GetawayGo

Architecture Diagrams

Fontys University of Applied Sciences

Anna Kadurina
17/01/2025

Table of Contents

Introduction

This document outlines the architecture of the GetawayGo platform, providing detailed diagrams and explanations for each level of the system's architecture. It includes interconnections between services, detailed explanations of the service bus, and specific diagrams for the service bus implementation.

C1

The C1 diagram provides a high-level overview of the GetawayGo system, focusing on the interactions between user roles and external systems.

Roles:

- Traveler: Searches and books properties.
- Host: Lists properties and manages bookings.
- Admin: Has access to all properties, users, reviews.

External Dependencies:

- Google Maps API: For location-based searches.
- Payment Gateway: For secure transactions.
- Email Service: SendGrid for sending booking confirmations.

Interconnections:

- The Traveler, Host and Admin interact with the GetawayGo system through a web interface.
- The system communicates with third-party APIs for locations, payments, and emails.

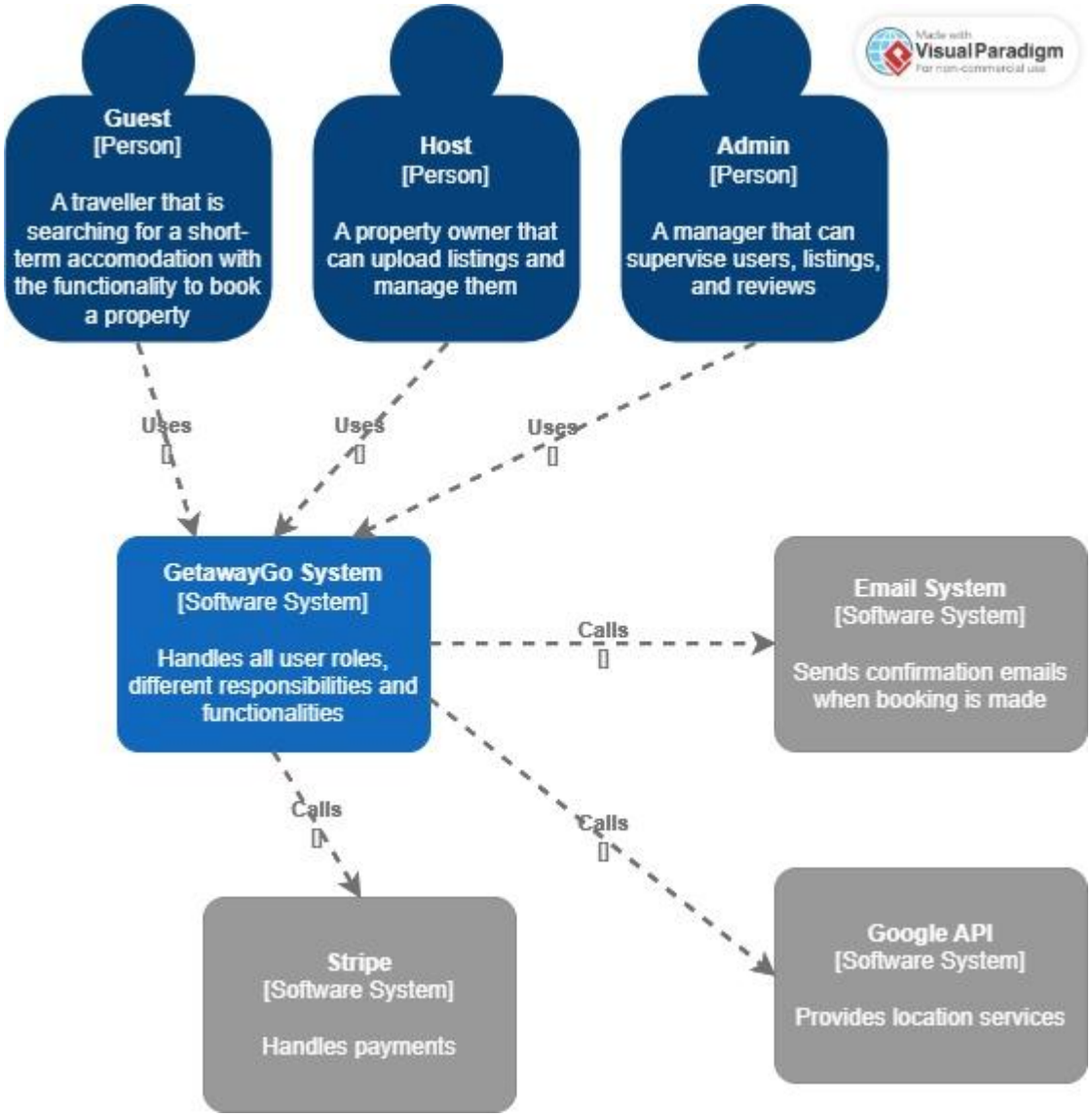

Figure 1 – C1 Diagram

C2

The C2 diagram zooms into the containers within the GetawayGo system. Each container represents a deployable unit or service.

Containers:

- API Gateway: Centralized entry point for client requests.
- User Service: Manages authentication and user profiles.
- Property Service: Handles property listings and availability.
- Booking Service: Processes bookings and manages statuses.
- Notification Service: Sends emails and notifications.
- Review Service: Handles reviews.
- AnalyticsService: Handles analytics.
- ChatService: Provides a chat.
- Service Bus: Enables asynchronous communication between services.

Interconnections:

- API Gateway routes requests to appropriate services.
- Services communicate directly using REST APIs or asynchronously through the Service Bus. Some examples are included in the below diagram, such as the connections from the BookingService to the PropertyService and NotificationService to BookingService. However, for readability, not all connections are included. Whenever, a service needs to call another, it uses REST API calls. Services communicate asynchronously through the service bus that is available in the diagram. Some services publish messages and other subscribe. Due to a lot of functionalities, not all arrows can be at the diagram.

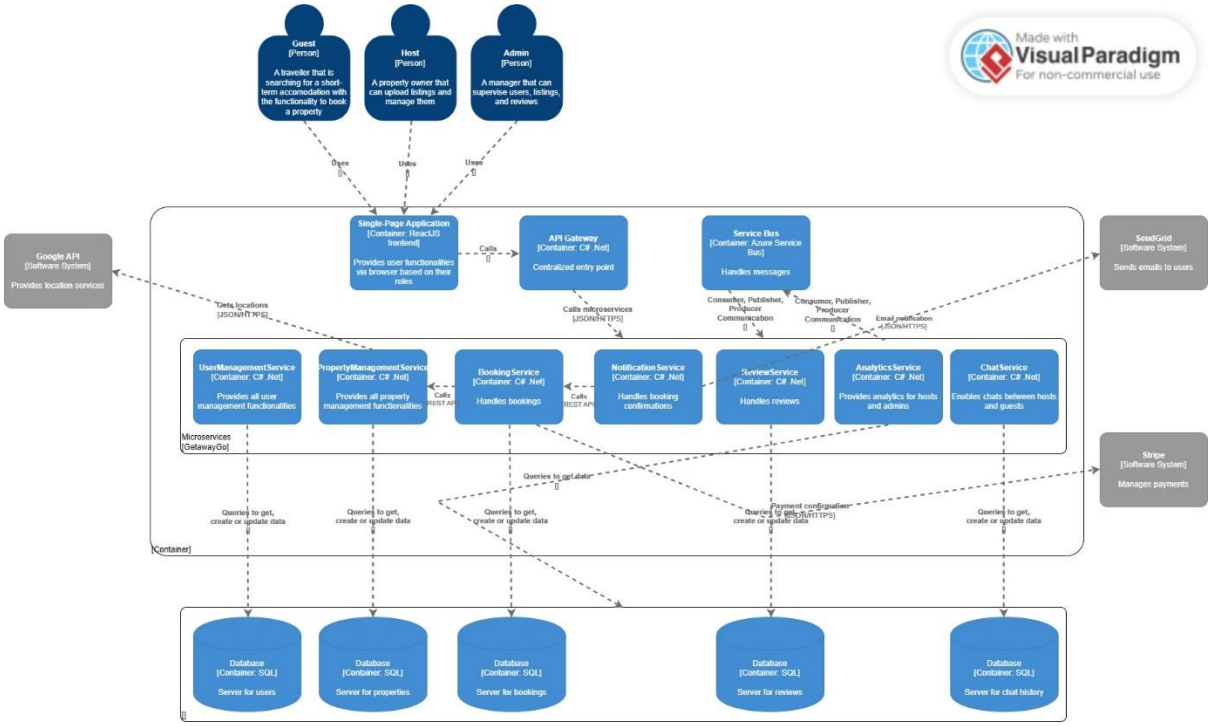

Figure 2 – C2 Diagram

C3

The C3 diagram details the internal components of each service. All services besides the NotificationService have Client, Controller, Service, and Repository that calls a database. The purpose of a Notification Service is to send notifications (emails, push notifications) to users. It acts more as a stateless service that processes requests and interacts with external systems (e.g., SendGrid). The AnalyticsService consumes data from other databases to gather analytics, so it calls the services and presents meaningful charts. Due to costs, it is best that this data is not persisted.

Interconnections**:**

- Each service's components interact internally via method calls (REST API).
- Events such as "PropertyBooked" are published to the Service Bus for other services to consume.

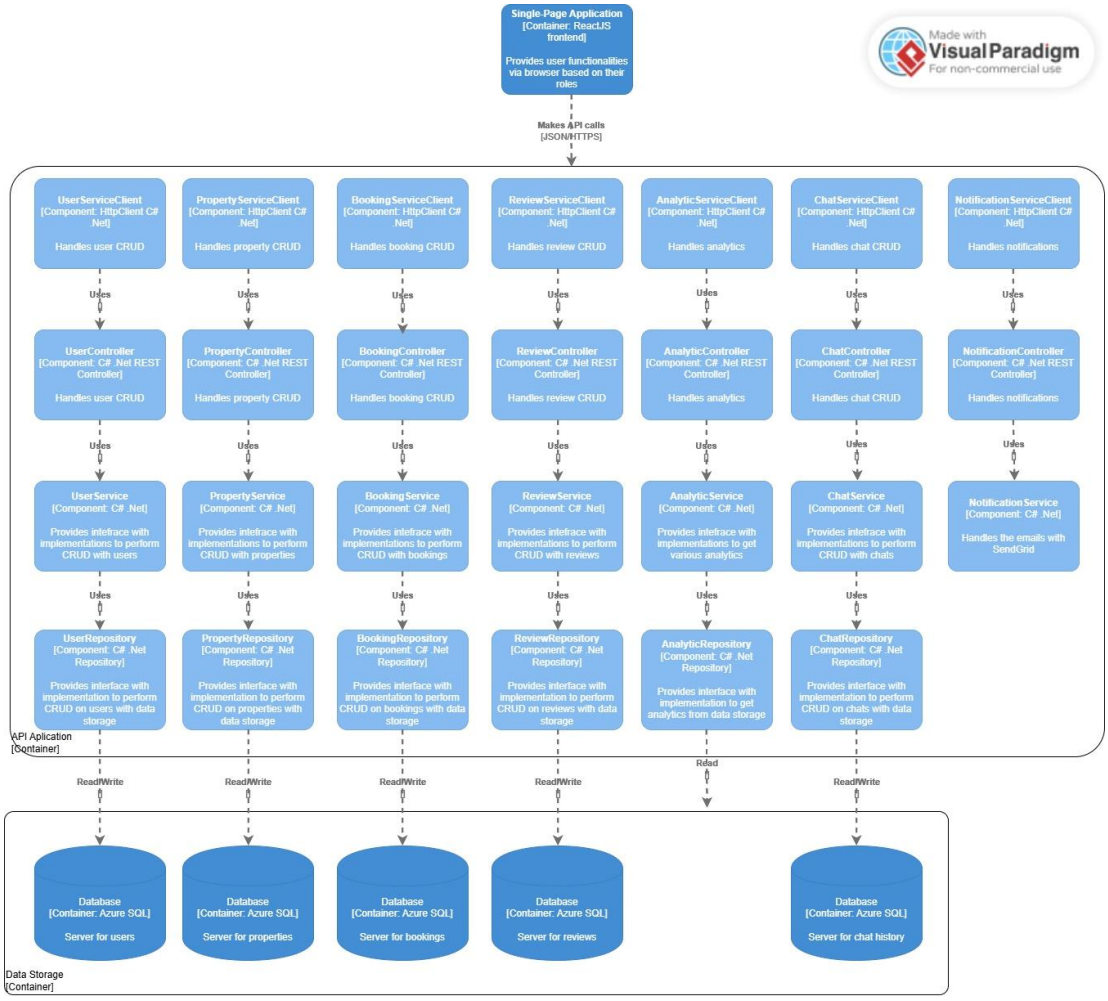

Figure 3 – C3 Diagram

Class Diagram

The class diagram illustrates the main classes in the system and their relationships.

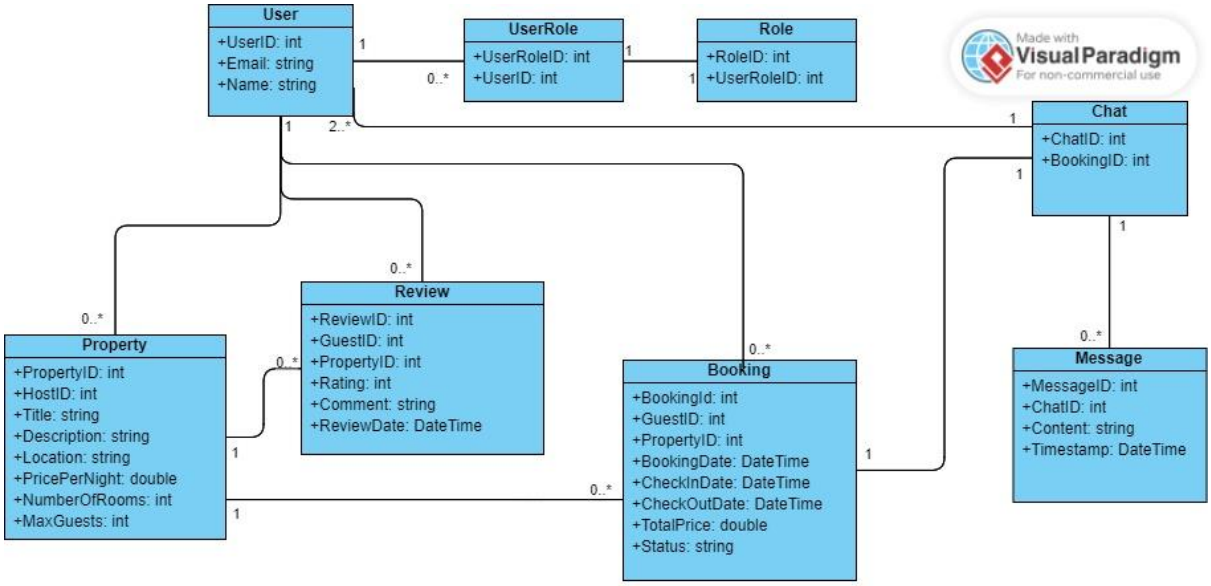

Figure 4 – Class Diagram

Service bus

The Service Bus is the backbone of the event-driven architecture in GetawayGo. It decouples services by facilitating asynchronous communication through publish-subscribe patterns.

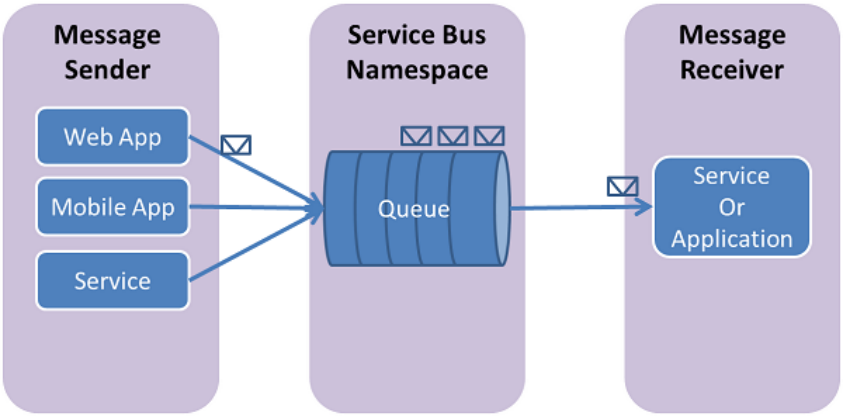

Figure 5 – Azure Service Bus

The example below shows how the Azure Service Bus works in the GetawayGo application. The API Gateway calls the service, then the service publishes a message on the respective queue of the service bus. Then the needed services subscribe to that message and perform the required actions.

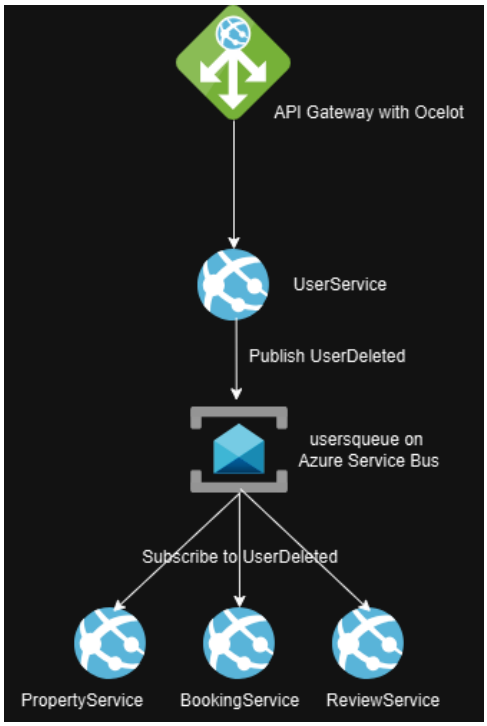

Figure 6 – Example of Azure Service Bus in GetawayGo