SBI Managment system

Kevin Patel
Jeff Jordan

Abstract

SBI Management System, named after the flagship Stone Brook Inn, is a program for use in the commercial Hospitality business in order to easily navigate the processes of booking, reserving, and maintaining rooms in an effective way to alleviate stresses of overbooking and unavailable rooms within businesses.

1. Introduction

The purpose of the SBI Management system is to remove the clerical errors of record keeping by hand and to cut down on the potential mistakes that could happen with note taking with rotating cast of managerial staff and employees. It is designed around the idea of condensing and centralizing all reservation and booking information into one simple, easy-to-use application in order to remove the stress of record keeping and room tracking.

On the backend, it will keep track of non-confidential customer information. The purpose of this is to implement features such as a customer rewards program, discounts, personalized service, etc. In the future, it will implement encryption for payment information.

1.1. Background

A member of our group's family owns a hotel. This project is being designed with the needs of an actual hotel business in mind. The design will be centered around a simple GUI to quickly assign rooms and check availability.

While the primary purpose of the project is for the Stone Brook Inn, we will design it to be implemented by any local small hotel or motel. The idea is to keep the functionality universal while making design edits simple and easy to understand.

1.2. Challenges

Integrating legal and secure encryption for web payments will be the ultimate challenge for the project. Ensuring that the program doesn't allow for any booking of unavailable rooms will be a challenge since there will be a lot of test cases that might go unnoticed. Designing a universal program that could be adjusted to fit any business model might present itself to be more challenging than we can currently foresee.

2. Scope

The scope of this program will be extensive. It will handle guest checkins and checkouts, it will feature an intuitive and powerful interface, it will be adjustable to any hotel, it will ensure no overbooking issues arise, will keep track of customer rewards points, and lastly, feature a cross-shift note system.

2.1. Requirements

The main requirement is a visual of the hotel's rooms and what it available, as well as what's already reserved. Another requirement is the functionality to take in a customer's information and turn a vacant room into a reserved room.

2.1.1. Functional.

- Visual map of the rooms.
- Color coding of availability.
- User information field.
- Notes

2.1.2. Non-Functional.

- Security – The note system should be hidden until clicked.

1	Release occupancy
2	Fill vacancy

TABLE 1. USE CASES

2.2. Use Cases

Use Case Number: 1
Use Case Name: Booking and Maintaining Rooms
Description: Administrators at motel and hotel businesses will be able to block rooms once they are booked by customers or repairs/cleaning needs to take place.
Use Case Number: 2
Use Case Name: Reservations
Description: Interact with a calender in order to properly plan out and schedule rooms to be booked and blocked to prevent overbooking.

2.3. Interface Mockups

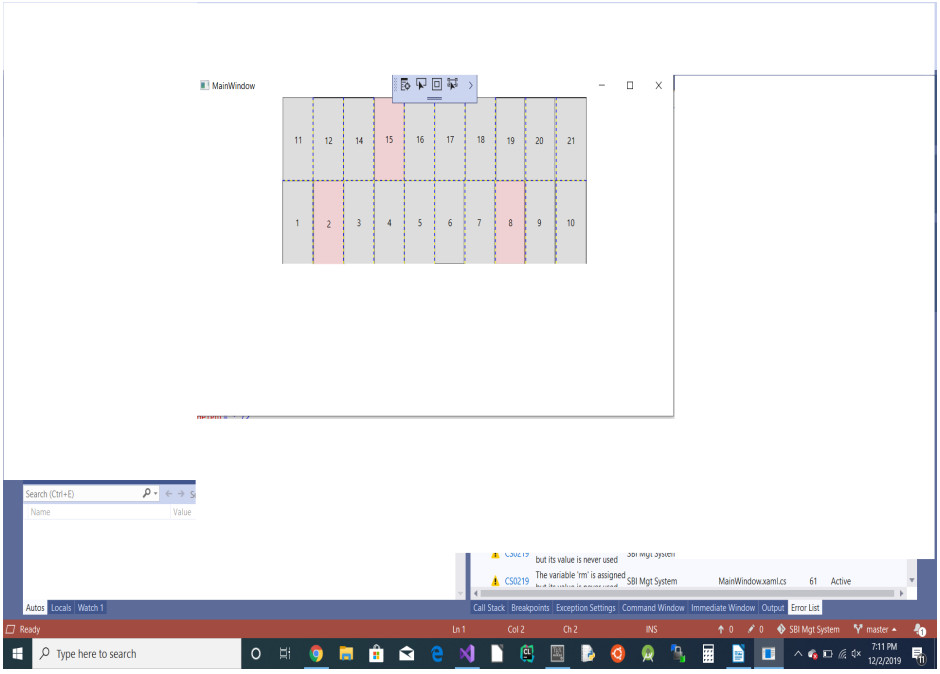

Figure 1. Main window of the program

3. Project Timeline

A large portion of our time spent on this project was on design. To be honest, too much time was spent on the design. It wasn't until, two months in, that we started the actual coding of the project, that the design fell into place all on its own.

Within two days, we had moved past the design and delved into the requirements. First came the buttons themselves. Each button needed to act separately from the others, so it took over a month to manage everything that needed to happen upon button clicks. Some of these issues were color changes, form population, and a note system.

This last month has been committed to polishing and trying to learn a database. We eventually gave up on the database and will store all information as notes.

4. Project Structure

By the project's culmination, it turned into a day to day system to manage availability of rooms for the hotel. It has a working GUI, and upon filling in the information about reservation, it will mark the room unavailable with a color change. There is another button that will change it back to available.

All information is stored in a display box for notes, in lieu of a database. We realized there is no point since they have no rewards system in place at the hotel yet.

4.1. UML Outline

4.2. Design Patterns Used

The decorator design pattern was used on buttons. Upon another button's clicking, a separate button will change color.

5. Results

This section will start out a little vague, but it should grow as your project evolves. With each deliverable you hand in, give me a final summary of where your project stands. By the end, this should be a reflective section discussing how many of your original goals you managed to attain/how many desired use cases you implemented/how many extra features you added.

5.1. Future Work

Future work will include tweaking as it's used in the field. There will need to be a save system in case of recovery upon crash. If the hotel ever adds a reward system, there will need to be a database implemented.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.