Hotel Management System

This is a Hotel Management System implemented in C++. It allows users to perform various tasks related to hotel management such as making reservations, checking in and out, viewing room details, and managing hotel amenities.

Features:

The Hotel Management System provides the following features:

1. **Add Room**: Allows adding rooms to the hotel with different types and availability.
2. **Make Reservation**: Enables users to make a reservation by providing personal details and check-in/check-out dates.
3. **Display Rooms**: Shows the available rooms and their details.
4. **Display Reservations**: Displays the reservations made by customers.
5. **Check-In**: Updates the status of a reservation to checked-in.
6. **Check-Out**: Updates the status of a reservation to checked-out and calculates the bill.
7. **Room Details**: Provides information about different types of rooms and their amenities.

8. **Hotel Amenities**: Allows users to select and calculate the cost of additional services such as breakfast, lunch, gym, etc.

9. **Print Bill**: Generates a bill for the customer based on the duration of stay and selected amenities.

How to Use:

1. Clone the repository or download the source code files.
2. Compile the code using a C++ compiler.
3. Run the compiled executable file.
4. Follow the menu instructions to perform various tasks.

Dependencies:

The Hotel Management System code has the following dependencies:

- C++ Standard Library
- C++ String Library
- C Time Library
- C Windows Library
- C Math Library

Contributing:

Contributions are welcome! If you find any bugs or want to add new features, please contact us.

Authors:

Raghav Kejriwal

Faiz Khan

Kavya Verma

Omar Shaikh

Pranath Kedia

Ridhimann Sabharwal

Vaibhav Jindal

Acknowledgments:

1. **cplusplus.com**: This website provides a comprehensive reference guide for the C++ language. It includes tutorials, a reference manual, examples, and a helpful forum for discussions.
2. **GeeksforGeeks**: GeeksforGeeks is a well-known platform for computer science resources. It offers a wide range of tutorials, articles, and coding practice problems for C++ and other programming languages.
3. **Codecademy**: Codecademy is an interactive learning platform that offers C++ courses for beginners. It provides an interactive coding environment and step-by-step lessons to help you grasp the fundamentals of the language.

4. **LearnCpp.com**: LearnCpp.com is an online tutorial website specifically dedicated to teaching C++ programming. It covers topics from basic syntax to advanced concepts, with code examples and exercises.
5. **SoloLearn**: SoloLearn is a mobile app that offers C++ courses with a gamified learning experience. It provides bite-sized lessons, quizzes, and a coding community where you can connect with other learners.
6. **Stack Overflow**: While not specifically a tutorial site, Stack Overflow is an invaluable resource for C++ programmers. It is a question and answer platform where you can find solutions to specific coding problems and learn from the expertise of the programming community.
7. **GitHub**: GitHub hosts a vast collection of open-source C++ projects. Exploring the codebase of well-established projects can help you learn best practices, coding techniques, and gain hands-on experience

Additional Notes:

Caution-

1. Validate input data types: Ensure that the user provides input in the expected data type. For example, if you're expecting an integer, verify that the input is indeed a valid integer before using it in calculations or assignments.
2. Bounds checking: Check if the user input falls within the acceptable range or bounds. For example, if the user is expected to enter a positive integer, validate that the input is greater than zero.
3. Handle input errors: Be prepared to handle cases where the user provides invalid input. For example, if the user enters a non-numeric value where a number is expected, provide an appropriate error message and prompt the user to enter valid input.

4. Buffer overflow protection: If you are reading input into a fixed-size buffer (e.g., using **fgets**), ensure that the input does not exceed the buffer size to prevent buffer overflow vulnerabilities.
5. Sanitize input: If the user input is used to construct queries, commands, or filenames, make sure to sanitize the input to prevent injection attacks. For example, when constructing database queries, use parameterized queries or prepared statements instead of directly concatenating user input into the query.
6. Handle exceptions: Use exception handling to gracefully handle any exceptional situations that may arise from user input. This will help prevent crashes or unexpected program behavior.
7. Test with malicious input: Perform testing with various types of input, including edge cases and potentially malicious input, to ensure that the program behaves correctly and securely in all scenarios.

SPECIAL MENTION:

This project could be made at this level only because of Professor Padma Priya R.

Prof. Padma Priya R is an exceptional OOPS teacher who possesses a profound understanding of object-oriented programming concepts. With her unwavering passion for teaching and vast knowledge in the field, she creates an engaging and supportive learning environment for her students. Prof. Padma Priya's dedication to delivering comprehensive and practical lessons has greatly influenced her students' understanding and application of OOPS principles. Her approachable nature and willingness to go the extra mile to ensure student success make her an invaluable asset in the classroom. Prof. Padma Priya's commitment to nurturing future programmers and fostering a deep appreciation for OOPS is truly commendable. Her impact on her students' academic and professional journeys is immeasurable, and she is an inspiration to both her peers and students alike.