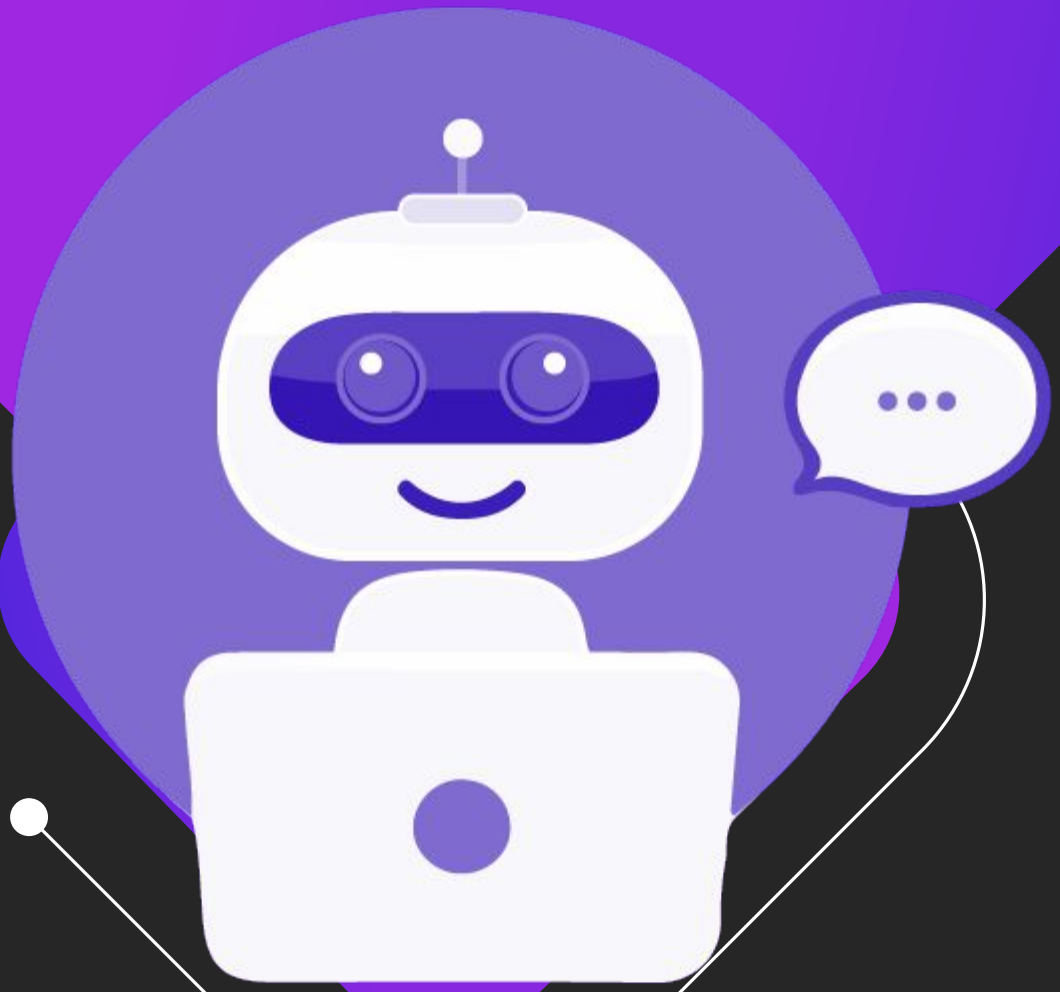
Intelligent Chatbot

Presenters
Stuti Jani (W1649923)
Sanfer Noronha(W1652189)
Anmol Varshney (W1652377)

Table Of Contents

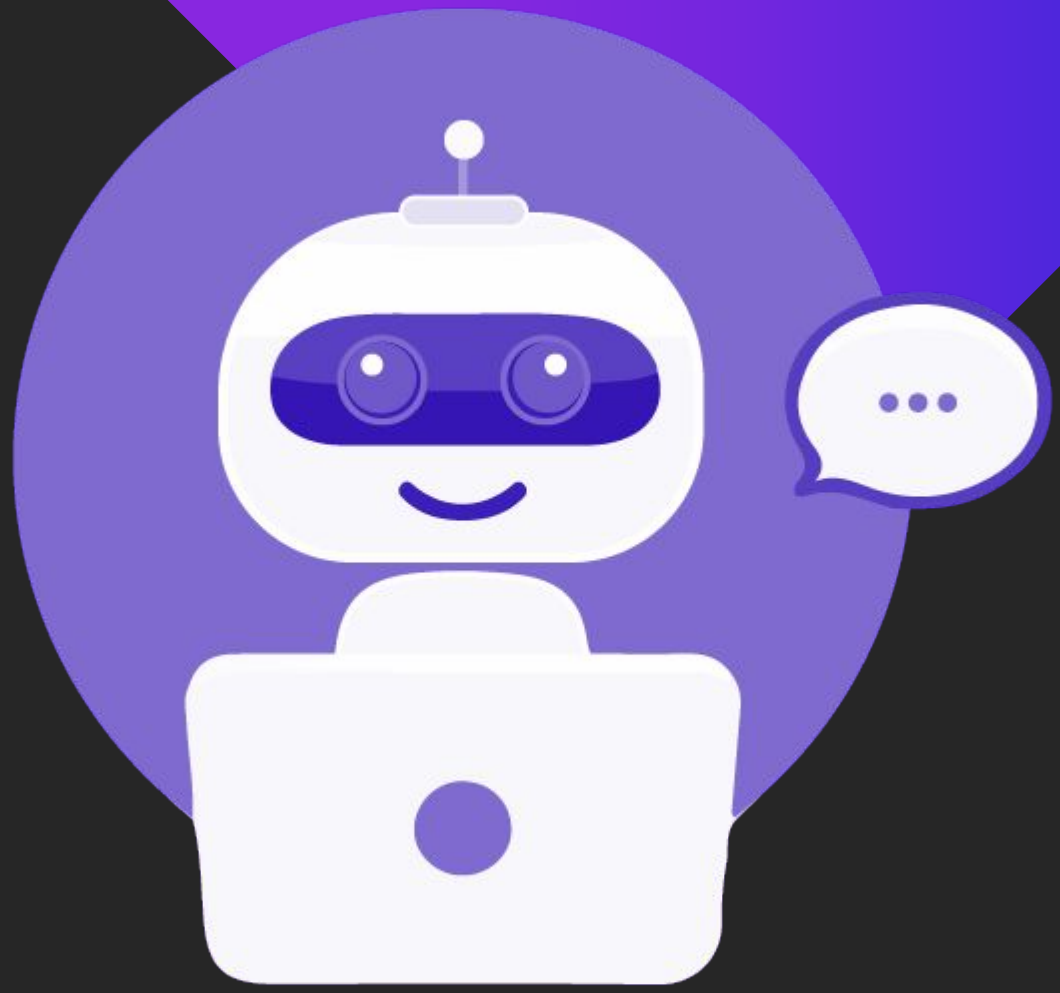


Problem Statement

The objective of this project is to create an intelligent chatbot tailored to serve as the primary customer service agent for a flight company.

Advantages In Customer Service

Chatbots in customer service offer 24/7 availability, quick response times, personalized interactions, and efficient issue resolution.



Enhancing User Experience

By providing seamless and personalized interactions, Chatbots enhance user experience, making interactions more efficient and satisfying.

Role Of NLP

Natural Language Processing (NLP) enables Chatbots to understand and process human language, allowing for more effective communication.

Requirement Analysis

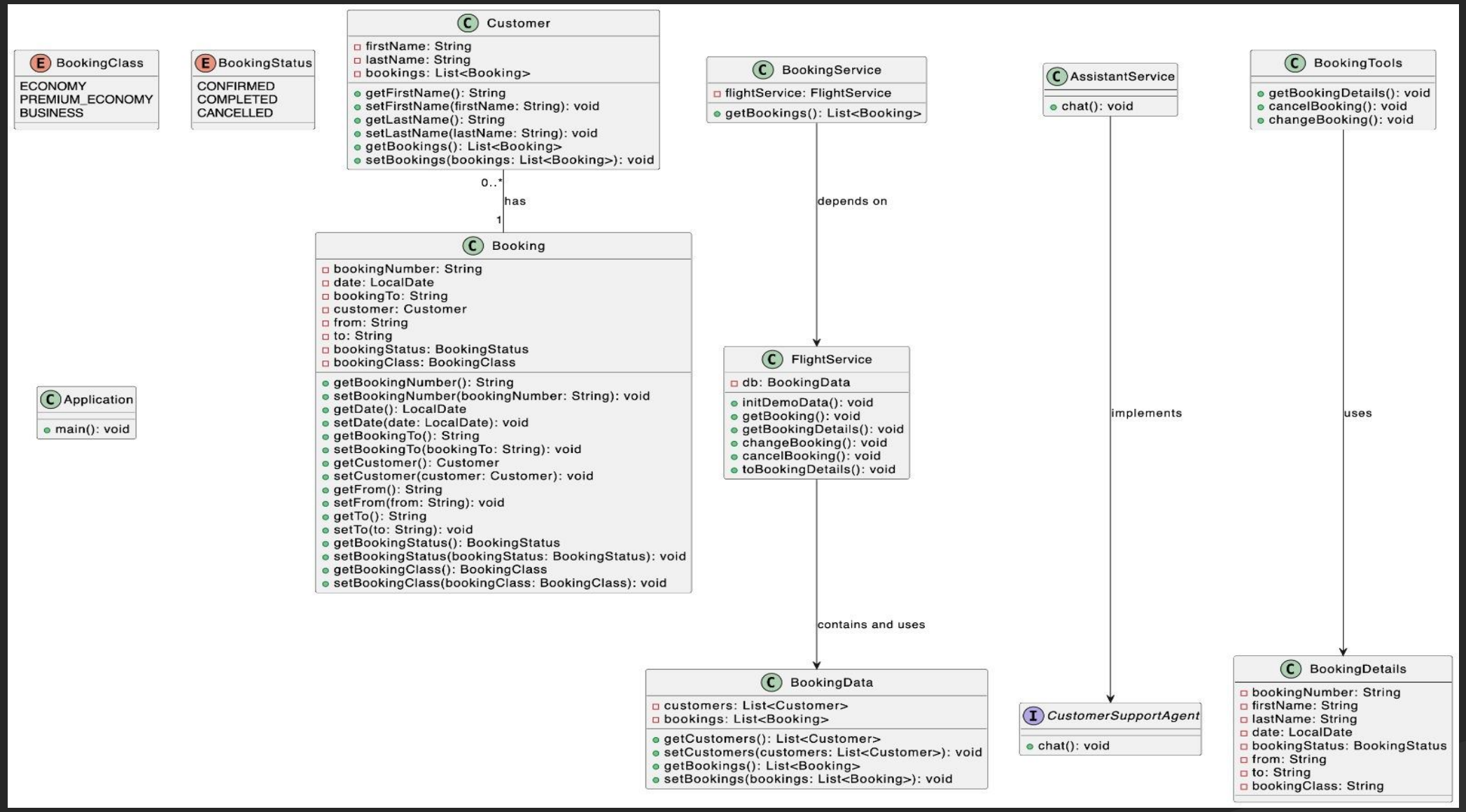
Functional Requirements

- **User Authentication**
- **Data Retrieval**
- **Booking Management**
- **Policy Enforcement**
- **User Consent**
- **Error Handling**

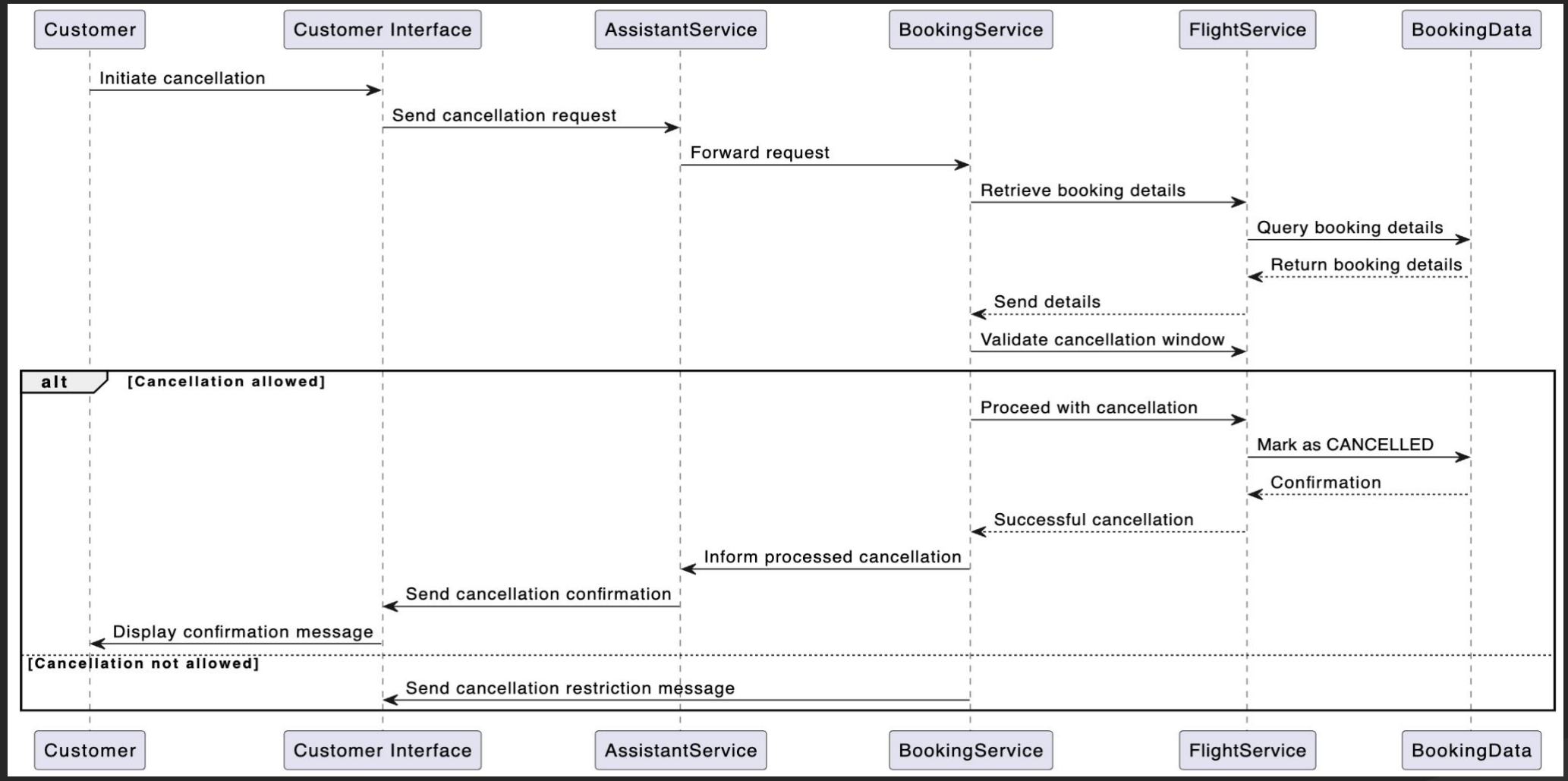
Non-Functional Requirements

- **Usability**
- **Reliability**
- **Performance**
- **Scalability**
- **Security**
- **Maintainability**

Design - Class Diagram



Design - Sequence Diagram



OO concepts

Classes and Objects

- Classes are fundamental units of structure in Java that encapsulate data and behavior. Objects are instances of classes that hold data and behavior defined by the class.

- Examples include classes like Application, BookingTools, CustomerSupportAgent, and interfaces like CustomerSupportAgent

OO concepts

Encapsulation

- Classes such as Application, BookingTools, CustomerSupportAgent, and interfaces like CustomerSupportAgent encapsulate related data and methods together, fostering modular and organized code structures.

- Encapsulation enhances code readability, maintainability, and scalability, promoting robust software design practices.

OO concepts

Composition

- Composition is a design principle in OOP where objects of one class are used as components in another class. It allows building complex functionalities by combining simpler, reusable components.

- Examples include the Application class using instances of other classes such as BookingTools, CustomerSupportAgent, etc.

OO concepts

Abstraction

- Abstraction in the provided codes is evident through interface definitions like `CustomerSupportAgent`, method declarations without implementations, dependency injection for decoupling dependencies, and annotations providing metadata and instructions.

- These abstractions enable modular, maintainable, and adaptable software design.

OO concepts

Modularity

- Modularity in the code is evident through separate classes like `Application`, `BookingTools`, and `CustomerSupportAgent`, each handling distinct functionalities. For instance, `BookingTools` manages booking-related operations, while `CustomerSupportAgent` focuses on customer interaction.
- This modular organization enhances code readability, maintainability, and scalability by isolating and encapsulating different aspects of the system.

DEMO

Conclusion

The SCU AIR Chatbot project presents a comprehensive solution for enhancing customer support services within the airline industry. By leveraging advanced technologies and maintaining a user-centric approach, the chatbot aims to streamline booking-related processes, improve customer satisfaction, and optimize operational efficiency for SCU AIR. With continuous refinement and adaptation to evolving customer needs and technological advancements, the chatbot holds the potential to become an indispensable asset in the airline's service ecosystem.

Thank You