Group 03 – Table of Five™

Hotel Management Website
Software Architecture Document

Version 1.3.0

Revision History

Date	Version	Description	Author
10/12/2022	1.3.0	Create and Write the Introduction, Architectural Goals and Constraints, Use-Case Model, Logical View of SAD	Le Tran Trung Hieu Vu Huy Hoang Tran Le Hoang Bao Lam Vi Nguyen

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

This Software Architecture Document provides an architectural overview of the Hotel management system. The Hotel management system is being developed to support managing the services of a hotel online

2. Architectural Goals and Constraints

2.1 Objective:

- The user who is the customer can book the room online on the website.
- The user who is the customer can buy the hotel service online in the website or in the hotel .
- The user who is the customer can be supported by the staff of the hotel online in the website.
- The user who is the staff can see their work schedule on the website.
- The user who is the staff role and the manager of the branch of the hotel can manage the employees by making the schedule for them, making a meeting, checking the process of working, salary,…
- All the users can check the personal information on this website. You can edit it if something you need to change.
- The bot can notify some news, some changes of the website or some important things to the user.
- The admin will create the database of the user, maintain them, and be responsible for all databases.
- The admin will maintain and update the website as they like and notify the user.

2.2 Software requirements:

- -Any device that support Chromium 107 or higher (*Have at least 2 physical cores at 1Ghz, 2GiB more more RAM)
- Any operating system that support Chromium 107 or higher
- The application must be able to handle 100 users at its peak. Every application resource should be smaller than 5MiB per file.
- Every request sent should be responded to in less than 1000ms within Vietnam. Package lost rate should be less than 1%.
- The environment must allow applications to perform read/write files/cookie operations.

3.   Use-Case Model
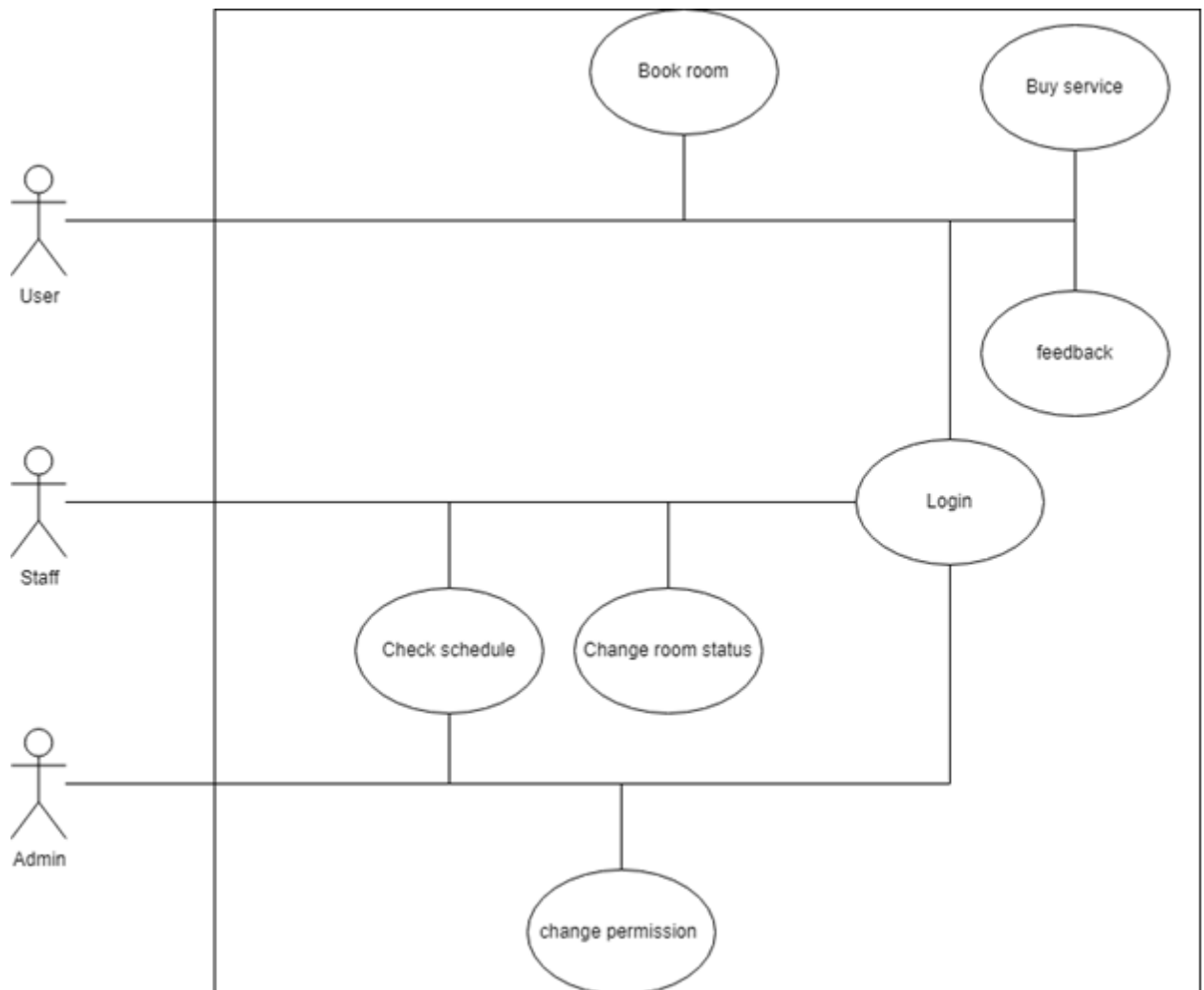
3.1: Login

This use case allows the user to access the services of the hotel management system. The actor is all users

3.2: Book room

This use case allows the user to choose a room they would like to book, see the information and book the room. The actor is the customer

3.3: Buy service

This use case allows the user to buy extra services not included with the room they book. The actor is the customer

3.4: Feedback

This use case allows the user to send their feedback about the hotel services after using it to help the hotel better improve their services. The actor is the user

3.5: Check schedule

This use case allows the user to see their schedule and which task has been assigned to them in the case of a staff member, and to see all of the schedule in the case of the admin. The actor is staff and admin
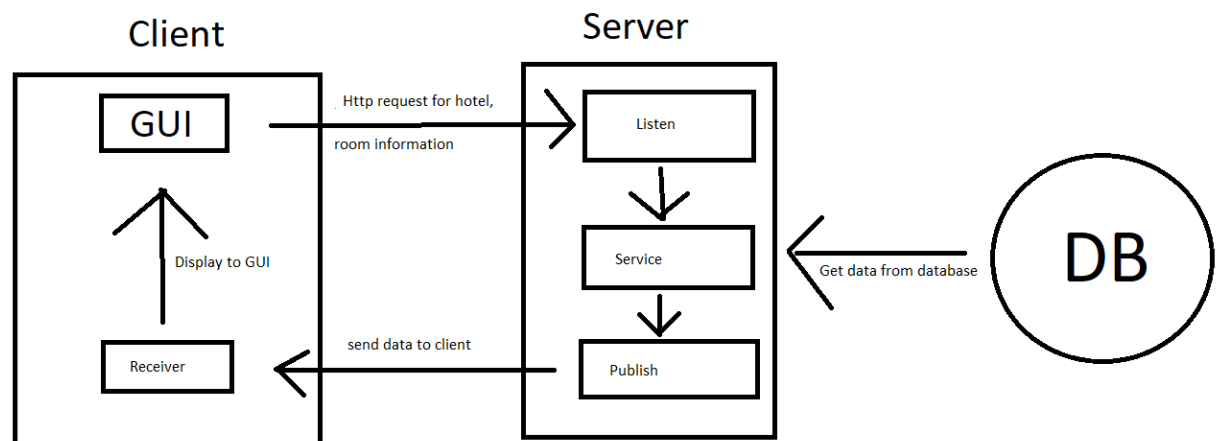
3.6: Change room status

This use case allows the user to change the room status into different states. The actor is the staff.
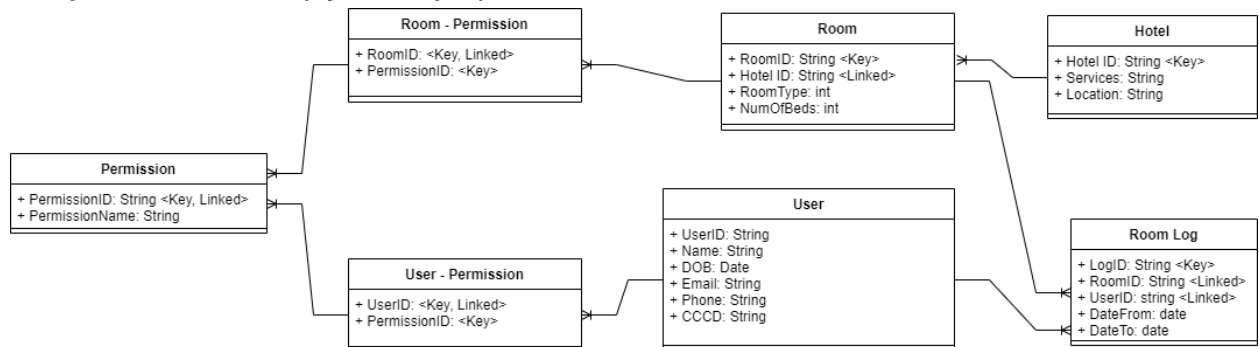
3.7: Change permission

This use case allows the user to change the permission of others with lower permission. The actors are the staff and admin.

4.   Logical View

4.1 Component: Client (Fluter)

4.2 Component: Server (Python REST API)

4.3 Component: Database (Python Sqlite)

5. Deployment

[Leave this section blank for PA4.

In this section, describe how the system is deployed by mapping the components in Section 4 to machines running them. For example, your mobile app is running on a mobile device (Android, iOS, etc), your server runs all components on the server side including the database]

6. Implementation View

[Leave this section blank for PA4.

In this section, provide folder structures for your code for all components described in Section 4.]