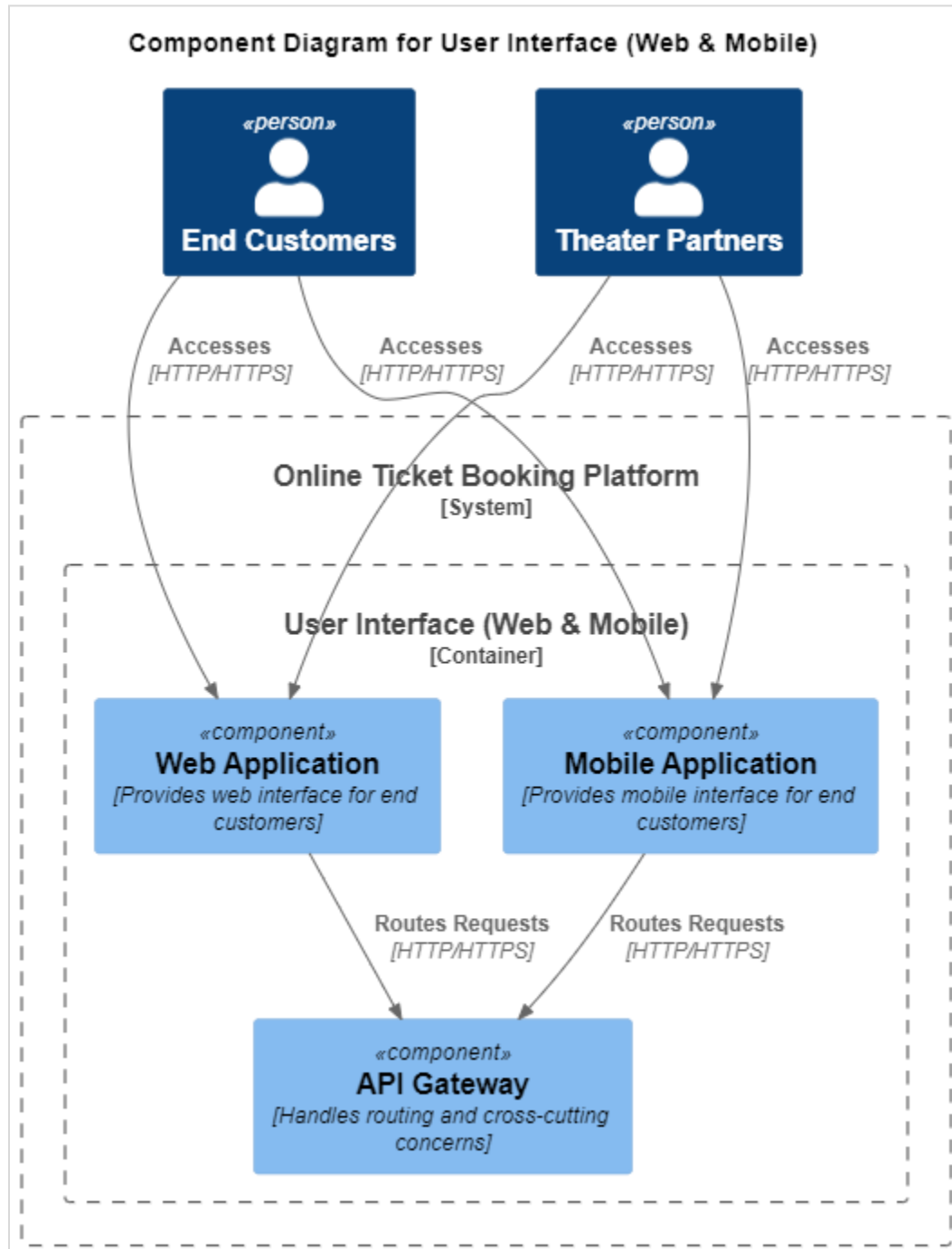
Theater partners depend on the Booking Service to manage bookings, helping them ensure a seamless experience for their patrons.

Administrator

Administrators use the Booking Service for monitoring and managing the booking process, including handling exceptions and resolving issues.

User Interface (Web & Mobile)

The User Interface (Web & Mobile) serves as the gateway to the Online Ticket Booking Platform, providing a user-friendly and accessible interface for both end customers and theater partners. This Component Diagram offers insights into the architectural components that power the web and mobile experiences.



Key Components

Web Application

The Web Application represents the web interface that end customers and theater partners use to access the platform. It offers a versatile and responsive design, making it accessible from a variety of devices.

Mobile Application

The Mobile Application complements the web interface, providing a mobile-friendly platform for end customers and theater partners. It ensures that users can access the platform on the go, enhancing convenience and accessibility.

API Gateway

The API Gateway acts as the central entry point to the platform. It handles routing and cross-cutting concerns, ensuring that requests are directed to the appropriate components. This component simplifies the user interface's interaction with the underlying services.

Interactions

End Customers

End customers engage with both the Web Application and the Mobile Application to browse movies, check showtimes, and book tickets. They experience a consistent and user-friendly interface across web and mobile platforms.

Theater Partners

Theater partners may also use the User Interface (Web & Mobile) to manage their theater listings, monitor bookings, and access important data. They benefit from a responsive and intuitive interface.

Central Entry Point

The API Gateway acts as a central entry point for the User Interface (Web & Mobile), ensuring that requests are routed efficiently to the appropriate components and services. This simplifies the communication between the user interface and the underlying platform.

The Component Diagram provides a clear view of the components that constitute the User Interface (Web & Mobile), emphasizing its role in providing a seamless and user-friendly experience for both end customers and theater partners. It serves as the primary channel through which users interact with the Online Ticket Booking Platform, contributing to a positive and engaging user experience.

Functional features to implement Good to have - Code Implementation (Read scenario)):

1. Anyone of the following read scenarios: (Only Service Implementation needed/ No UI required)

- Browse theaters currently running the show (movie selected) in the town, including show timing by a chosen date
- Booking platform offers in selected cities and theaters
 - 50% discount on the third ticket
 - Tickets booked for the afternoon show get a 20% discount

Here I have considered the first point by creating a **movie catalog microservice** as below:

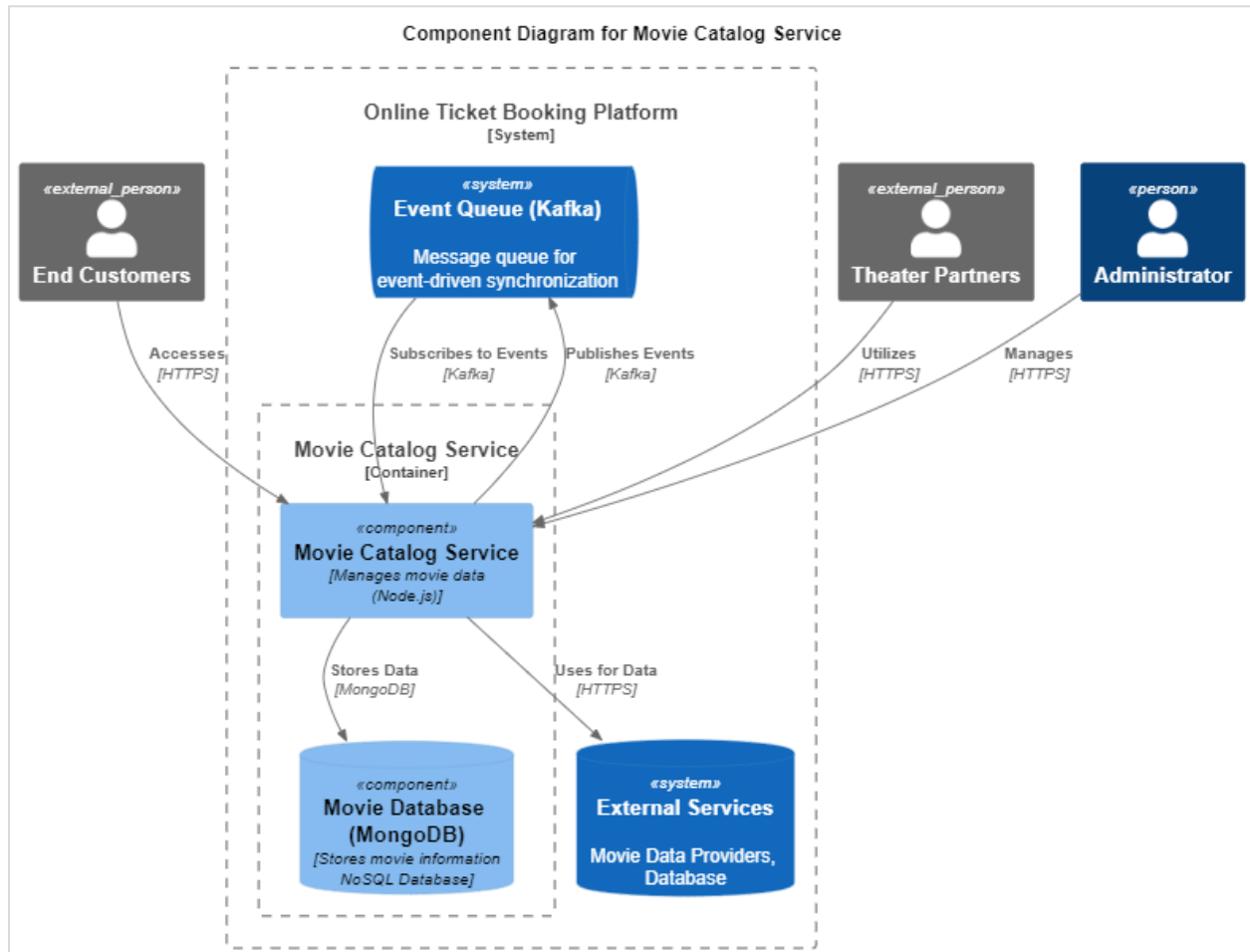
API: /showtimes/:movieId

Method: GET

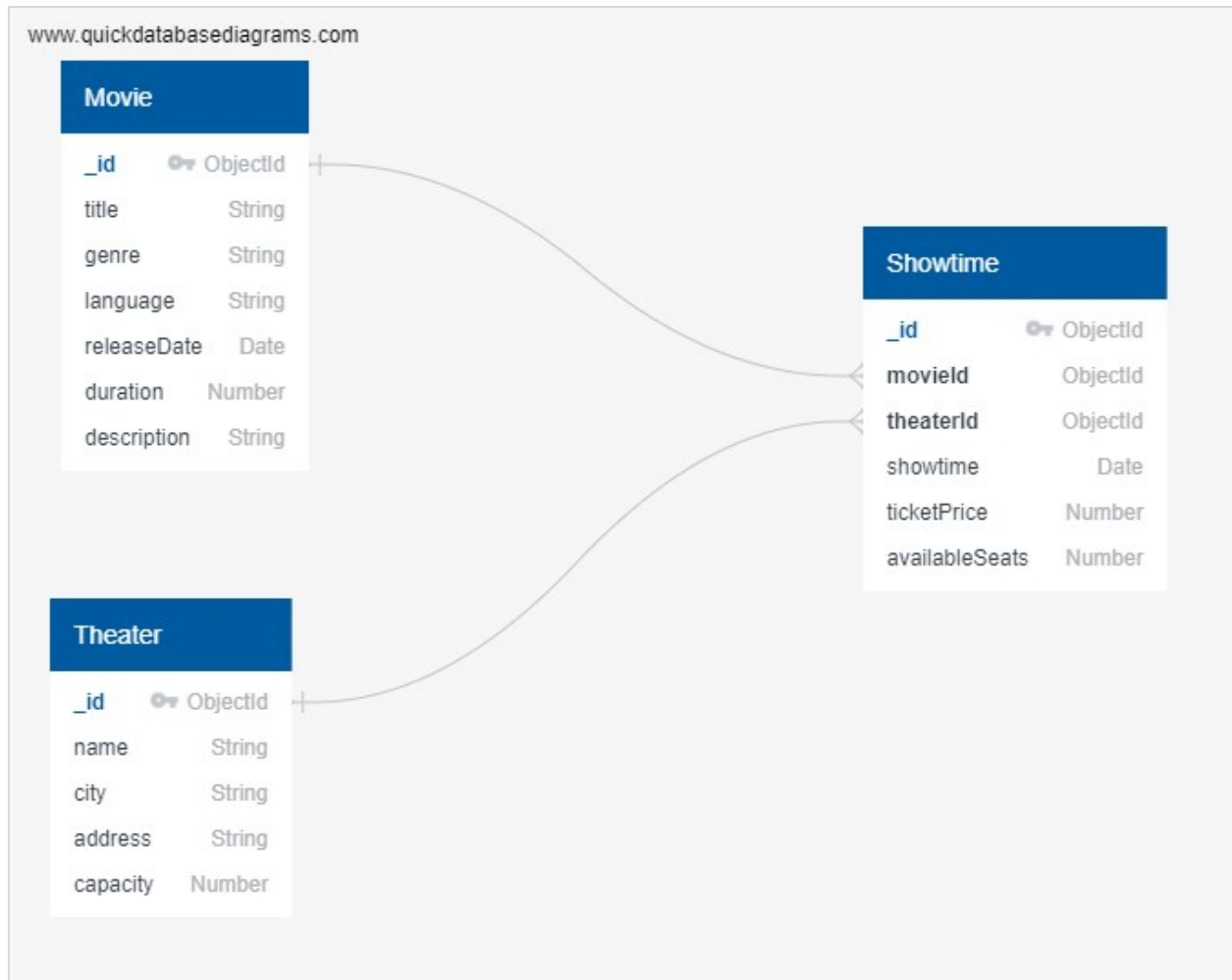
Repo: <https://github.com/prijeshm/bct-assesment>

Component Diagram:

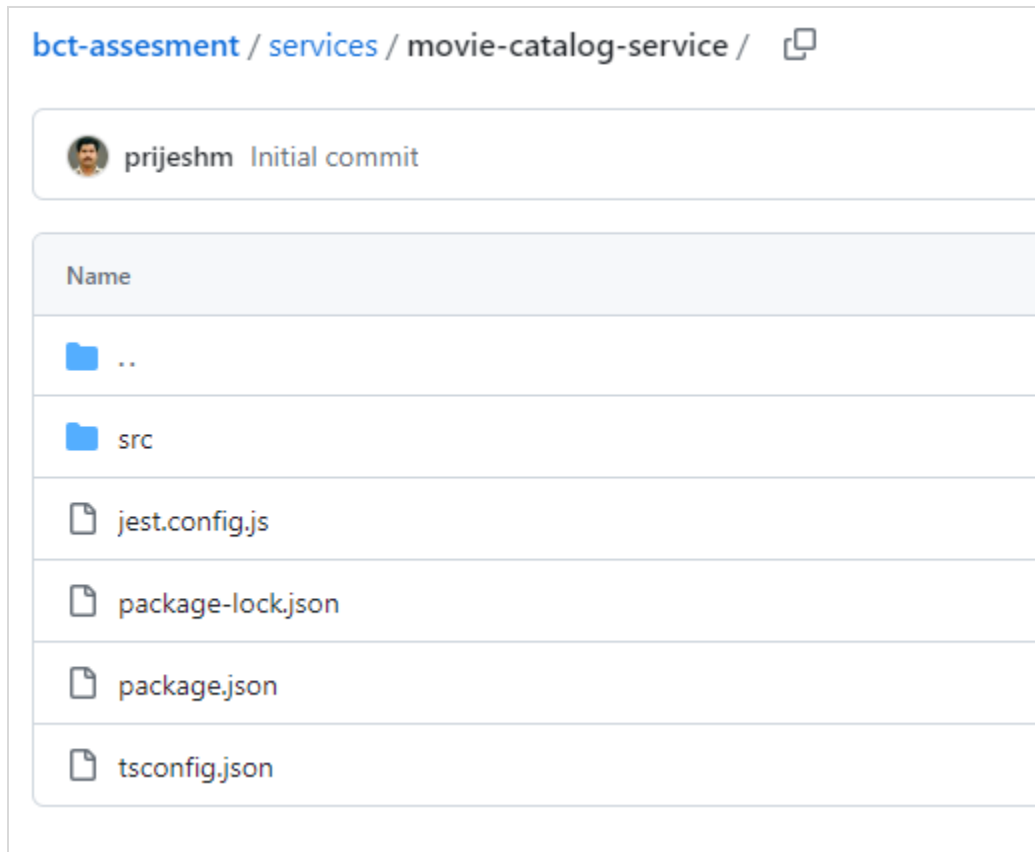
support-files/c4model-diagrams/component_movie_catalog_service.puml



Data Model: support-files/data-models/movie_catalog_service_datamodel.dm



Code Path: services/movie-catalog-service



Technologies Used

1. Language -TypeScript / JavaScript
2. Frameworks - **Express.js**
3. Database - MongoDB
4. Integration technologies- Kafka(not used in the code)
5. Cloud technologies- Any (Dockerized)
6. Preferred editor to build and present solution - VS Code

2. Anyone of the following write scenarios: Good to have - Code Implementation (write scenario):

- Book movie tickets by selecting a theater, timing, and preferred seats for the day
- Theaters can create, update, and delete shows for the day.
- Bulk booking and cancellation
- Theaters can allocate seat inventory and update them for the show

Here I have considered the first point by creating a **booking microservice** as below:

API: /bookings

Method: POST

Body: {

"seats": [3],

"customer": "654a0df0fd88b0c948e0d2cd",

"movie": "654a0e0afd88b0c948e0d2d4",

"theater": "654a0dfbfd88b0c948e0d2d1"

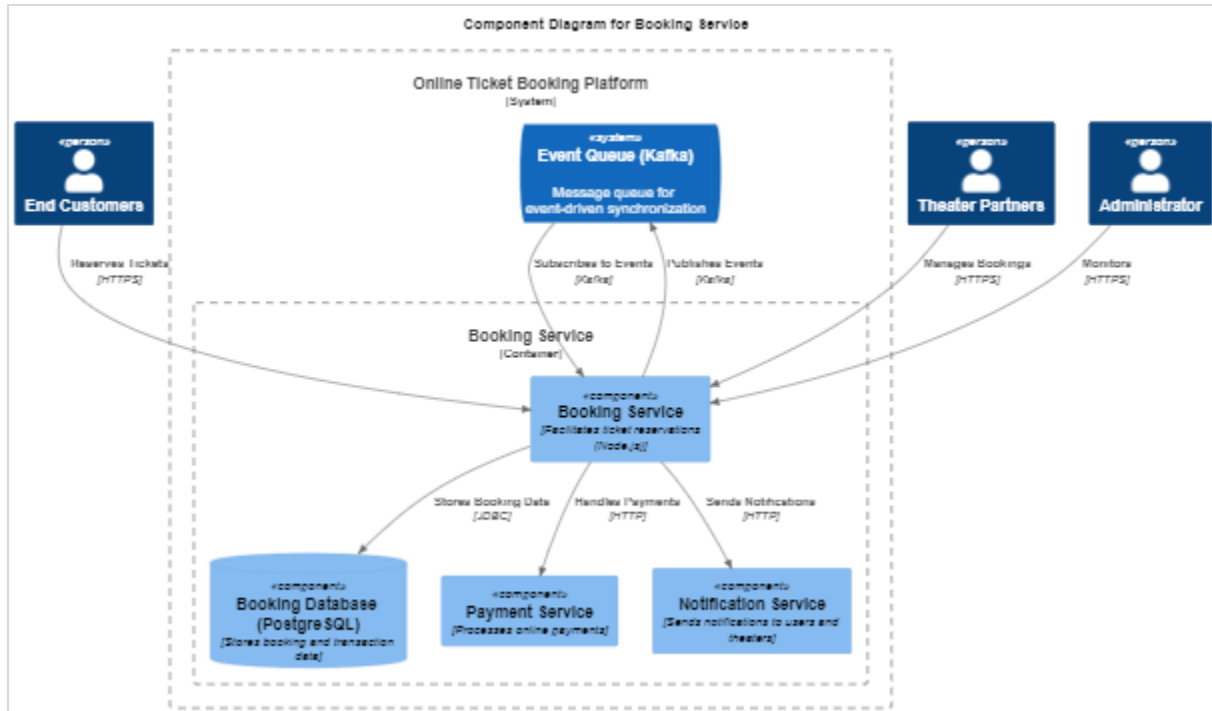
}

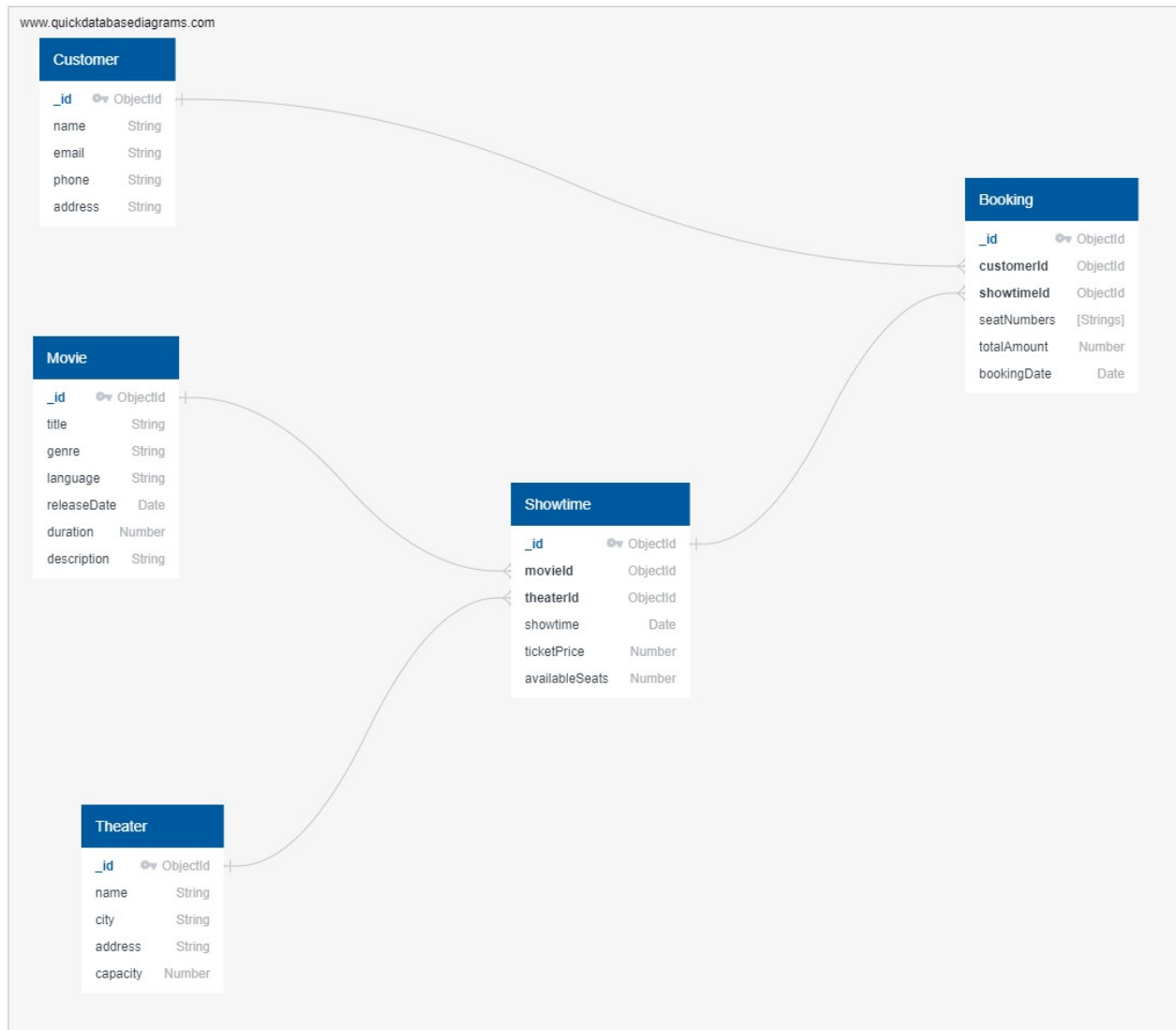
Note: As per the component diagram I have mentioned PostgreSQL as the dedicated microservice database. For coding easiness I have used MongoDB

Repo: <https://github.com/prijeshm/bct-assesment>















Component Diagram:

support-files/c4model-diagrams/component_booking_service.puml



Data Model: support-files/data-models/movie_catalog_service_datamodel.dm

Code Path: services/booking-service

bct-assesment / services / booking-service / 	
	prijeshm Initial commit
Name	
	..
	src
	test
	.eslintrc.js
	.gitignore
	.prettierrc
	README.md
	nest-cli.json
	package-lock.json
	package.json
	tsconfig.build.json
	tsconfig.json

Technologies Used

7. Language -TypeScript / JavaScript
8. Frameworks- **Nest.js**
9. Database - MongoDB
10. Integration technologies- Kafka(not used in the code)
11. Cloud technologies- Any (Dockerized)
12. Preferred editor to build and present solution - VS Code