# Project Plan Documentation: Hotel Reservation System for Hotel "Astana"

Margulan Zhamantayev

Smart Technologies, Astana IT University

ST-2504, Object-Oriented Programming

Mr. Imran Khaider

January 27, 2026

# Project Plan Documentation: Hotel Reservation System for Hotel "Astana"

## Team Members and Role Distribution

The project is led by a specialized development role to ensure technical integrity:

- **Margulan (Lead Java Developer):** Responsible for the core Java implementation, designing the Object-Oriented Programming (OOP) structure, and developing the primary business logic.

## Project Description, Goals, and Objectives

### Description

The Hotel Reservation System is a console-based Java application designed to simulate the essential operations of a professional hotel environment. It serves as a management tool for hotel staff to handle room allocations, guest registrations, and reservation lifecycles.

### Goals and Objectives

The primary goal is to provide a structured, efficient system for managing a small hotel's daily operations. Key objectives include:

- Efficiently tracking room availability and guest information.
- Preventing operational errors such as double bookings.
- Demonstrating advanced OOP principles, including encapsulation, inheritance, abstraction, composition, and aggregation.

### Novelty and Uniqueness

Unlike generic database-driven applications, this project focuses on the **architectural integrity** of OOP. By utilizing complex relationships like **composition** and **aggregation**, the project provides a high-fidelity simulation of real-world business logic within a lightweight, console-based environment.

## Technologies Used

The system leverages a modern Java-based stack to ensure portability and code quality:

- **Programming Language:** Java (Standard Edition).
- **Paradigm:** Object-Oriented Programming (OOP) for modular and maintainable code.
- **Development Environment (IDE):** IntelliJ IDEA.
- **Version Control:** Git and GitHub for collaborative tracking and source management.
- **Interface:** A text-based Console Interface for streamlined user interaction without the overhead of a GUI.

## Project Structure and Class Interactions

The system is designed around five central entities that interact to maintain state and logic:

### Class Responsibilities

- **Room:** Manages specific room attributes (number, type, price) and its current status.
- **Guest:** Stores personal identifiers and contact details.
- **Reservation:** Acts as the bridge between a Guest and a Room, handling check-in/out dates.
- **Hotel:** The logic engine that holds the collection of rooms and reservations; it handles the creation and cancellation of bookings.
- **HotelSystem (Main):** The entry point that orchestrates the user interface and program flow.

## Minimum Viable Product (MVP)

The MVP focuses on the core functional requirements necessary to run a hotel:

1. **Room Management:** The ability to add and store hotel rooms in the system.
2. **Guest Registration:** Capturing guest details for the records.
3. **Booking Engine:** Creating reservations while checking for room availability to prevent double-booking.
4. **Operational Tools:** Functions to cancel reservations and view currently available rooms or existing booking details.

## Post-MVP Improvements

Once the core functionality is stabilized, the following features are planned:

- **Dynamic Pricing:** Calculating total costs based on the duration of stay.
- **Room Categorization:** Support for Standard, Deluxe, and Suite types.
- **Search Functionality:** Locating reservations specifically by guest name.
- **Administrative Features:** An admin-specific menu and status tracking (e.g., Checked-in vs. Pending).

## Development Timeline

The project follows a four-week lifecycle:

| Phase | Timeline | Primary Tasks |
|---|---|---|
| **Phase 1: Planning** | Week 1 | Requirements definition, class identification, and GitHub setup. |
| **Phase 2: Core Coding** | Week 2 | Implementation of Room, Guest, and Reservation classes; core Hotel logic. |
| **Phase 3: Integration** | Week 3 | Implementation of Main class, MVP testing, and documentation. |
| **Phase 4: Finalization** | Week 4 | Optional features, code refactoring, and final report preparation. |

## GitHub Repository

The complete source code and version history can be found at:

https://github.com/tashtik6-png/HotelReservationSystem/