Fontys University of Applied Sciences

Non-functional requirements

GetawayGo

Anna Kadurina
13/11/2024

Table of Contents

Introduction

This document outlines the non-functional requirements for the GetawayGo platform, which is a web-based application designed to provide travellers with short-term accommodation. Non-functional requirements define the operational attributes and quality characteristics that the system must meet.

Goal

The goal of this document is to ensure that the platform is designed and developed to meet critical non-functional requirements that are essential for the smooth functioning of the application.

Scope

This document applies to the entire GetawayGo system, including all microservices, frontend, and API Gateway, as well as all the hosting infrastructure on Azure.

Non-functional Requirements

Performance

Performance refers to how quickly the system responds to user requests and how well it handles different levels of load.

- Response time: All pages should load within 3 seconds under normal usage.
- Throughput: The system should support at least 5000 concurrent users with less than 5% degradation in performance.
- Availability: The platform should be available 99.9% of the time, ensuring minimal downtime.

Strategies

- Caching: Implement caching mechanisms using Azure Cache for Redis
- Load balancing: Utilize Azure's load balancing features
- Performance Testing: Regularly conduct load testing using Azure Load Testing that utilizes Apache Jmeter

Scalability

Scalability refers to the system's ability to handle an increasing number of users and transactions without performance degradation.

- Vertical scalability: The platform should scale vertically to handle increased workloads (CPU, memory)
- Horizontal Scalability: The platform should be able to scale horizontally, adding more instances of services or databases
- Elasticity: The platform should scale dynamically based on traffic and usage patterns without manual intervention

Strategies

- Cloud Infrastructure: Leverage Azure App Service's auto-scaling capabilities
- Microservices: Design each service to be stateless and independently scalable

Security

Security refers to the system's ability to protect against unauthorized access, data breaches, and other malicious activities.

- Authentication and Authorization: Use OAuth 2.0 for user authentication and role-based access control (RBAC) for authorization
- Data Encryption: All sensitive data, such as user passwords and financial information should be encrypted both in transit and at rest
- GDPR compliance: The platform should adhere to the General Data Protection Regulation (GDPR) for user data storage, processing, and privacy
- Vulnerability testing: Implemented automated security testing and usage of tools like OWASP Zap in CI/CD pipeline
- Access logs: Maintain detailed access logs of user activity

Strategies

- Implement authorization and authentication with OAuth 2.0
- Follow OWASP best practices for securing APIs and web applications
- Schedule regular security testing (automatic as well)

Usability

Usability refers to how easy and intuitive the platform is for users to navigate and perform actions.

- User Interface: The platform should offer an intuitive, user-friendly interface with easy navigation

Strategies

- Accessibility testing: Perform manual and automated accessibility testing

Maintainability

Maintainability refers to how easily the system can be updated, monitored, and modified over time.

- Code Modularity: The platform should follow a microservices architecture.
- Logging and Monitoring: All critical services should have centralized logging.
- Automated Testing: The system should have a comprehensive suite of automated unit tests, integration tests, and end-to-end tests.

Strategies

- CI/CD pipelines: Set up CI/CD pipelines with Azure DevOps to ensure smooth deployment and testing

Reliability

Reliability refers to the system's ability to perform its intended function consistently without failure, and to recover from failures quickly.

- Disaster Recovery**:** The system should have a disaster recovery plan in place.

Strategies

- Disaster Recovery Plan: Create a detailed plan to follow in case of a disaster

Conclusion

This document outlines key attributes that are crucial for GetawayGo. By adhering to these requirements and implementing the outlined strategies, the system will be able to support high-performance demands, scale with user growth, maintain security and comply with legal and ethical standards.