Detailed Project Report

on

Automated Time-table scheduling for IIIT Dharwad

SUBMITTED BY

print("Hello World!");

NIHAL SINGH
24BCS085

NIRBHAY KUMAR
24BCS089

NISCHAY R GOWDA
24BCS090

V MARUTHI
24BCS159

Under the guidance of

Dr. Vivekraj V K

Assistant Professor, Department of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD

September 30, 2025

Contents

List of Figures

List of Tables

1 Introduction

Creating and maintaining a timetable at IIIT Dharwad is a complex scheduling task that goes beyond simply listing class hours. Traditional formats such as spreadsheets, PDFs, and static images, while informative, are rigid and fail to adapt to real-time changes. They lack cross-platform accessibility and do not provide features like instant updates or reminders, leaving faculty and students vulnerable to missed classes, scheduling conflicts, and inefficient time management.

Manual timetable creation is labor-intensive and prone to errors. Factors such as course structures, lab sessions, classroom allocation, faculty availability, and adequate break times must be considered simultaneously. As the number of students and courses grows, these manual methods become increasingly time-consuming and impractical in a fast-paced academic environment.

To address these issues, the **Automated Time-Table Scheduling System** aims to provide a dynamic, intelligent, and scalable alternative to traditional methods. By incorporating advanced scheduling algorithms, real-time updates, and modern technologies, this system ensures optimal resource utilization, minimal conflicts, and seamless accessibility for students and faculty alike.

1.1 Objectives

The proposed system is designed to:

- Generate conflict-free, optimized timetables that respect academic and administrative constraints.
- Provide real-time updates and instant notifications, eliminating the need for manual adjustments.
- Offer cross-platform access, including synchronization with Google Calendar for mobile reminders.
- Enable dynamic rescheduling in case of faculty absence, public holidays, or weather-related disruptions.

1.2 System Features and Constraints

The system incorporates the following functionalities and design considerations:

- **Availability Tracking:** Professors, classrooms, laboratories, lab assistants, and enrolled students.
- **User Inputs:**

  - **Course Code:** e.g., CS101
  - **Course Name:** e.g., Data Structures and Algorithms
  - **LTPSC Structure:** e.g., 3-1-2-0-4 (Lectures-Tutorials-Practicals-Self-study-Credits)
  - **Instructor:** e.g., Dr. A. Kumar
  - **Course Offered To:** e.g., B.Tech CSE, Semester 3

- Allocate dedicated study hours to provide sufficient preparation time between lectures.
- Avoid continuous classes for professors by ensuring a minimum 3-hour gap between two sessions.
- Restrict one lecture/tutorial per day per course to promote balanced learning.
- Schedule lunch breaks within the mess timing to avoid clashes.
- Provide **color-coded timetables** for quick visual recognition:
  - Green for Core Courses
  - Blue for Labs
  - Orange for Electives
- Group labs on the same day as their corresponding lecture to enhance retention and application.
- Support **dynamic rescheduling** for half-semester courses to avoid overlap with full-semester ones.
- Ensure elective and minor courses share common slots across batches to prevent conflicts.
- **Vision-based automation:** Detect empty classrooms via CCTV for instant reallocation.
- **QR code integration:** Students can scan to view their updated personal timetable after any change.
- **AI-powered suggestions:** Recommend optimal lab batches based on attendance patterns.
- **Weather-based rescheduling:** Automatically shift physical classes to online mode during heavy rains.

2 Existing System

Present Manual Approach to Timetable Scheduling at IIIT Dharwad Timetable scheduling at IIIT Dharwad is currently carried out using a manual, slot-based system with the help of spreadsheets and static timetable templates. While this method provides a structured framework, it is still largely dependent on human coordination and adjustments, making it less adaptive to dynamic changes.

2.1 Understanding LTPSC

L – Lectures, **T** – Tutorials, **P** – Practicals/Labs, **S** – Seminars, **C** – Credits.
Example: `3-0-2-0-4` = 3 hours lecture, 0 tutorials, 2 labs, 0 seminars, 4 credits.

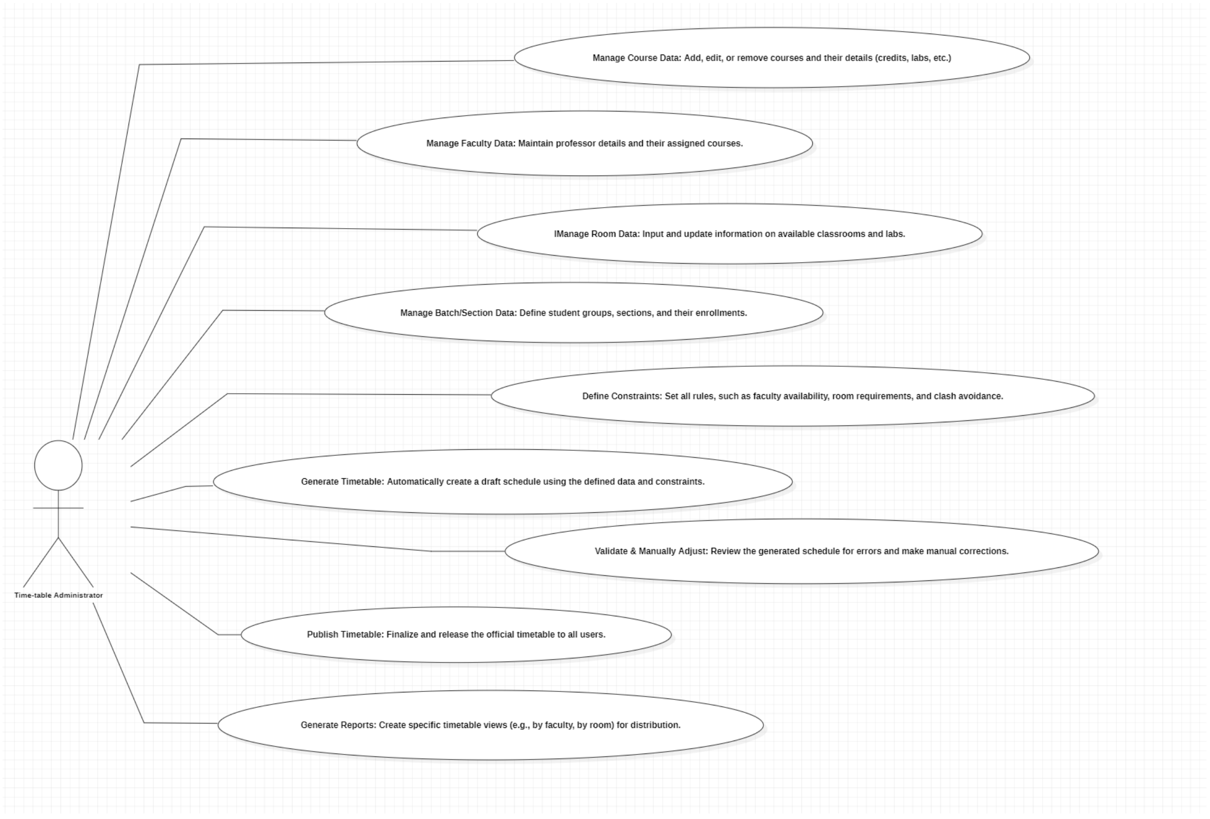

Manage Course Data: Add, edit, or remove courses and their details (credits, labs, etc.)
Manage Faculty Data: Maintain professor details and their assigned courses.
IManage Room Data: Input and update information on available classrooms and labs.
Manage Batch/Section Data: Define student groups, sections, and their enrollments.
Define Constraints: Set all rules, such as faculty availability, room requirements, and clash avoidance.
Generate Timetable: Automatically create a draft schedule using the defined data and constraints.
Validate & Manually Adjust: Review the generated schedule for errors and make manual corrections.
Publish Timetable: Finalize and release the official timetable to all users.
Generate Reports: Create specific timetable views (e.g., by faculty, by room) for distribution.
Time-table Administrator

Figure 1: Use-Case Diagram 1.1

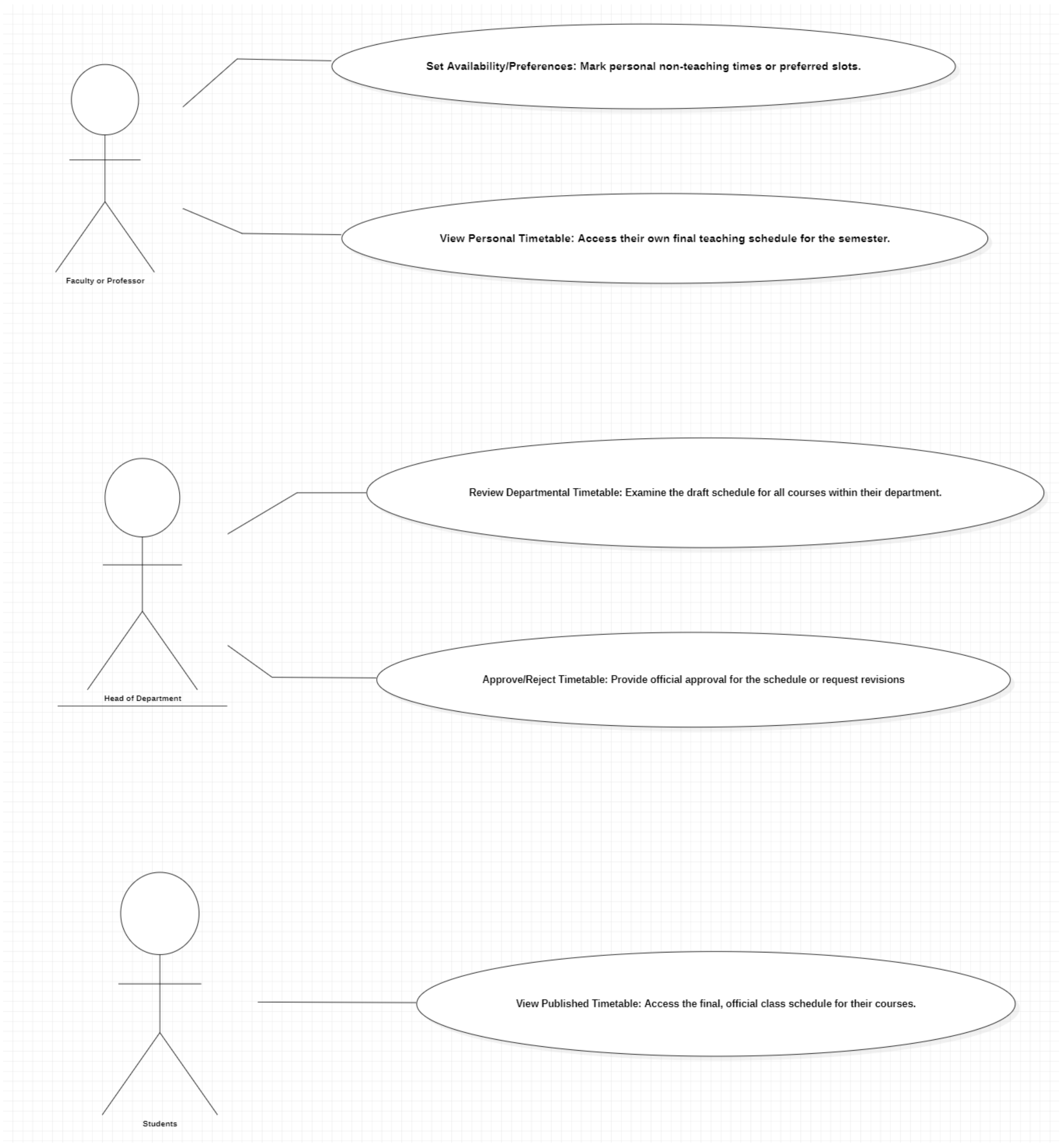

Set Availability/Preferences: Mark personal non-teaching times or preferred slots.
View Personal Timetable: Access their own final teaching schedule for the semester.
Faculty or Professor
Review Departmental Timetable: Examine the draft schedule for all courses within their department.
Approve/Reject Timetable: Provide official approval for the schedule or request revisions
Head of Department
View Published Timetable: Access the final, official class schedule for their courses.
Students

Figure 2: Use-Case Diagram 1.2

2.2 Current timetable

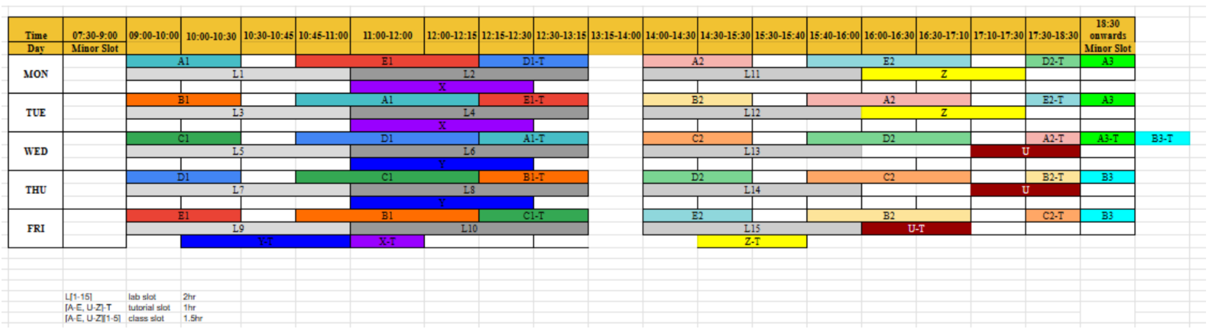
At most institutions, timetables are created manually using spreadsheets, shared documents, or emails. Department heads, faculty, and admin staff coordinate to assign courses, time slots, and classrooms.

The process usually involves:

- Listing courses and faculty assignments.
- Checking faculty teaching loads and preferences.

- Allocating classrooms based on class size.
- Distributing lectures, tutorials, and labs into weekly slots.

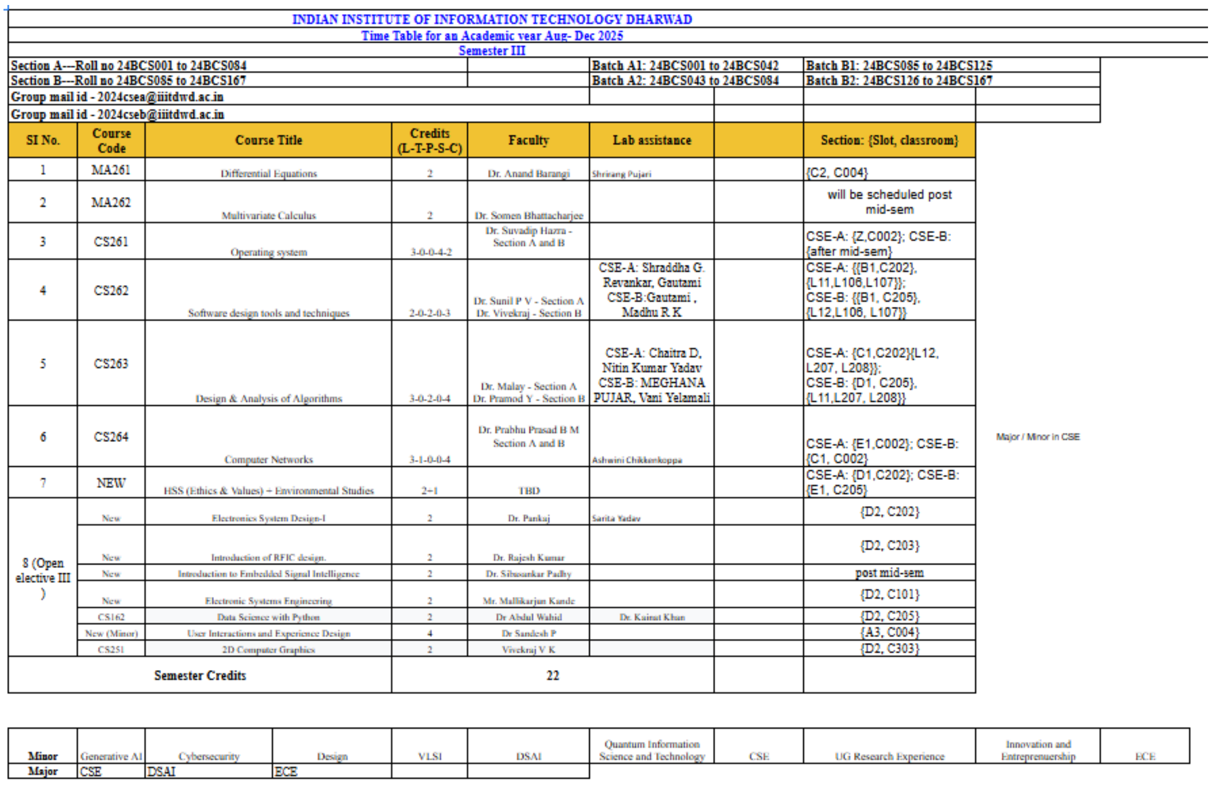
The final timetable is shared as a static PDF or print copy. Any change requires manual edits and redistribution.



Time / Day	07:30-9:00 Minor Slot	09:00-10:00	10:00-10:30	10:30-10:45	10:45-11:00	11:00-12:00	12:00-12:15	12:15-12:30	12:30-13:15	13:15-14:00	14:00-14:30	14:30-15:30	15:30-15:40	15:40-16:00	16:00-16:30	16:30-17:10	17:10-17:30	17:30-18:30	18:30 onwards Minor Slot	
MON		A1			E1			D1-T			A2			E2				D2-T	A3	
		L1				L2					L11				Z					
						X														
TUE		B1			A1			E1-T			B2			A2				E2-T	A3	
		L3				L4					L12				Z					
						X														
WED		C1			D1			A1-T			C2			D2				A2-T	A3-T	B3-T
		L5				L6					L13						U			
						Y														
THU		D1			C1			B1-T			D2			C2				B2-T	B3	
		L7				L8					L14						U			
						Y														
FRI		E1			B1			C1-T			E2			B2				C2-T	B3	
		L9				L10					L15				U-T					
			Y-T			X-T						Z-T								

L[1-15]	lab slot	2hr
[A-E, U-Z]-T	tutorial slot	1hr
[A-E, U-Z][1-5]	class slot	1.5hr

Figure 3: Existing Timetable



INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD
Time Table for an Academic year Aug- Dec 2025
Semester III

Section A---Roll no 24BCS001 to 24BCS084	Batch A1: 24BCS001 to 24BCS042	Batch B1: 24BCS085 to 24BCS125
Section B---Roll no 24BCS085 to 24BCS167	Batch A2: 24BCS043 to 24BCS084	Batch B2: 24BCS126 to 24BCS167
Group mail id - 2024csea@iiitdwd.ac.in		
Group mail id - 2024cseb@iiitdwd.ac.in		

SI No.	Course Code	Course Title	Credits (L-T-P-S-C)	Faculty	Lab assistance		Section: {Slot, classroom}	
1	MA261	Differential Equations	2	Dr. Anand Barangi	Shrirang Pujari		{C2, C004}	
2	MA262	Multivariate Calculus	2	Dr. Somen Bhattacharjee			will be scheduled post mid-sem	
3	CS261	Operating system	3-0-0-4-2	Dr. Suvadip Hazra - Section A and B			CSE-A: {Z,C002}; CSE-B: {after mid-sem}	
4	CS262	Software design tools and techniques	2-0-2-0-3	Dr. Sunil P V - Section A Dr. Vivekraj - Section B	CSE-A: Shraddha G. Revankar, Gautami CSE-B:Gautami , Madhu R K		CSE-A: {{B1,C202}, {L11,L106,L107}}; CSE-B: {{B1, C205}, {L12,L106, L107}}	
5	CS263	Design & Analysis of Algorithms	3-0-2-0-4	Dr. Malay - Section A Dr. Pramod Y - Section B	CSE-A: Chaitra D, Nitin Kumar Yadav CSE-B: MEGHANA PUJAR, Vani Yelamali		CSE-A: {C1,C202}{L12, L207, L208}}; CSE-B: {D1, C205}, {L11,L207, L208}}	
6	CS264	Computer Networks	3-1-0-0-4	Dr. Prabhu Prasad B M Section A and B	Ashwini Chikkenkoppa		CSE-A: {E1,C002}; CSE-B: {C1, C002}	Major / Minor in CSE
7	NEW	HSS (Ethics & Values) + Environmental Studies	2+1	TBD			CSE-A: {D1,C202}; CSE-B: {E1, C205}	
8 (Open elective III)	New	Electronics System Design-I	2	Dr. Pankaj	Sarita Yadav		{D2, C202}	
	New	Introduction of RFIC design.	2	Dr. Rajesh Kumar			{D2, C203}	
	New	Introduction to Embedded Signal Intelligence	2	Dr. Sibsankar Padhy			post mid-sem	
	New	Electronic Systems Engineering	2	Mr. Mallikarjun Kande			{D2, C101}	
	CS162	Data Science with Python	2	Dr Abdul Wahid	Dr. Kainat Khan		{D2, C205}	
	New (Minor)	User Interactions and Experience Design	4	Dr Sandesh P			{A3, C004}	
	CS251	2D Computer Graphics	2	Vivekraj V K			{D2, C303}	
Semester Credits			22					

Minor	Generative AI	Cybersecurity	Design	VLSI	DSAI	Quantum Information Science and Technology	CSE	UG Research Experience	Innovation and Entrepreneurship	ECE
Major	CSE	DSAI	ECE							

Figure 4: Timetable Courses

2.3 Comparison of Timetable Examples

The two provided timetables represent schedules for the same academic program but with noticeable differences in timing, course allocation, and facilities. The key observations are:

- **Timing Differences:**
  - The first timetable starts with a 9:00 AM session, whereas the second timetable begins earlier at 7:30 AM.
  - The first timetable has defined morning and lunch breaks, whereas the second timetable's breaks are less explicitly defined but have minor slots.
  - The second timetable has minor slots indicated between 18:00-18:45 and 18:45-19:30, which are not present in the first timetable.
- **Course and Slot Allocations:**
  - Both timetables include similar courses like IET, Economics, Open Electives, and lab sessions, but their timings and batch distributions vary.
  - The second timetable appears to have additional course slot divisions with detailed abbreviations (e.g., LI-1H, LU-2T), possibly indicating lab/tutorial hours more precisely.
  - The first timetable shows lab sessions labeled with specific room numbers (e.g., L207, L208), while the second timetable uses codes like "LI", "LU", etc.
- **Facilities and Room Usage:**
  - The first timetable assigns specific classrooms for labs and tutorials, facilitating better classroom management.
  - The second timetable indicates multiple minor slots possibly for extra classes or remedial sessions.
  - Both timetables provide batch-wise distinctions (Batch B1, B2) but their scheduling differs, reflecting changes in classroom or batch management.
- **Break and Minor Slot Management:**
  - The first timetable has designated break times, with a clear lunch break and morning break.
  - The second timetable includes multiple minor slots throughout the day which may indicate more flexibility or additional session types.

The changes between the two timetables primarily reflect optimization in timing, session management, and finer classroom assignment details. The second timetable introduces earlier start times and minor slots possibly for enhanced academic support, while the first timetable offers a more traditional schedule with defined breaks and batch-wise lab allocation.

2.4 Limitations of current timetable

From the existing figures:

- Fixed grids (Figure 1) are clear but hard to reschedule.
- Course-slot mapping (Figure 2) is helpful but doesn't prevent clashes.

Other problems:

- Manual checks cause scheduling conflicts.
- Overlapping classroom bookings.
- Poorly planned breaks.
- No calendar integration.
- Difficulty managing labs and electives without overlaps.
- Accessibility issues — no live updates.

Advantages vs Disadvantages

#	Advantages	Disadvantages
1	Human oversight and flexibility.	Time-consuming and error-prone.
2	Low-tech — simple tools suffice.	Hard to update and redistribute.
3	Customization per department.	No live synchronization or scalability.

Summary

The manual system is flexible and simple but inefficient and prone to errors. Moving to an automated system with optimization algorithms can resolve most issues while keeping human control where needed.

3 Requirements Modeling

3.1 List of Requirements

Table 2: List of requirements table

No.	Requirement Description
1	**One Lecture Slot Per Course Per Day:** A single course must have only one lecture scheduled per day to avoid redundancy and cognitive overload.
2	**Refresh Breaks:** Allocate a 10-minute break after each class.
3	**Fixed Lunch Break:** Schedule a fixed lunch break aligned with mess timings.
4	**Color-Coded Timetable:** Use distinct colors to represent different courses.
5	**Lab and Lecture on Same Day:** Schedule the lab session on the same day as the lecture.
6	**Google Calendar Integration:** Integrate timetable with Google Calendar.
7	**Half-Semester Course Handling:** Schedule half-semester courses in their designated half.
8	**Common Slot for Electives:** Assign common elective slots across departments.
9	**Resource Availability Tracking:** Ensure faculty, classrooms, and facilities are available before scheduling.
10	**Course Detail Input:** Allow input for code, title, LTPSC structure, instructor, semester, and enrollment count.
11	**Self-Study Hours:** Allocate specific self-study slots.
12	**Faculty Gap Between Classes:** Maintain at least a three-hour gap between a faculty member's classes.

4 Exam Scheduling & Seating (Add-on Requirement)

Goal: Since the class timetable is being automated, the software must also automate the *exam timetable*, *classroom assignment*, and *seating arrangement*.

Core Requirements

Exam Timetable Generation

Produce a conflict-free exam schedule: no student should have two exams at the same time; respect exam duration and buffer times between slots.

Classroom Assignment

Allocate classrooms for exams based on seating capacity and accessibility.

Seating Arrangement

Constraint: No two students writing the same exam should occupy the same desk. For dual benches, enforce alternate seating or mix with other courses.

Exportable Outputs

Generate per-room seating charts, per-student hall tickets (with room/seat), invigilator duty lists, and attendance sheets. Export in PDF/CSV formats.

Seating Policy Details

- **Desk Rule:** No two students of the same exam at the same desk.
- **Interleaving:** Prefer mixing different exams in one room when capacity requires.
- **Mapping:** Seat assignment based on roll numbers to maintain reproducibility.

Acceptance Criteria

- Each student has at most one exam per time slot.
- Room capacity never exceeded.
- Seating plan always satisfies Desk Rule.

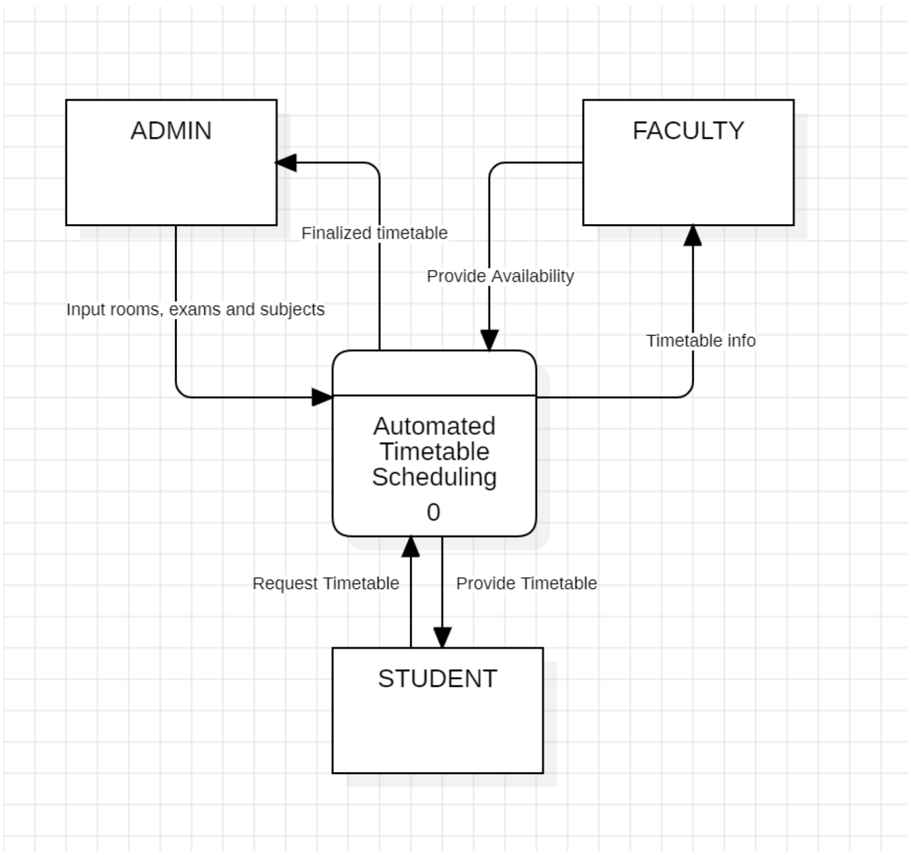


Figure 5: Level 0 DFD

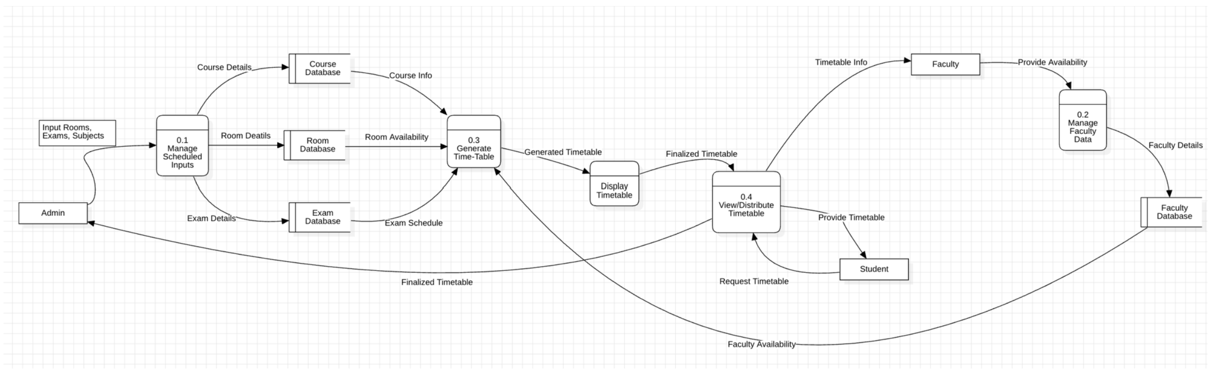


Figure 6: Level 1 DFD

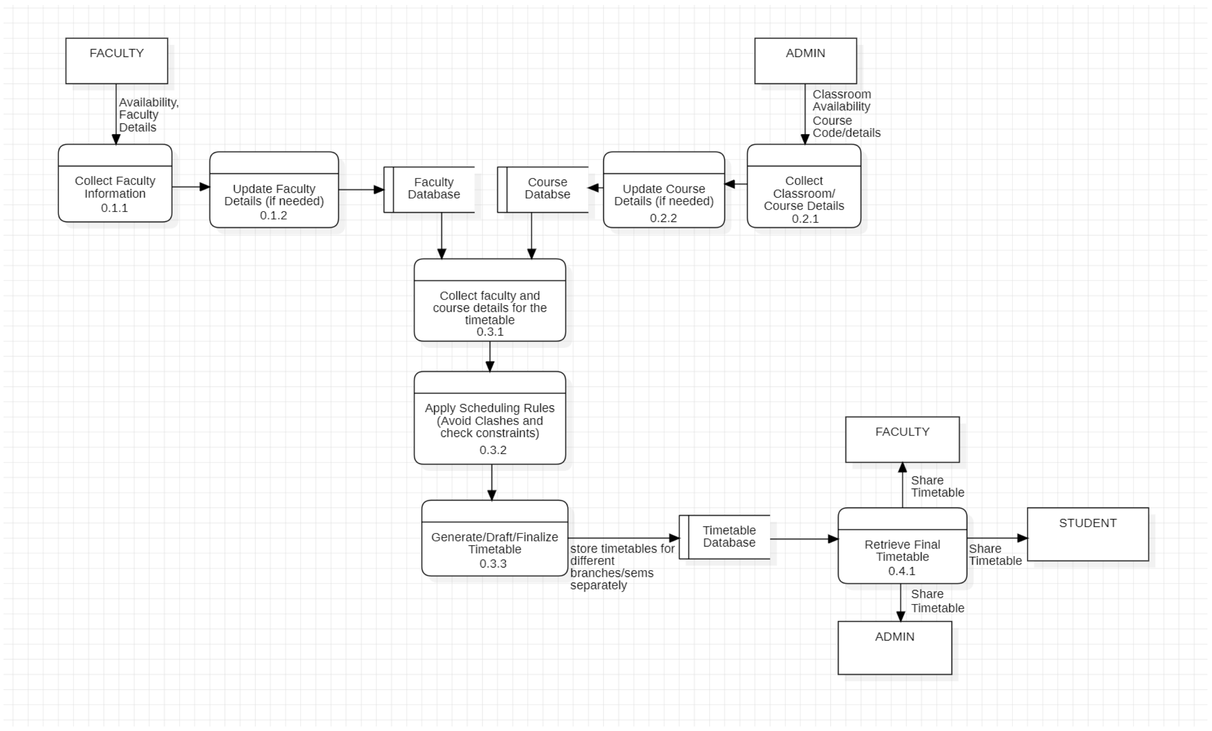

FACULTY
Availability, Faculty Details
Collect Faculty Information 0.1.1
Update Faculty Details (if needed) 0.1.2
Faculty Database
Course Databse
Update Course Details (if needed) 0.2.2
Collect Classroom/ Course Details 0.2.1
ADMIN
Classroom Availability Course Code/details
Collect faculty and course details for the timetable 0.3.1
Apply Scheduling Rules (Avoid Clashes and check constraints) 0.3.2
Generate/Draft/Finalize Timetable 0.3.3
store timetables for different branches/sems separately
Timetable Database
Retrieve Final Timetable 0.4.1
FACULTY
Share Timetable
Share Timetable
STUDENT
Share Timetable
ADMIN

Figure 7: Level 2 DFD

4.1 Data Dictionary

- Course-Details: {Course-ID: string, Course-Name: string, Credits: integer, Department: string, Semester: integer, Duration: integer}
- Faculty-Details: {Faculty-ID: string, Faculty-Name: string, Dept: string, Availability: timeslot, Assigned-Courses: list}
- Student-Registration: {Student-ID: string, Name: string, Program: string, Semester: integer, Registered-Courses: list}

- Resource-Availability: {Resource-ID: string, Type: string, Capacity: integer, Availability: timeslot}
- Institutional-Constraints: {Constraint-ID: string, Constraint-Description: string}
- Scheduling-Rules: {Rule-ID: string, Rule-Type: string, Priority: integer}
- Draft-Timetable: {Timetable-ID: string, Course-Schedule: list, Faculty-Allocation: list, Room-Allocation: list}
- Validation-Feedback: {Feedback-ID: string, From: Faculty/Student, Conflict-Details: string, Status: string}
- Finalized-Timetable: {Draft-Timetable + Approval-Status: string}
- Faculty-Timetable: {Faculty-ID: string, Assigned-Courses: list, Schedule: list}
- Student-Timetable: {Student-ID: string, Registered-Courses: list, Schedule: list}
- Generated-Timetable: {Draft: Draft-Timetable, Final: Finalized-Timetable}
- Calendar-Event: {Event-ID: string, Course: string, Faculty: string, Room: string, Time: datetime}

5 High-Level & Low-Level Design

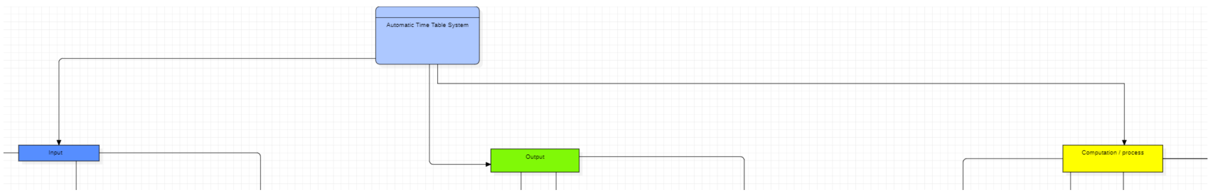


Figure 8: High-Level Design 1.1

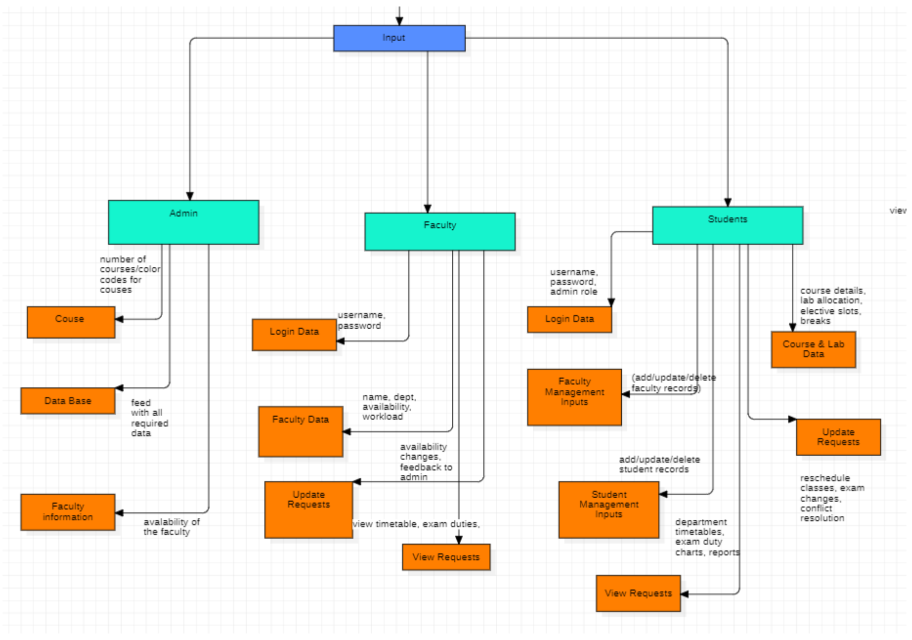
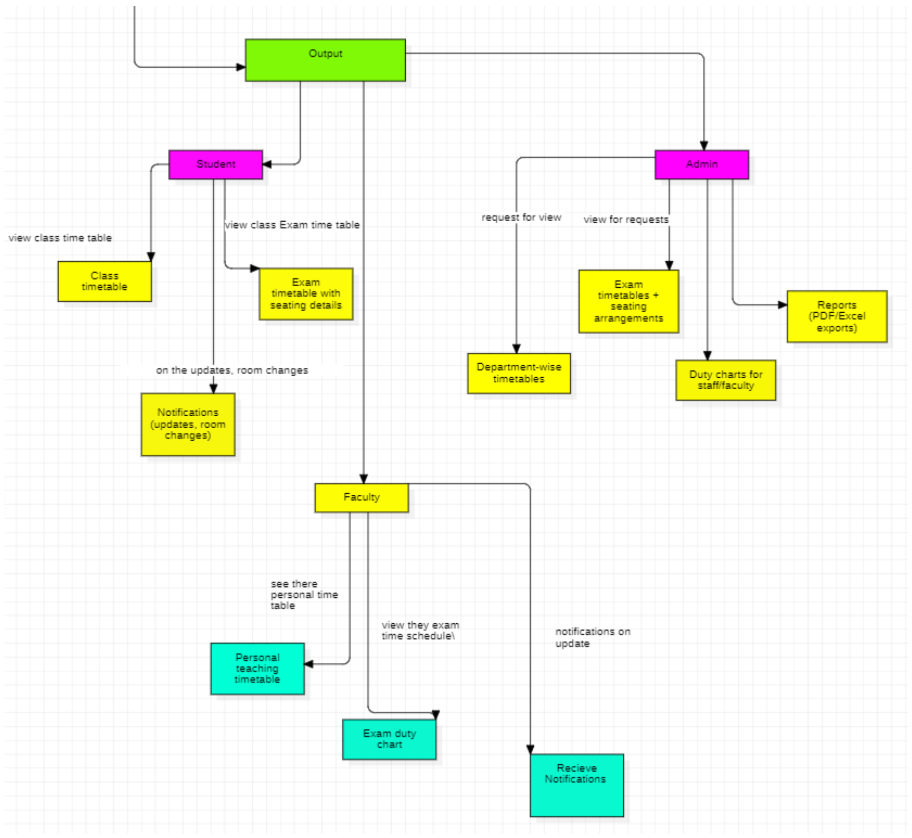
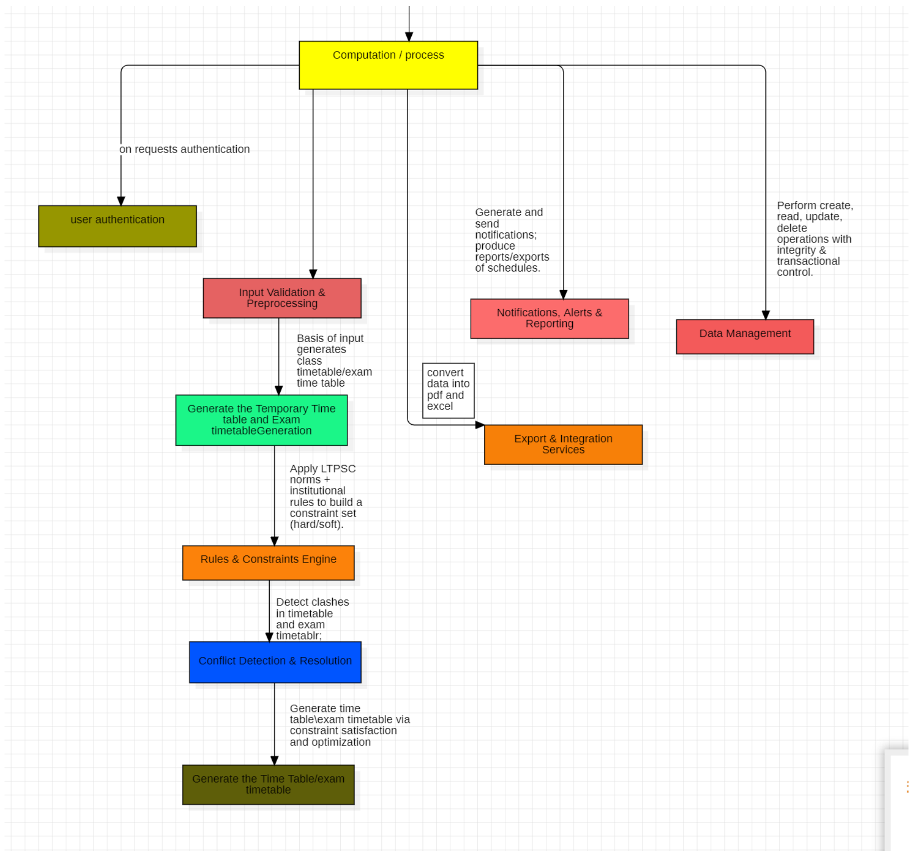

Input
Admin
Faculty
Students
viev
number of courses/color codes for couses
Couse
Data Base
feed with all required data
Faculty information
avalability of the faculty
Login Data
username, password
Faculty Data
name, dept, availability, workload
availability changes, feedback to admin
Update Requests
view timetable, exam duties,
View Requests
username, password, admin role
Login Data
Faculty Management Inputs
(add/update/delete faculty records)
add/update/delete student records
Student Management Inputs
department timetables, exam duty charts, reports
View Requests
course details, lab allocation, elective slots, breaks
Course & Lab Data
Update Requests
reschedule classes, exam changes, conflict resolution

Figure 9: High-Level Design 1.2 (Input)



Output
Student
Admin
view class time table
view class Exam time table
Class timetable
Exam timetable with seating details
on the updates, room changes
Notifications (updates, room changes)
request for view
view for requests
Exam timetables + seating arrangements
Reports (PDF/Excel exports)
Department-wise timetables
Duty charts for staff/faculty
Faculty
see there personal time table
view they exam time schedule\
notifications on update
Personal teaching timetable
Exam duty chart
Recieve Notifications

Figure 10: High-Level Diagram 1.3 (Output)



Computation / process
on requests authentication
user authentication
Input Validation & Preprocessing
Generate and send notifications; produce reports/exports of schedules.
Notifications, Alerts & Reporting
Perform create, read, update, delete operations with integrity & transactional control.
Data Management
Basis of input generates class timetable/exam time table
Generate the Temporary Time table and Exam timetableGeneration
convert data into pdf and excel
Export & Integration Services
Apply LTPSC norms + institutional rules to build a constraint set (hard/soft).
Rules & Constraints Engine
Detect clashes in timetable and exam timetablr;
Conflict Detection & Resolution
Generate time table\exam timetable via constraint satisfaction and optimization
Generate the Time Table/exam timetable

Figure 11: High-Level Design 1.4 (Process)

Low-Level Design

Data Structures

```
struct Faculty {
    int facultyID;
    string name;
    vector<string> courses;          // Courses taught
    vector<string> availableSlots;   // Time slots available
};
```

Listing 1: Faculty Information

```
struct Course {
    string courseCode;
    string courseName;
    int credits;
    string facultyID;
    vector<string> preferredSlots;
};
```

Listing 2: Course Information

```
struct Room {
    string roomID;
    int capacity;
    bool isLab;
    vector<string> availableSlots;
};
```

Listing 3: Classroom/Lab Information

```
struct Exam {
    string courseCode;
    string examDate;
    string roomID;
};
```

Listing 4: Exam Information

```
struct TimeTableEntry {
    string day;
    string timeSlot;
    string courseCode;
    string facultyID;
    string roomID;
};
```

Listing 5: Timetable Entry

```
struct TimeTable {
    vector<TimeTableEntry> entries;
};
```

Listing 6: Timetable

Function Declarations and Descriptions

1. Admin Module

```
void inputSubjects(vector<Course>& courseList);
void inputExams(vector<Exam>& examList);
void inputRooms(vector<Room>& roomList);
void finalizeTimeTable(TimeTable& tt);
```

2. Faculty Module

```cpp
void provideAvailability(Faculty& f, vector<string> slots);
void viewFacultyTimeTable(Faculty f, TimeTable tt);
```

3. Student Module

```cpp
void requestTimeTable(string studentID);
void viewStudentTimeTable(string studentID, TimeTable tt);
```

4. Course Management Module

```cpp
void collectCourseDetails(Course& c);
void updateCourseDetails(Course& c, string newName, int newCredits);
```

5. Faculty Management Module

```cpp
void collectFacultyInfo(Faculty& f);
void updateFacultyDetails(Faculty& f, vector<string> newCourses);
```

6. Time-Table Generation Module

```cpp
void collectDetails(vector<Course> courses, vector<Faculty>
    faculties, vector<Room> rooms);
void applySchedulingRules(TimeTable& tt, vector<Course> courses,
    vector<Faculty> faculties, vector<Room> rooms);
void generateTimeTable(TimeTable& tt, vector<Course> courses, vector
    <Faculty> faculties, vector<Room> rooms, vector<Exam> exams);
bool validateTimeTable(TimeTable tt);
```

7. Database Management Module

```cpp
void saveCourseDB(vector<Course> courses);
vector<Course> loadCourseDB();

void saveFacultyDB(vector<Faculty> faculties);
vector<Faculty> loadFacultyDB();

```

```
void saveExamDB(vector<Exam> exams);
vector<Exam> loadExamDB();

void saveRoomDB(vector<Room> rooms);
vector<Room> loadRoomDB();

void saveTimeTableDB(TimeTable tt);
TimeTable loadTimeTableDB();
```

8. View/Retrieve Timetable Module

```
TimeTable retrieveFinalTimeTable();
void viewTimeTable(TimeTable tt, string userType, string userID);
```

6 Conclusion

A well-designed timetable is more than just a schedule—it is a strategic plan that balances teaching effectiveness, student convenience, and resource optimization. Implementing the above guidelines ensures that the timetable is fair, flexible, and future-ready.

By integrating modern features such as color coding and calendar synchronization, while respecting human factors like break times, meal schedules, and classroom capacity, institutions can significantly improve the learning environment.

Automating this process reduces human error and increases flexibility for last-minute changes, ensuring the timetable remains a living document—adaptable, accessible, and future-ready.

7 Coding/Implementation

Tech Stack

- Programming Language: C++ / Python (for scheduling algorithms and data structures)
- Database: SQLite / JSON-based lightweight storage
- Version Control: Git and GitHub
- Visualization: Matplotlib / Web UI for timetable display
- Export Formats: PDF, CSV

- Optional: Integration with Google Calendar API

  GiT repository link: https://github.com/FuriousYETI/Automatic-Time-Table-Generation-Hello-World-.git

8 References

The reference materials include current professor lectures and the existing timetable.