TicketBoss

Team 35 - André Morais, Miguel Rodrigues, Paul Rodrigues

Logical View

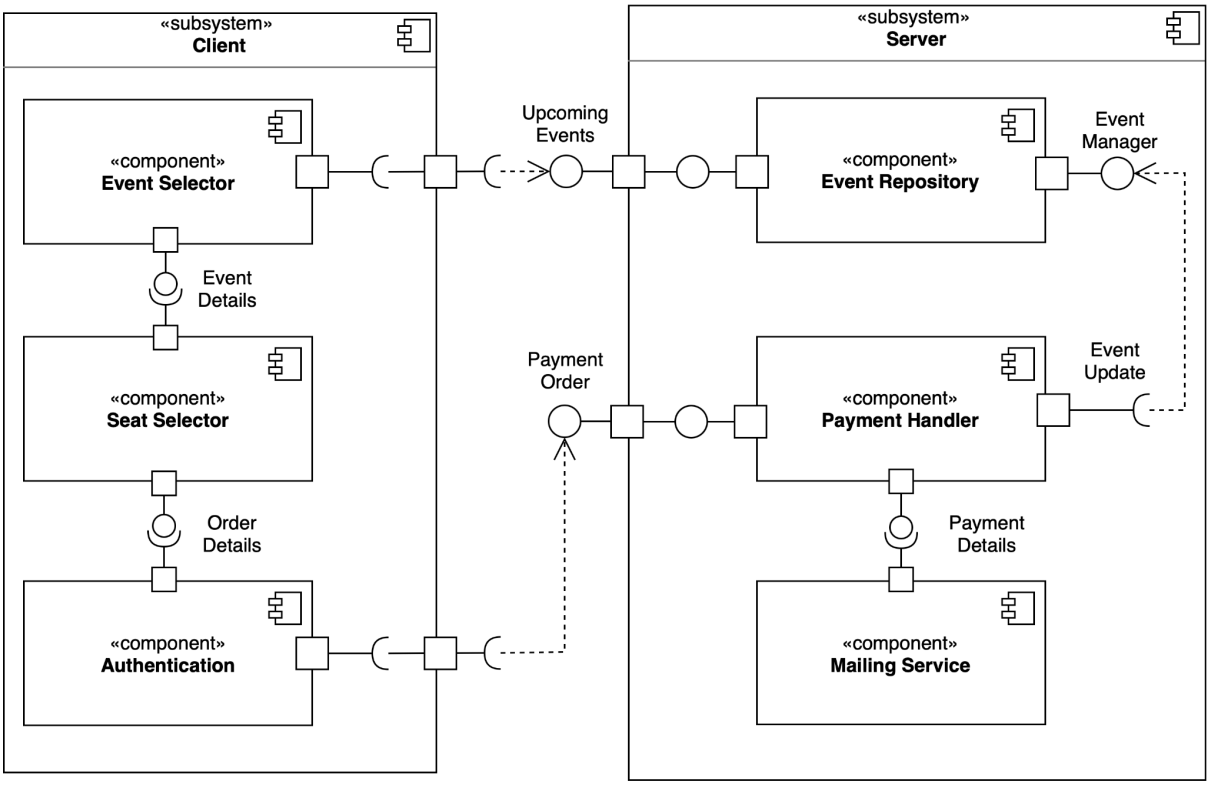

Fig. 1 – Logical view diagram of the TicketBoss system.

Given its simplicity, the TicketBoss system has two subsystems: Client and Server. There are six main components: Event Selector, Seat Selector, Authentication, Event Catalog, Payment Handler, and Email Service. The Event Selector, Seat Selector, and Authentication are part of the Client subsystem, and the others are part of the Server subsystem.
Each component has a specific role in the ticket purchase process:

1. **Event Selector:** This component allows users to browse and select upcoming events. It interacts with the Event Catalog component to retrieve information about available events.
2. **Seat Selector:** Once the user selects an event, this component presents the available seats and allows the user to choose their preferred seats.
3. **Authentication:** Handles user authentication and authorization. It ensures that only users with inserted valid payment information can proceed with the ticket purchase process. It communicates with the Seat Selector component to validate the user's order details.
4. **Event Repository:** Manages the catalog of available events/seats. It provides information about upcoming events to the Event Selector component.

5. **Payment Handler:** Handles payment processing for completed orders. It receives payment orders from the Authentication component and interacts with external payment processing services to complete the transaction. It also communicates with the Email Service component to send payment confirmation emails to users.
6. **Email Service:** Sends confirmation emails to users after successful payment transactions. It receives payment details from the Payment Handler component.

Process View

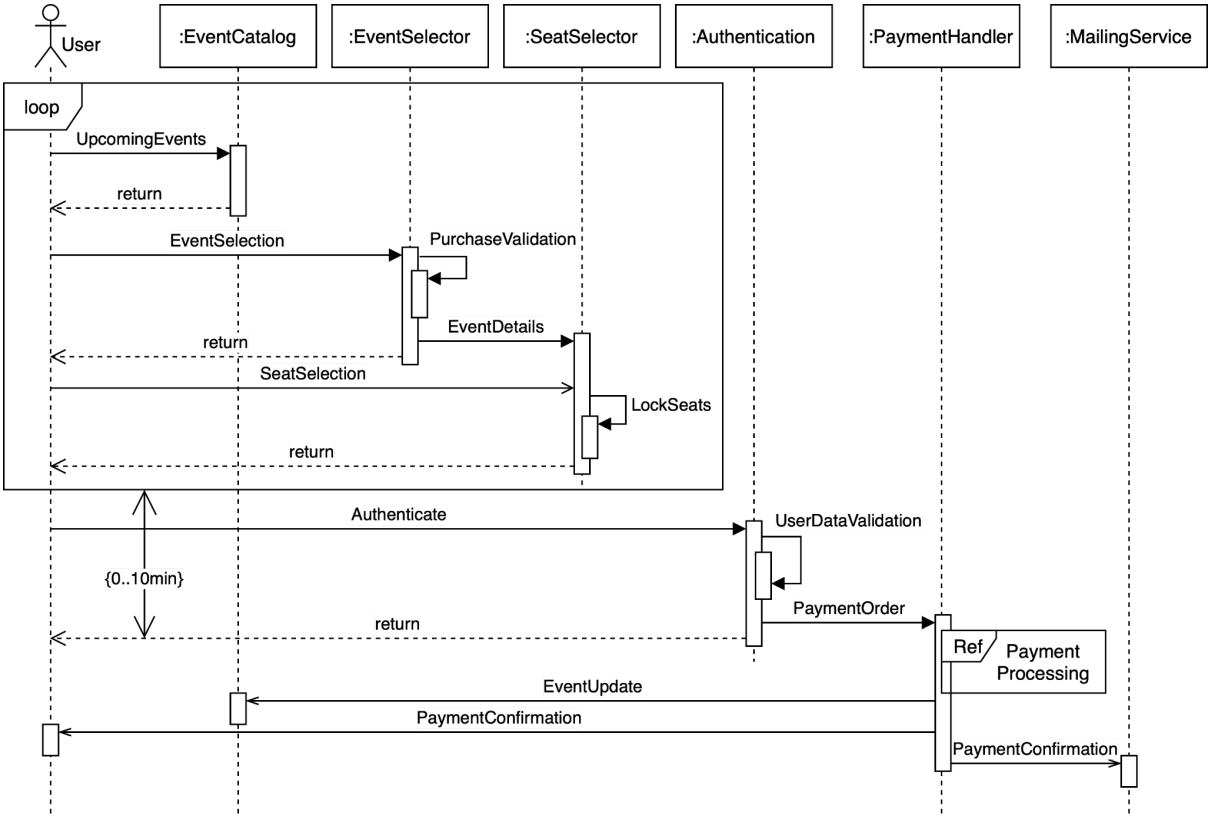

Fig. 2 – Process view diagram containing the ticket purchase flow.

The process view diagram above shows a simplified version of the ticket purchase flow, in a simplified manner without error handling.

The loop block represents the display and selection of different events and their respective seats. First, the user requests the list of upcoming events. Then it selects the desired event and a validation occurs in order to ensure that the user associated with the current session is eligible to purchase for the selected event.

If the validation succeeds, the event details are sent to the seat selector responsible for locking the seats upon selection. The seats get locked for 10 minutes. In the meantime, the user is able to make additional reservations for other events, however the user must pay for the selected seat in such a time frame.

Proceeding to checkout is represented by the Authentication message in the diagram. A payment order is placed after a user data validation. With the payments processed, the user receives a confirmation in its mailbox and a notification, while the event catalog is updated accordingly.

This flow was thought of with UX in mind. It gives the user the ability to secure the seats for their favorite events quickly and only after a few minutes proceed to the payment. We believe that, this way, customer satisfaction is improved.

Use-case View

The use-case view provides a perspective on how users interact with the TicketBoss system to achieve specific goals. Through different use cases, we outline the essential functionalities of the system from the user's standpoint. Each use case represents an interaction between the user and the system, capturing the flow of events, preconditions, postconditions, and the involvement of actors. Here, we describe the principal use cases.

UC1 - User selects seats and buys them

Actor: User, Seat Selector, Payment Handler, Mailing Service.
Description:

- User selects an event.
- User proceeds to select available seats for an event.
- The Seat Selector locks those seats for 10 minutes.
- User inserts valid payment information that is verified by the Payment Handler.
- User completes the purchase and the Mailing Service sends an email with the tickets.

Preconditions: Seats are available for purchase and the user inserts payment information.
Postconditions: Seats are reserved, and the transaction is completed.

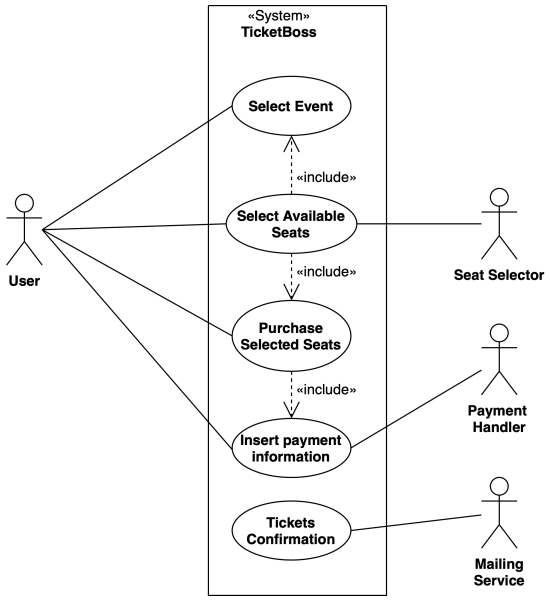

Fig. 3 – Use-Case diagram for UC1.

UC4 - Two or more people select seats for the same event at the same time (within ten minutes of each other.)

Actors: Two or more Users, Seat Selector, Payment Handler, Mailing Service.
Description: Multiple users attffempt to select the same seats for an event but only the first one is able to complete the purchase since the seat selector verifies the availability of the seats before proceeding with the purchase. There are no conflicts.

- Multiple Users select the same event.
- Multiple Users attempt to select the same seats for an event.
- The User proceeds to select available seats for an event.
- The Seat Selector locks those seats for the other Users, during 10 mins, and this way the other Users can't proceed with the purchase of the seats without any conflict occurring.
- User inserts valid payment information that is verified by the Payment Handler.
- User completes the purchase and the Mailing Service sends an email with the tickets.

Preconditions: Seats are available for selection.
Postconditions: First user purchases the seat while other users resolve the conflict by selecting different seats.

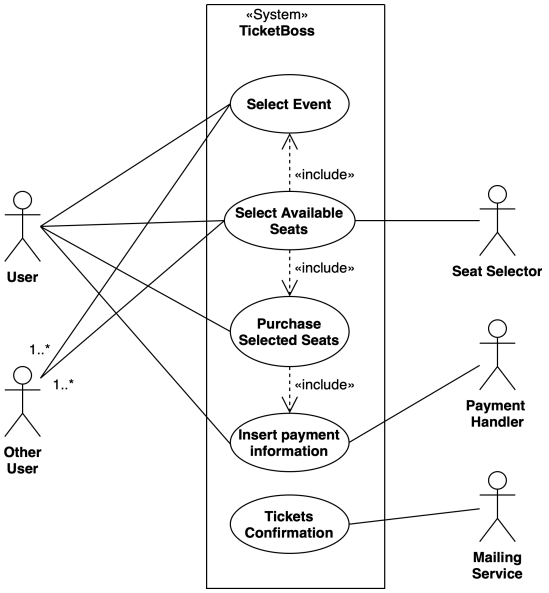

Fig. 4 – Use-Case diagram for UC4.

Physical View

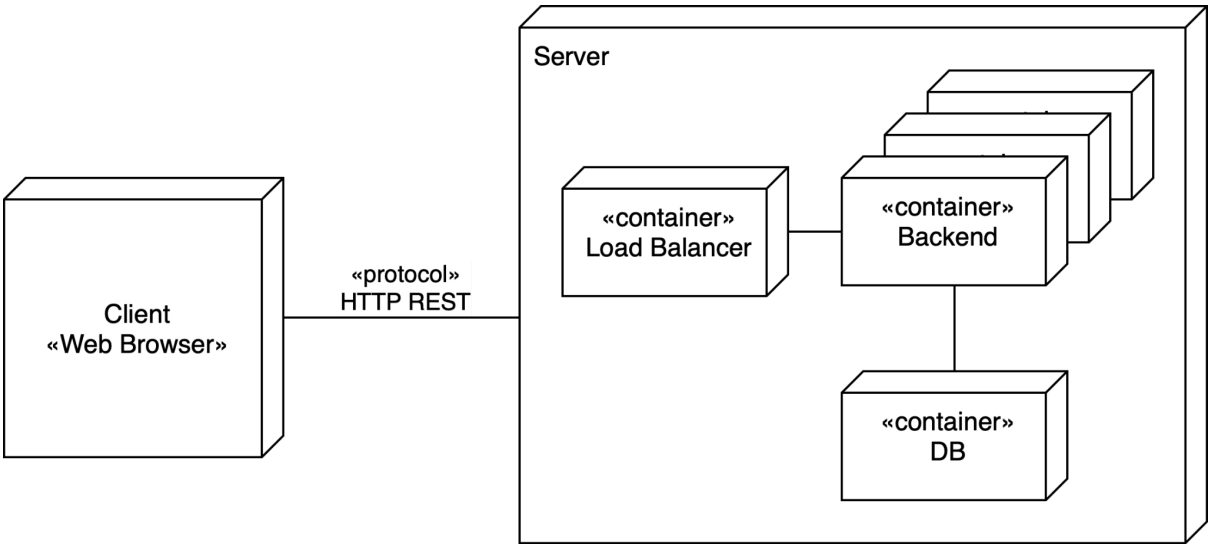

Fig. 5 – Physical view diagram for the TicketBoss system.

The TicketBoss system works under the client-server pattern. The client is deployed as a server application. On the server side, everything is containerized. The incoming requests are handled by a load balancer which distributes the requests among the many backend containers. Such containers communicate with the database, which is also containerized.

Quality Attributes

Quality Attribute	What does it mean?	Acceptance level?	How important?
Security	Data Security, Transaction Security, System Security, User Authentication, Authorization.	Security must not fail at any time as it is of major importance.	Security is the most important quality attribute, and is of paramount importance due to the sensitive nature of the data involved, in particular the user's financial information.
Availability	User Satisfaction, Revenue Generation, Competitive Advantage.	The availability should follow a minimum standard of for example, only 5 minutes of downtime per year.	It should be heavily considered in order to increase user satisfaction needs and in consequence increasing the company sales of

Quality Attribute	What does it mean?	Acceptance level?	How important?
			tickets too.
Reliability	Consistency, Faultness, Recoverability, Fault Tolerance.	Should minimize the occurrence of system failures and errors but a minimum number of errors could still occur without damaging the system too much.	It is important so the system is trustworthy to the clients.
Performance	Efficiency, Throughput, Time Behaviour	Should provide users with the expected fast response times to interact smoothly with the system.	Performance directly impacts user satisfaction and the purchase time should be quick.
Scalability	Flexibility, Expandability, Load Balancing.	The system must seamlessly accommodate a growing user base and transaction volume.	Ensuring scalability is essential for meeting the demands of a potentially large user base.
Usability	Intuitiveness, Ease of Navigation, Clarity on error handling.	Should allow users to navigate through the system smoothly and purchase tickets without any confusion.	It is important to maintain customer satisfaction and avoid leading the users into errors when purchasing tickets.
Portability	Cross-platform compatibility, Interoperability	Depending on the target platforms, some level of portability may be essential.	Portability becomes more critical if the system needs to run on various platforms.
Maintainability	Modularity, Customizability	The system must have some long-term viability and be easy to maintain.	Maintaining the system efficiently reduces downtime, minimizes technical issues, and supports future upgrades or updates.
Extensibility	Adaptability, Modularity, Customizability	While important for long-term adaptability, flexibility and extensibility may not	Building a flexible and extensible architecture enables the system to evolve if needed.

Quality Attribute	What does it mean?	Acceptance level?	How important?
		be immediate user requirements.	
Concurrency	Multi-user support, Concurrent request handling	The system must support concurrent access to events to serve multiple users efficiently.	Handling concurrency effectively ensures a smooth user experience and prevents conflicts during seat selection.