**Project: Hospital Management System**

**Project Description:**

You are tasked with developing a Hospital Management System using object-oriented programming in C++ with a strong focus on utilizing the Standard Template Library (STL). The system should provide functionalities for managing patient records, scheduling appointments, tracking medical staff, and handling medical procedures. It should incorporate various STL containers, algorithms, and iterators to optimize data storage and processing.

**Requirements:**

1. Patient Class Hierarchy:

- Create a base class called "Patient" to represent general patient information (name, age, gender, etc.).

- Derive specific patient types, such as "Inpatient" and "Outpatient", from the base "Patient" class. Include additional attributes and behaviors specific to each type.

- Implement appropriate constructors, destructors, and member functions for the classes.

2. Exception Handling:

- Implement exception handling to handle potential errors or exceptional situations in the program, such as invalid input, scheduling conflicts, or file operations.

- Create custom exception classes for specific scenarios, such as "InvalidAgeException" or "AppointmentConflictException".

3. STL Integration - Data Storage and Manipulation:

- Utilize appropriate STL containers, such as vector, list, or map, to efficiently store and manage patient records.

- Implement operations to add new patients, search for patients based on name or ID, and display the list of all patients using STL algorithms and iterators.

4. Appointment Scheduling:

- Implement a class called "Appointment" to represent an appointment. It should store the patient information, the medical staff assigned, the appointment time, and any necessary procedures.

- Utilize STL containers to handle appointment scheduling, track appointment conflicts, and manage appointment cancellations or rescheduling.

5. Medical Staff Management:

- Create classes to represent different medical staff, such as "Doctor" and "Nurse". Include attributes such as name, specialization, and availability.

- Utilize STL containers and algorithms to manage medical staff, assign them to appointments, and handle their schedules.

6. Virtual Functions and Polymorphism:

- Implement virtual functions in the base "Patient" class and override them in the derived patient types.

- Demonstrate polymorphism by utilizing STL containers of base class pointers to store and process both base "Patient" and derived patient type objects.

7. User Interface:

- Implement a user interface to interact with the Hospital Management System.

- Allow users to add patients, schedule appointments, manage medical staff, handle exceptions gracefully, and display relevant information.

- Provide appropriate menus, prompts, and error messages to enhance usability.

8. Data Validations:

- Implement data validations to ensure the integrity of user inputs, such as validating age, appointment time, and other relevant data.

- Display appropriate error messages when invalid data is entered.

9. Documentation:

- Write a comprehensive project report documenting the design and implementation of the Hospital Management System.

- Include an overview of the system, class diagrams, data flow diagrams, and detailed explanations of the implemented functionalities.

- Clearly explain the usage of STL containers, algorithms, and iterators in the project.

- Add comments throughout the code to explain the functionality of each class, function, and exception handling mechanism.

10. Testing:

- Write test cases to validate the implemented functionalities and handle various scenarios, such as scheduling conflicts, searching for non-existent patients, etc.

- Document the test cases and their expected outcomes in the project report.

**Project Report Guidelines:**

- Write a project report using a word processor or document editor.

- Include a cover page with your name, student ID, course details, and submission date.

- Structure the report with appropriate headings and subheadings.

- Provide an introduction to the Hospital Management System project and its objectives.

- Include class diagrams illustrating the class hierarchy and relationships between classes.

- Explain the implemented functionalities, highlighting the integration of STL containers, algorithms, and iterators.

- Describe the exception handling mechanism and the custom exception classes.

- Document any design patterns or design principles applied during the implementation.

- Discuss the data validations implemented and how they ensure data integrity.

- Provide screenshots or examples of the user interface to illustrate the system's usage.

- Present the test cases and their outcomes, demonstrating the correctness of the implemented functionalities.

- Conclude the report by summarizing the project's achievements, challenges faced, and lessons learned.

- Add a reference section citing any external resources or libraries used.

Additional Instructions:

- Write clean, readable, and well-documented code.

- Use meaningful variable and function names to enhance code readability.

- Comment your code extensively, explaining the purpose of classes, functions, and complex logic.

- Follow coding style guidelines and indentation practices consistently.

- Apply appropriate error handling techniques and provide descriptive error messages.

- Conduct thorough testing to ensure the correctness of your implementation.

- Include relevant error handling and validation cases in your testing.

- Use git or other version control systems to track your code changes and ensure backup.

- Make regular commits with meaningful commit messages to track your progress.

- Prepare a README file explaining how to compile and run the program, as well as instructions for interacting with the user interface.

**Grading Criteria:**

- Correctness and functionality of the implemented Hospital Management System.

- Proper usage of inheritance, polymorphism, encapsulation, exception handling, and data validations.

- Effective utilization of STL containers, algorithms, and iterators for data storage and manipulation.

- Quality and organization of the code.

- Clarity, completeness, and professionalism of the project report.

- Demonstrated understanding of object-oriented programming concepts, exception handling, and STL integration in C++ and other relevant topics.

**Deliverables:**

1. Project Report (Template will be provided, you can add additional sections as per project report guidelines)
2. Code files (.cpp/.h only)
3. Output screenshots
4. Recorded Presentation (YouTube link)

*Please note that there is contributions table and academic integrity statement included in the Project Report that you are required to fill in.*

Note: This project requires a solid understanding of object-oriented programming concepts, exception handling, and STL integration in C++. It is crucial to apply appropriate design principles, follow best practices, and utilize STL containers, algorithms, and iterators effectively to optimize data storage and processing in the Hospital Management System.