SOFTWARE REQUIREMENT SPECIFICATION FOR TIME TABLE GENERATION

STUDENT NAME: DHARSHINI R
PROJECT ID: 23
PROJECT TITLE: TIME TABLE GENERATION

TECHNICAL COMPONENTS :

COMPONENT TECH STACK
BACKEND NODE JS , EXPRESS JS
FRONT END NODE JS
DATA BASE MONGODB
API OPEN API

PROBLEM STATEMENT:

The input consists of the number of classes, subjects, teachers, and time slots available each week. Ensure no teacher is double-booked and that each class has the required subjects scheduled without conflicts. The output should display a weekly timetable for each class, indicating the subject and teacher assigned to each time slot. The program should handle constraints like specific teacher availability and subject . Implement error checking for input validity and provide a clear, formatted output for easy readability of the schedules.

PROJECT - FLOW:

Purpose:

Creating timetables for classes ensures efficient use of resources, optimizes learning, and maintains a balanced workload for students and teachers. It coordinates classroom availability, subject allocation, and teacher schedules, preventing conflicts and overlapping. Timetables help manage time effectively, ensuring each subject receives adequate attention and instructional time. They also facilitate smooth school operations, improve academic planning, and enhance overall productivity. Additionally, timetables support extracurricular activities, breaks, and administrative tasks, promoting a well-rounded educational environment. Ultimately, well-structured timetables contribute to a disciplined, organized, and productive learning experience for all stakeholders involved.

Scope:

The scope of this problem includes developing an algorithm to generate weekly timetables for classes, ensuring no teacher is double-booked and all required subjects are scheduled without conflicts. It must consider teacher availability, subject frequency, and perform error checking for input validity, producing a clear, formatted output.

Business context:

The specialized platform caters to a BIT institution, use timetables to optimize classroom and teacher usage, ensuring smooth delivery of courses and efficient student learning.

Consideration:

- All users possess active Google accounts for authentication.
- Users have regular access to internet-enabled devices.
- **Camp Management:** Streamline camp activity scheduling by automating timetable creation. This ensures efficient allocation of resources and staff, allowing for a diverse and experience for all participants.

Dependencies :

Generating a timetable for classes on the CAMPS website requires several dependencies:

1. **Database:** To store and retrieve information about classes, subjects, teachers, rooms, and schedules.
2. **User Interface:** For inputting data and displaying the timetable.
3. **Algorithms:** For conflict-free schedule generation considering constraints like teacher availability, subject requirements, and room allocation.
4. **Authentication System:** To ensure only authorized personnel can create or modify timetables.
5. **Error Handling:** To manage and resolve scheduling conflicts and input errors.

The timetable should be accessible and editable online, ensuring efficient and streamlined class management.

User Personas:

Creating timetables for classes on a campus website involves designing a user-friendly interface where students and faculty can manage schedules efficiently. The process includes:

1. **Students:** View and customize class schedules, check room allocations, and see available slots.
2. **Teachers:** Input availability, manage class timings, and access assigned classrooms.
3. **Administrators:** Oversee the entire scheduling process, resolve conflicts, and allocate resources.

Time Table Generation :

1. List all courses, instructors, and rooms.
2. Use algorithms to optimize schedules based on availability and requirements.
3. Allow real-time updates and notifications for changes.

Functional Requirements

User Authentication:

- Implement Google account authentication for all users.
- Ensure only authorized personnel can create, modify, or view timetables.

Input Handling:

- Allow input of the number of classes, subjects, teachers, and time slots available each week.
- Implement error checking for input validity.

Teacher Availability:

- Enable input of specific teacher availability.
- Ensure no teacher is double-booked in the timetable.

Class Scheduling:

- Ensure each class has all required subjects scheduled without conflicts.
- Consider subject frequency and distribute subjects evenly throughout the week.

Timetable Generation:

- Develop an algorithm to generate conflict-free weekly timetables.
- Optimize schedules based on teacher availability, subject requirements, and room allocation.

Output Display:

- Provide a clear, formatted output displaying the weekly timetable for each class.
- Include subject and teacher assigned to each time slot.

User Interface:

- Design a user-friendly interface for inputting data and displaying the timetable.
- Ensure the interface supports students, teachers, and administrators in managing schedules efficiently.

Error Handling:

- Implement robust error handling to manage and resolve scheduling conflicts and input errors.
- Provide clear error messages and guidance for resolving issues.

Non-functional requirements:

Performance: The timetable generation algorithm should complete within a reasonable time frame (e.g., under 5 seconds) for up to 200 classes, 100 teachers, and 10,000 time slots.

Usability: The user interface should be intuitive and easy to use, with clear instructions and prompts to guide users through the timetable creation process.

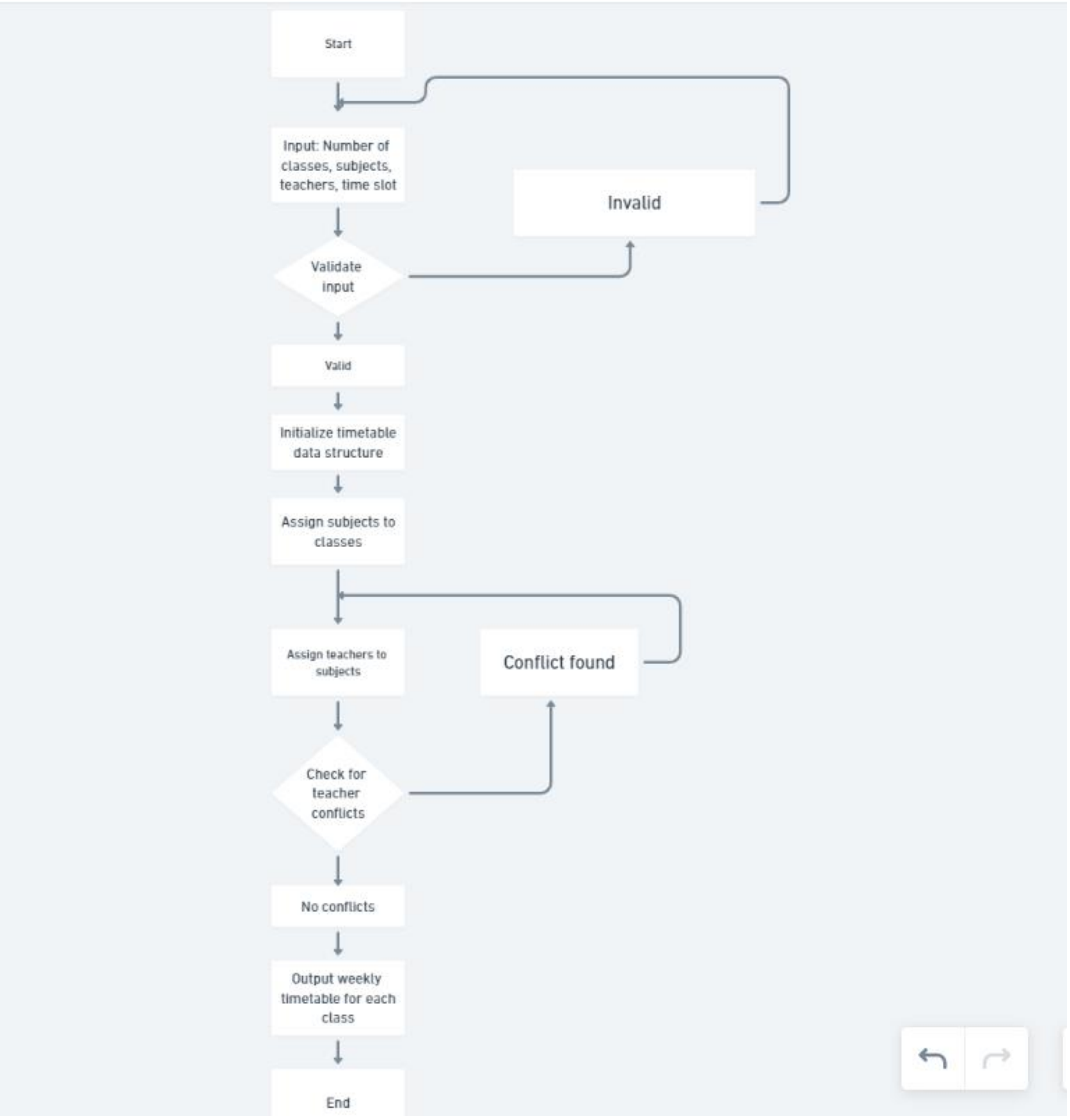
Scalability: The system should be able to handle increasing numbers of users, classes, teachers, and time slots without significant performance degradation.

Security: User data, including schedules and personal information, should be protected through secure authentication (e.g., Google accounts) and data encryption.

Maintainability: The codebase should be modular and well-documented, allowing for easy updates and bug fixes.

Data Integrity: The system should ensure the accuracy and consistency of data, preventing double bookings and ensuring correct schedule generation.

FLOWCHART:


Start
Input: Number of classes, subjects, teachers, time slot
Validate input
Invalid
Valid
Initialize timetable data structure
Assign subjects to classes
Assign teachers to subjects
Conflict found
Check for teacher conflicts
No conflicts
Output weekly timetable for each class
End