DETAILED PROJECT REPORT

on

AUTOMATED TIME-TABLE SCHEDULING FOR IIIT DHARWAD

Submitted by

Team: *SoftwareSages*

Nikunj Srivastav 24BCS087

Sudhanshu Baberwal 24BCS147

Thejas Gowda U M 24BCS157

Shaik Moiz 24BCS133

Under the guidance of

Vivekraj VK

Assistant Professor

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD

12/08/2025

Contents

List of Figures **iv**

List of Tables **v**

1 Introduction **1**

1.1 Constraints on the System 1

1.2 Available Infrastructure 2

2 Existing System **3**

2.1 Overview of the Timetable 3

2.2 Problems in the Existing System 6

2.3 Implications of the Current System 7

3 Requirements Modeling **8**

3.1 Functional Requirements 9

3.1.1 Timetable Generation 9

3.1.2 Student Personalization 9

3.1.3 Visual Interface and Usability 9

3.1.4 Conflict Management and Validation 10

3.2 Non-Functional Requirements 10

3.2.1 Performance Requirements 10

3.2.2 Usability and Interface Requirements 10

3.2.3 Reliability and Availability 10

3.2.4 Security Requirements 11

3.2.5 Interoperability and Integration 11

3.2.6 Maintainability and Scalability 11

4 Examination : Existing Model **12**

5 Existing Model : The Challenges **13**

5.1 Challenges faced by the Scheduler 13

5.2 Challenges faced by the Student 14

6 Existing Model : The Solution **15**

7 Requirements for Automating the Examination Timetable **16**

7.1 Functional Requirements 16

7.1.1 Course and Subject Data 16

7.1.2 Student Information Management 16

7.1.3 Faculty and Invigilator Availability 16

7.1.4 Examination Parameters 17

7.1.5 Room and Hall Management 17

7.1.6 Rules and Constraints Enforcement 17

7.1.7 System and User Interfaces 18

7.2 Non-Functional Requirements 18

7.2.1 Performance Requirements 18

7.2.2 Usability Requirements 18

7.2.3 Reliability and Availability 18

7.2.4 Scalability 19

7.2.5 Security Requirements 19

7.2.6 Compatibility and Integration 19

7.2.7 Maintainability and Modifiability 19

8 Software Design **20**

8.1 Data Flow Diagrams 20

8.1.1 Level 0: 20

8.1.2 Level 1: 21

8.1.3 Level 2: 22

8.2 Data Dictionary 23

8.2.1 Level 0: 23

8.2.2 Level 1 24

8.3 Low-Level Design 27

8.3.1 Data Structures 27

8.3.2 Function Declarations 29

8.4 Module Structure 31

9 Coding/Implementation **32**

10 Conclusion **33**

List of Figures

1 The front page providing the slots 4
2 The Last-SEM time-table 4
3 The section for the Alphanumeric codes for IIIrd SEM 5
4 Use-Case Diagram for Automated Time-Table 8
5 Level - 0 of the Automated Time-Table DFD 20
6 Level - 1 of the Automated Time-Table DFD 21
7 Level - 2 of the Automated Time-Table DFD 22
8 Module Structure 31

List of Tables

1 Level 0 Data Dictionary 23
2 Level 1 Faculty Record 24
3 Level 1 Course Record 24
4 Level 1 Exam Schedule 24
5 Level 1 Room Availability Record 25
6 Level 1 Mess Timing Record 25
7 Level 1 Time-Table Record 25
8 Level 1 Database 26

1 Introduction

In academic institutions, especially at the college level, the timetable serves as a foundational framework that dictates the schedule of lectures, practical sessions, and academic activities. Despite its critical role, timetable systems are often complex, manually designed, and prone to inefficiencies. The traditional method of timetable management involves static spreadsheets or documents that lack adaptability and often demand significant cognitive effort from students and faculty alike to interpret and follow. Much like navigating a city without a map, understanding such disorganized schedules can be time-consuming, error-prone, and frustrating.

In the context of a fast-paced academic environment, where clarity and efficiency are paramount, the need for an automated timetable system becomes apparent. Automation not only streamlines the process of generating and updating schedules but also enhances accessibility, consistency, and user experience. With advancements in computational tools and algorithms, it is now feasible to design systems that can dynamically create and manage timetables based on a set of academic constraints and user preferences.

This document aims to explore the automation of the college timetable by analyzing the limitations of the existing system, modeling the essential requirements for an automated solution, and proposing a structured approach to its implementation. The goal is to reduce the overhead associated with manually decoding schedules and to introduce a system that is both intuitive and scalable. In essence, automating the timetable is akin to replacing a static road map with a GPS system—responsive, intelligent, and efficient.

1.1 Constraints on the System

When creating an automated timetable system, the following constraints must be considered to ensure that the generated schedule is practical and efficient:

- Instructor Availability: Each instructor has specific workdays and hours. The system shouldn't assign them when they're not available.

- Limitations on Classrooms and Resources: A limited number of class- rooms and labs are available. The system must prevent reservations from overlapping.
- Course Requirements: A number of courses require specific equipment, like computer labs, which need to be allocated appropriately.
- Overlapping of Sessions: Neither a teacher nor a student may be enrolled in two classes at the same time.
- Working Hours and Breaks: Lectures should be planned around regular business hours, accounting for intermissions and lunch breaks.
- Balanced Workload: The system should distribute lectures evenly through- out the week to avoid overloading instructors or students on a single day.
- Priority Rules: Certain courses (like core subjects) may need to be given more weight than electives when allocating slots.
- Semester/Batch Dependencies: Students should not be enrolled in classes that conflict with required subjects during the same semester.

1.2 Available Infrastructure

The automated timetable system will operate within the existing institutional infrastructure:

- Classrooms should be equipped with enough seating and basic teaching facilities.
- Computer and electronic labs are required for certain courses.
- List of professors along with their preferred teaching slots.
- Google calendar to be integrated with the timetable for easy access.
- Institutional servers and computers that can host the timetable generation system.

2 Existing System

The current timetable system employed by the institution is a manually designed and visually dense framework that attempts to accommodate a wide range of academic activities across multiple disciplines. While the system technically fulfills its functional purpose — mapping courses to time slots, classrooms, and instructors — it presents significant usability and comprehension challenges to the students. This section provides a detailed overview of the existing system's structure, the difficulties associated with it, and the broader implications of its shortcomings.

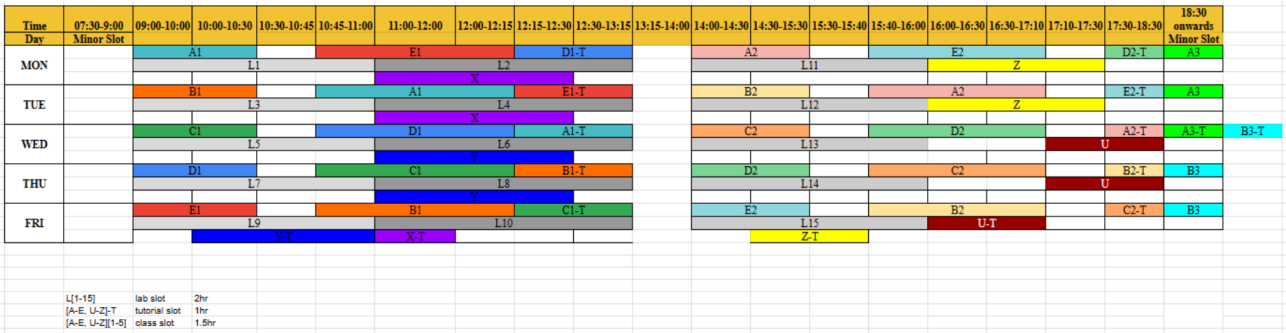
2.1 Overview of the Timetable

The present timetable consists of two main components:

- Course Information Table: This table outlines essential details of each course, including:
  - Course Code and Title (e.g., MA261 - Differential Equations)
  - Credit Structure (L-T-P-S-C format — Lecture-Tutorial-Practical-Self Study-Credits)
  - Assigned Faculty
  - Lab Assistance (if applicable)
  - Section-wise Slot and Classroom Mapping (e.g., C2, C004)

  The academic load includes core subjects like Differential Equations, Operating Systems, and Design and Analysis of Algorithms, along with open electives and minor courses. Each subject is allocated specific slots (A1, B1, C1, etc.) and laboratories (L1 to L15), with classrooms changing depending on the section (A or B).

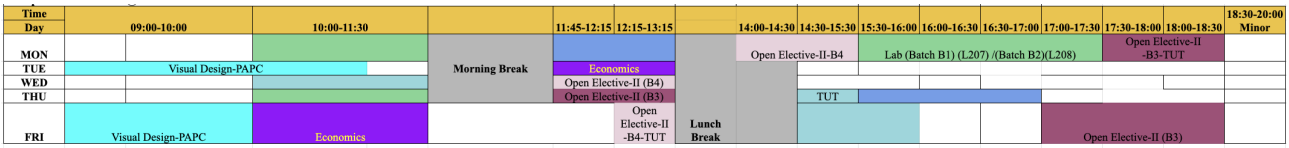
- Weekly Slot-Based Schedule (Time Table Grid):
  A second visual representation provides a weekly calendar spanning Monday to Friday, with time slots beginning at 07:30 AM and extending beyond 06:30 PM. Courses are mapped to slot identifiers (e.g., A1, L2, Z-T) rather than course names, which requires manual cross-referencing with the course information table.



Time	07:30-9:00	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
Day	Minor Slot																		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

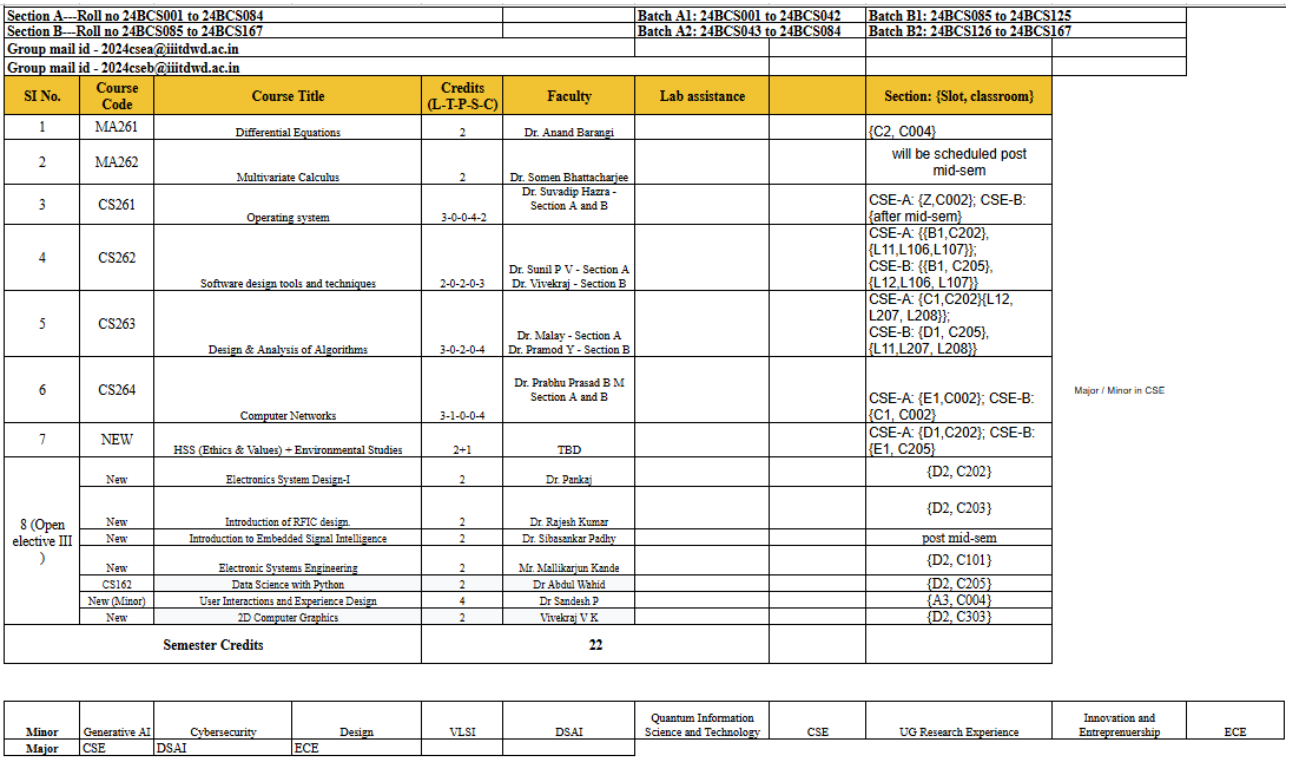
Figure 1. The front page providing the slots



Time	09:00-10:00	10:00-11:30		11:45-12:15	12:15-13:15		14:00-14:30	14:30-15:30	15:30-16:00	16:00-16:30	16:30-17:00	17:00-17:30	17:30-18:00	18:00-18:30	18:30-20:00
Day															Minor
MON			Morning Break			Lunch Break	Open Elective-II-B4		Lab (Batch B1) (L207) /(Batch B2)(L208)				Open Elective-II -B3-TUT		
TUE	Visual Design-PAPC			Economics											
WED				Open Elective-II (B4)											
THU				Open Elective-II (B3)				TUT							
FRI	Visual Design-PAPC	Economics			Open Elective-II -B4-TUT							Open Elective-II (B3)			

SI No.	Course Code	Course Title	Credits (L-T-P-S-C)	Faculty	Lab assistance	Room No.
3		Linear Algebra	3-1-0-0-2	Dr. Somen Bhattacharjee		C203
4	CS163	Data Structures & Algorithms	3-0-2-0-4	Dr. Suvadip Hazra	Sahana E Punagin Chaitra D	C203
5	HS204 / HS153	Economics/ Innovation and Entrepreneurship	2-1-0-0-3	Dr. Anushree Kini/ Dr.Deepak k T		C202/ C004
6	CS165	Mathematical foundations of Computing	3-0-0-0-3	Dr. Animesh Roy		C203
7		Data Science with Python	3-1-0-0-2	Dr. Abdul Wahid		C002
		Introduction to Digital VLSI Design	3-1-0-0-2	Dr Prakash Pawar		C203
	B3	Industry Insights Program Part 1	3-0-0-0-1	Mr. Ram Subramanian & Mr. Sasi Kumar Sundara Rajan		
		Linux for Engineers	3-1-0-0-2	Dr. Shirshendu L		C203
		Introduction to Holistic Personality Development	3-0-0-0-1	Prof. Chachadi and Dr. Chandrika K		C203
	B4	Photography101	3-0-0-0-1	Dr Prabhu Prasad BM		C204
	New	Visual Design-PAPC	3-1-0-0-2	Dr. Sandesh Phalke		C304

Figure 2. The Last-SEM time-table



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										
Sl 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			{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: {{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: {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			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			{D2, C202}			
	New	Introduction of RFIC design.	2	Dr. Rajesh Kumar			{D2, C203}			
	New	Introduction to Embedded Signal Intelligence	2	Dr. Sibasankar Padhy			post mid-sem			
	New	Electronic Systems Engineering	2	Mr. Mallikarjun Kande			{D2, C101}			
	CS162	Data Science with Python	2	Dr Abdul Wahid			{D2, C205}			
	New (Minor)	User Interactions and Experience Design	4	Dr Sandesh P			{A3, C004}			
	New	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 3. The section for the Alphanumeric codes for IIIrd SEM

2.2 Problems in the Existing System

Despite the structured presentation, the existing timetable poses several operational and cognitive difficulties, particularly from the students' perspective. The following outlines the major issues observed:

- Complex Cross-Referencing:
  Students must frequently switch between the two tables: the timetable grid and the course mapping table, to identify which course is scheduled at any given time.
  Slots such as "C1" or "L12" do not inherently indicate the subject, instructor, or location. These details must be manually interpreted by matching the slot with the corresponding entry in the course table.
  For instance, to know what "C1" on Thursday at 10:00 AM represents, students must refer back to the course list and locate all subjects that use "C1" (e.g., CS263 and CS264 for Section A and B respectively).

- Non-Uniform Slot Utilization:
  Courses are not Utilized/distributed throughout the day. Some days are heavily loaded (e.g., Wednesday and Thursday), while others like Monday have lighter schedules.
  Certain time slots (e.g., 07:30–09:00 and post-06:30 PM) are dedicated to minor courses or elective modules but are inconsistently populated across the week, adding to scheduling confusion.

- Lack of Personalization:
  The timetable is designed generically for entire sections (A and B), with no student-level customization.
  Students taking open electives, minor courses, or interdisciplinary subjects must manually identify and filter their relevant entries from the pool of possible slots.

- Visual Overload and Color Clutter:
  While colors are used to differentiate slots, the sheer variety of colors used for A1–E2, L1–L15, and Z/U/X slots leads to visual fatigue.
  There is no legend or guide to explain the meaning of the color codes, leaving interpretation up to the students.

- Scattered Course Information:
  Information such as course credits, lab assistance, and faculty details are buried within the course table and are not accessible when viewing the slot-based schedule alone.

For example, students cannot immediately know the number of contact hours for CS263 unless they inspect the "Credits (L-T-P-S-C)" column.

2.3 Implications of the Current System

The complexity of the current timetable has several negative downstream effects:

- Cognitive Load: Students invest significant time and mental effort decoding their daily and weekly schedules, often leading to errors or missed classes.
- Inefficient Time Management: Without a clear, consolidated view of their personal schedule, students struggle to plan study sessions, extracurriculars, or breaks.
- Administrative Overhead: Faculty and staff must manually resolve conflicts or handle queries regarding slot clashes and classroom confusion.
- Low Scalability: As class sizes increase or new electives/minors are introduced, the manual nature of the timetable system becomes increasingly unsustainable.

3 Requirements Modeling

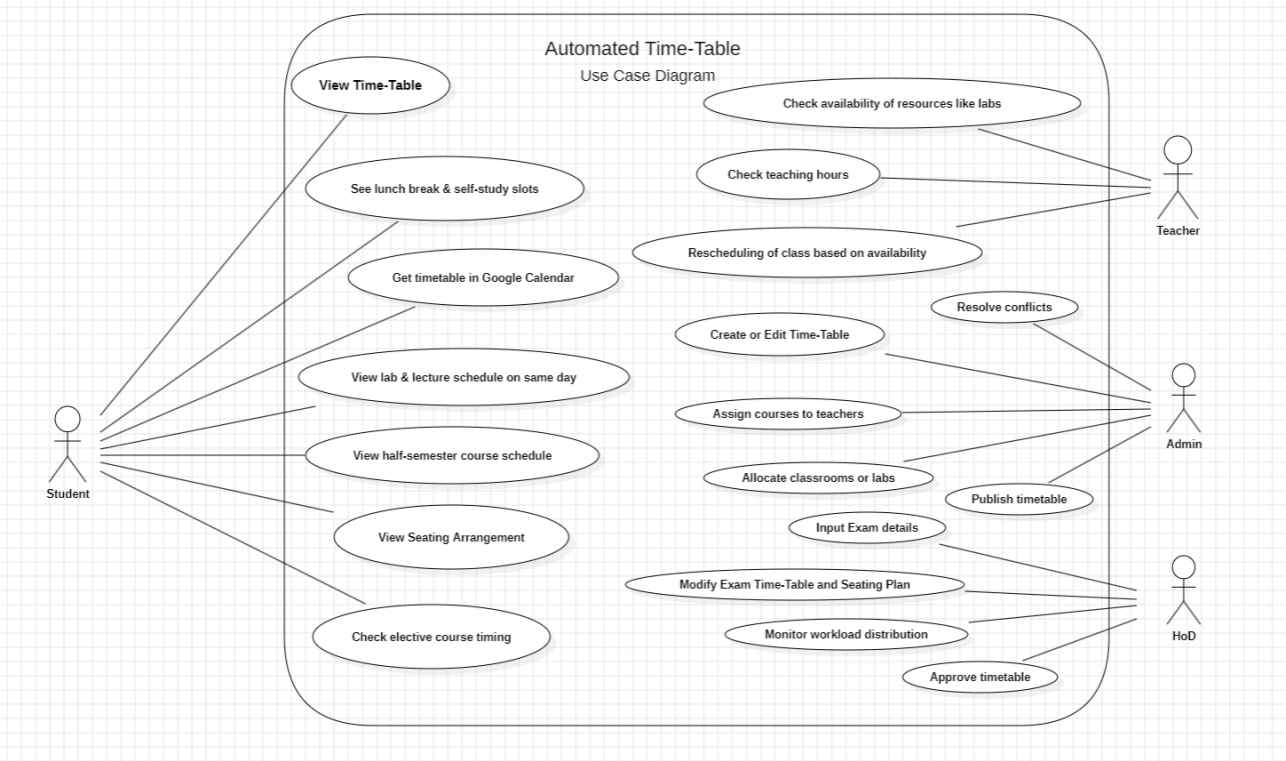


Figure 4. Use-Case Diagram for Automated Time-Table

As a student in a highly competitive academic environment, efficient time management, clarity of schedule, and adaptability to academic changes are crucial. An automated timetable system must address the inefficiencies of the current static system and offer features that improve accessibility, personalization, and reliability. The following are the key functional and non-functional requirements that the proposed automated system must fulfill:

3.1 Functional Requirements

3.1.1 Timetable Generation

- The system shall automatically assign each registered course to its respective time slot and classroom, eliminating manual mapping.
- The system shall integrate lab sessions clearly within the weekly timetable, showing lab room numbers and associated course names.
- The system shall support updates for courses that start mid-semester or post-semester, ensuring accurate timetables throughout the academic period.

3.1.2 Student Personalization

- The system shall generate a personalized timetable for each student based on their registered courses, including core, elective, and minor subjects.
- The system shall highlight free time slots in the student timetable to assist with self-study, meetings, or extracurricular planning.
- The system shall allow students to export their personalized timetables as PDF or image formats for offline viewing or printing.
- The system shall provide optional synchronization with calendar apps (Google Calendar, Outlook) and support alerts for upcoming sessions.

3.1.3 Visual Interface and Usability

- The system shall display a weekly, interactive, and color-coded timetable interface, including course names, faculty, room numbers, and session type (lecture/lab).
- The system shall allow users to view faculty assignments and classroom details linked to each course.

3.1.4 Conflict Management and Validation

- The system shall detect and flag scheduling conflicts (e.g., overlapping electives or labs) during course registration or when the timetable is updated.
- The system shall prevent the assignment of the same resource (faculty, room) to multiple sessions in the same time slot.

3.2 Non-Functional Requirements

3.2.1 Performance Requirements

- The system shall process conflict detection operations in real time, providing feedback within 1 second of input.
- The timetable generation process for a department shall complete within 15 seconds of input submission.
- The system shall support concurrent access by up to 500 users without noticeable performance degradation.

3.2.2 Usability and Interface Requirements

- The visual timetable shall be intuitive, color-coded, and mobile-friendly to ensure ease of use for students and faculty.
- Tooltips, icons, and minimal-click interactions shall be incorporated to reduce user learning time.
- Students and faculty shall require no more than 15 minutes of training to use the system effectively.

3.2.3 Reliability and Availability

- The system shall maintain at least 99.5

- In case of system failure, automatic recovery shall occur within 2 minutes without data loss.
- All changes (manual or automated) shall be logged with version control for auditing.

3.2.4 Security Requirements

- Only authorized users (e.g., admins, faculty) shall have access to modification features, enforced via role-based access control (RBAC).
- All communication between clients and the server shall be encrypted using SSL/TLS protocols.
- Sensitive user data (e.g., student schedules, faculty assignments) shall be stored securely with access logging and anonymized backups.

3.2.5 Interoperability and Integration

- The system shall support exporting timetables in PDF, image, and CSV formats.
- Calendar integration shall support standard calendar protocols (e.g., iCal) for compatibility with Google Calendar, Outlook, etc.
- The system shall be compatible with all modern browsers: Chrome, Firefox, Edge, and Safari.

3.2.6 Maintainability and Scalability

- The system architecture shall be modular and well-documented to support maintenance and future enhancements.
- It shall be scalable to accommodate:
- Additional departments and courses
- Increased numbers of concurrent users
- Larger elective groups or cross-department registrations

4 Examination : Existing Model

In the college, the current examination timetable is managed using a manual or semi-digital process. Typically, academic administrators or examination controllers create the schedule using spreadsheet software like Microsoft Excel or even handwritten drafts. The timetable is designed based on departmental inputs, subject lists, and exam duration guidelines, following institutional academic calendars.

The process begins with the collection of subject-wise exam requirements from each department. Dates are then assigned manually, ensuring that department-specific subjects are distributed across the exam window. Room allocations are performed based on estimated student strength and hall capacities, usually with limited software assistance.

Once finalized, the timetable is circulated as a single master document, usually in PDF or Excel format. This document includes course codes, subject names, exam dates, times, and corresponding room numbers. It is posted on college notice boards, sent via email, or uploaded to the college intranet or website.

Students are expected to refer to this master document, cross-check their registered subjects, and extract their own schedules. No personalization, automated clash detection, or centralized access system is typically provided in this model.

This method, though widely used, is still largely dependent on manual data entry and oversight.

5 Existing Model : The Challenges

In a national college like IIIT DHARWAD, where hundreds of students from multiple departments take diverse courses, managing an examination timetable manually is not only outdated but incredibly inefficient. As a Computer Science student, the stark contrast between manual processes and automated systems is striking—especially when we face the chaos of exam season.

5.1 Challenges faced by the Scheduler

1. Scheduling Conflicts

Creating a manual exam timetable involves coordinating a lot of courses, overlapping electives, and varied program structures. Human schedulers must:

- Avoid subject overlaps for students with electives from different departments.
- Ensure no student has two exams in a single day or back-to-back sessions.
- Prevent staff and room clashes.

The sheer volume of permutations makes manual planning a nightmare. A single misstep can lead to last-minute changes, impacting thousands.

2. Room Allocation Problems

Assigning appropriate rooms manually is riddled with inefficiencies:

- Large courses may be assigned to smaller rooms, leading to overcrowding.
- Rooms may be double-booked, especially during peak slots.
- Facilities like projectors or special seating (needed for specific exams) are often overlooked.

3. Lack of Real-Time Visibility

Manual schedules are typically released as static PDFs or Excel sheets. If a change occurs:

- Students may not be informed promptly.
- Version control becomes difficult.
- Miscommunication spreads confusion.

5.2 Challenges faced by the Student

1. Complex Layouts

Manual schedules are often crammed tables with course codes, dates, and rooms—all in small font and poorly organized. This leads to:

- Students misreading dates or room numbers.
- Increased dependency on word-of-mouth updates.

2. Scattered Locations

The campuses has multiple blocks and annexes. Without proper mapping:

- First-year students often struggle to find halls.
- Students waste crucial pre-exam minutes locating their venue.
- Delays and unnecessary stress become common.

3. No Personalization

Every student must decode the timetable based on their subjects. Manual systems do not offer:

- Individualized schedules.
- Alerts or reminders.
- Location guidance.

This lack of personalization leads to missed exams or misinformed attendance.

6 Existing Model : The Solution

1. Efficient Scheduling Algorithms

Automated systems can use constraint-based scheduling algorithms to:

- Avoid conflicts across departments and electives.
- Optimize room usage.
- Distribute exams to reduce stress on students and faculty.

2. Smart Room Allocation

AI and optimization models can:

- Assign rooms based on real-time capacity and requirements.
- Ensure accessibility for students with special needs.
- Prevent double-bookings and underutilization.

3. Student-Centric Access

Automation enables:

- Personalized exam schedules via web portals.
- Map integration for hall locations.
- Notifications for time, venue, and changes.

4. Real-Time Updates

Changes in schedule, venue, or invigilation can be pushed instantly to stakeholders. This drastically reduces errors and last-minute panic.

7 Requirements for Automating the Examination Timetable

7.1 Functional Requirements

7.1.1 Course and Subject Data

- The system shall maintain a list of all courses and subjects offered during the semester.
- The system shall store subject codes and titles to enable easy mapping and display.
- The system shall categorize subjects department-wise for accurate allocation and filtering.
- The system shall record credit hours and subject types (e.g., theory, lab, elective, core).
- The system shall track subject enrollment numbers to ensure suitable room allocation.

7.1.2 Student Information Management

- The system shall store and manage student enrollment data, including student ID, registered subjects, department, and year of study.
- The system shall account for elective combinations to avoid examination clashes.
- The system shall store and apply special accommodations for specific students (e.g., extra time, special seating).

7.1.3 Faculty and Invigilator Availability

- The system shall maintain a mapping of faculty allocations for each subject.
- The system shall allow input of faculty availability schedules, including leaves or other duties.
- The system shall optionally record preferred invigilation time slots.
- The system shall limit the number of exams a faculty member can supervise per day.

7.1.4 Examination Parameters

- The system shall allow configuration of exam durations per subject.
- The system shall support multiple exam types (e.g., theory, lab, viva).
- The system shall define the examination window (total number of exam days and working hours per day).
- The system shall enforce institutional constraints such as:
- No exams on weekends or public holidays.
- Maximum number of exams per student per day.

7.1.5 Room and Hall Management

- The system shall maintain a list of available examination halls and classrooms.
- The system shall store seating capacity for each room.
- The system shall tag rooms with special facilities (e.g., projector, AC, wheelchair access).
- The system shall include block and floor information for map-based student navigation.

7.1.6 Rules and Constraints Enforcement

- The system shall enforce clash rules to ensure no overlapping exams for any student.
- The system shall enforce a minimum cooling-off period between exams for each student.
- The system shall avoid scheduling same-department exams at the same hour.
- The system shall implement room distribution logic to ensure compliance with rules (e.g., social distancing, splitting large courses across multiple rooms).

7.1.7 System and User Interfaces

- The system shall provide an admin panel for timetable managers to manage all data inputs and configurations.
- The system shall offer a student-facing portal or app interface to view personalized schedules.
- The system shall support real-time updates for last-minute changes in the exam schedule.

7.2 Non-Functional Requirements

7.2.1 Performance Requirements

- The system shall generate complete examination timetables for a department within 15 seconds of input submission.
- The system shall support simultaneous access by at least 500 users without performance degradation.
- Timetable views (for students/faculty) shall load in under 2 seconds under normal network conditions.

7.2.2 Usability Requirements

- The system shall feature a responsive, intuitive UI requiring minimal training for admin staff and faculty.
- The student portal shall present personalized schedules in a clean, color-coded, and mobile-friendly format.
- Tooltips and help icons shall be available for every major operation in the admin panel.

7.2.3 Reliability and Availability

- The system shall maintain 99.5

- System recovery from crash or power failure shall not exceed 2 minutes, and no data shall be lost during rollback.

7.2.4 Scalability

- The system architecture shall support future expansion to accommodate:
- Additional departments or campuses
- Multiple concurrent examination sessions
- Large-scale elective courses with hundreds of students

7.2.5 Security Requirements

- Only authenticated users shall access admin functionalities; role-based access control shall be enforced.
- All communication between client and server shall be SSL/TLS encrypted.
- Student and faculty data shall be securely stored following data protection guidelines (e.g., anonymized logs, encrypted storage).

7.2.6 Compatibility and Integration

- The system shall be compatible with modern web browsers (Chrome, Firefox, Safari, Edge).
- The system shall optionally support integration with external calendar apps like Google Calendar and Microsoft Outlook.
- Exported timetables shall be downloadable in PDF, CSV, and image formats.

7.2.7 Maintainability and Modifiability

- Configuration files shall support editable rules for constraints (e.g., max exams per day, blackout dates) without changing source code.

8 Software Design

These are the Data flow Diagrams of our model/software:

8.1 Data Flow Diagrams

8.1.1 Level 0:

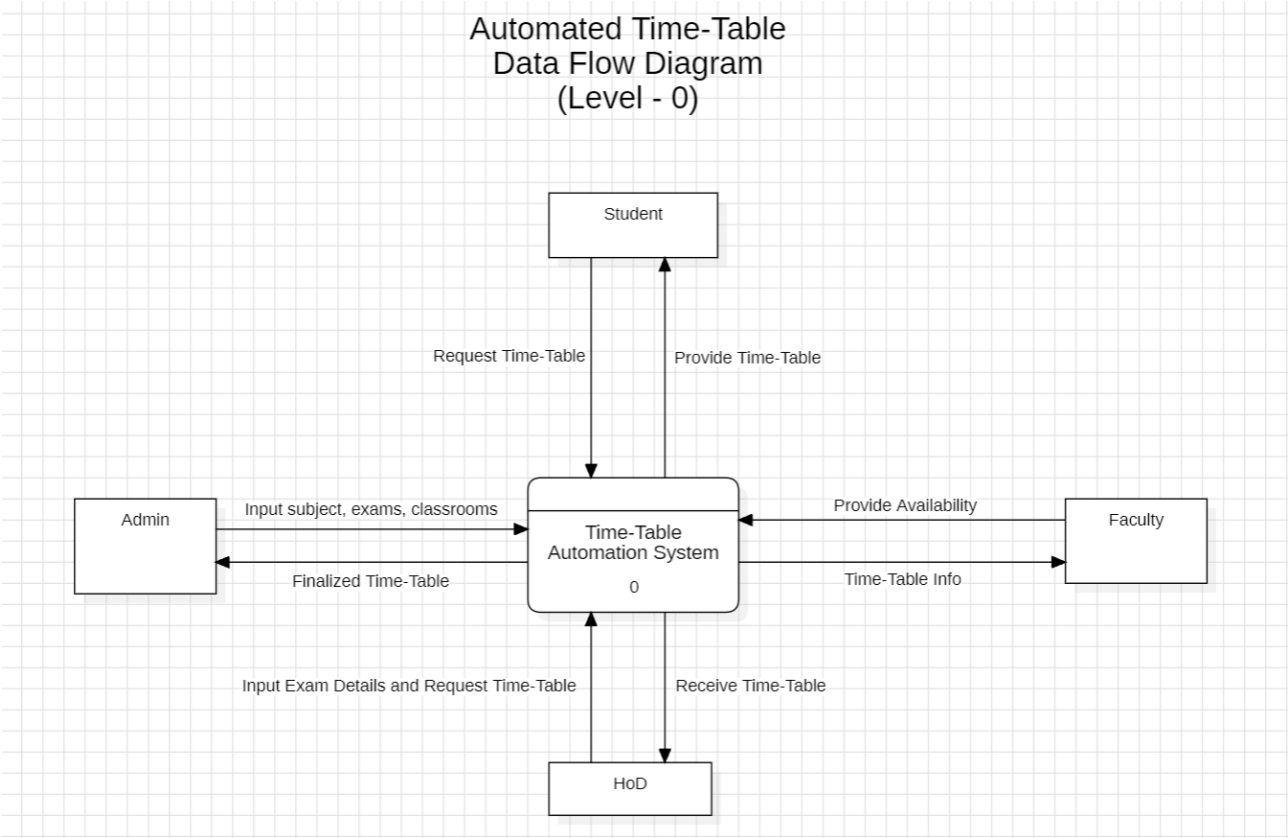


Figure 5. Level - 0 of the Automated Time-Table DFD

8.1.2 Level 1:

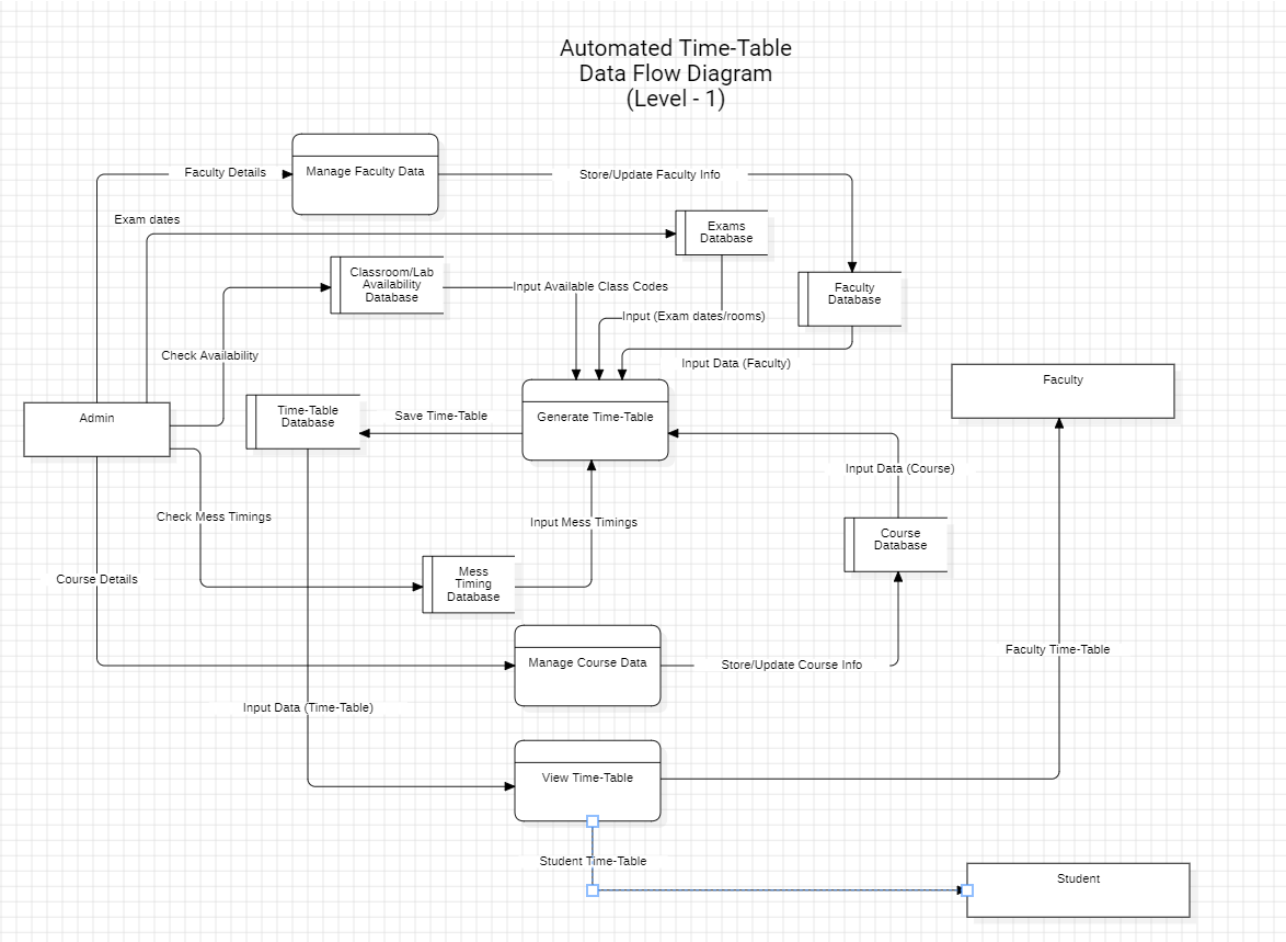


Figure 6. Level - 1 of the Automated Time-Table DFD

8.1.3 Level 2:

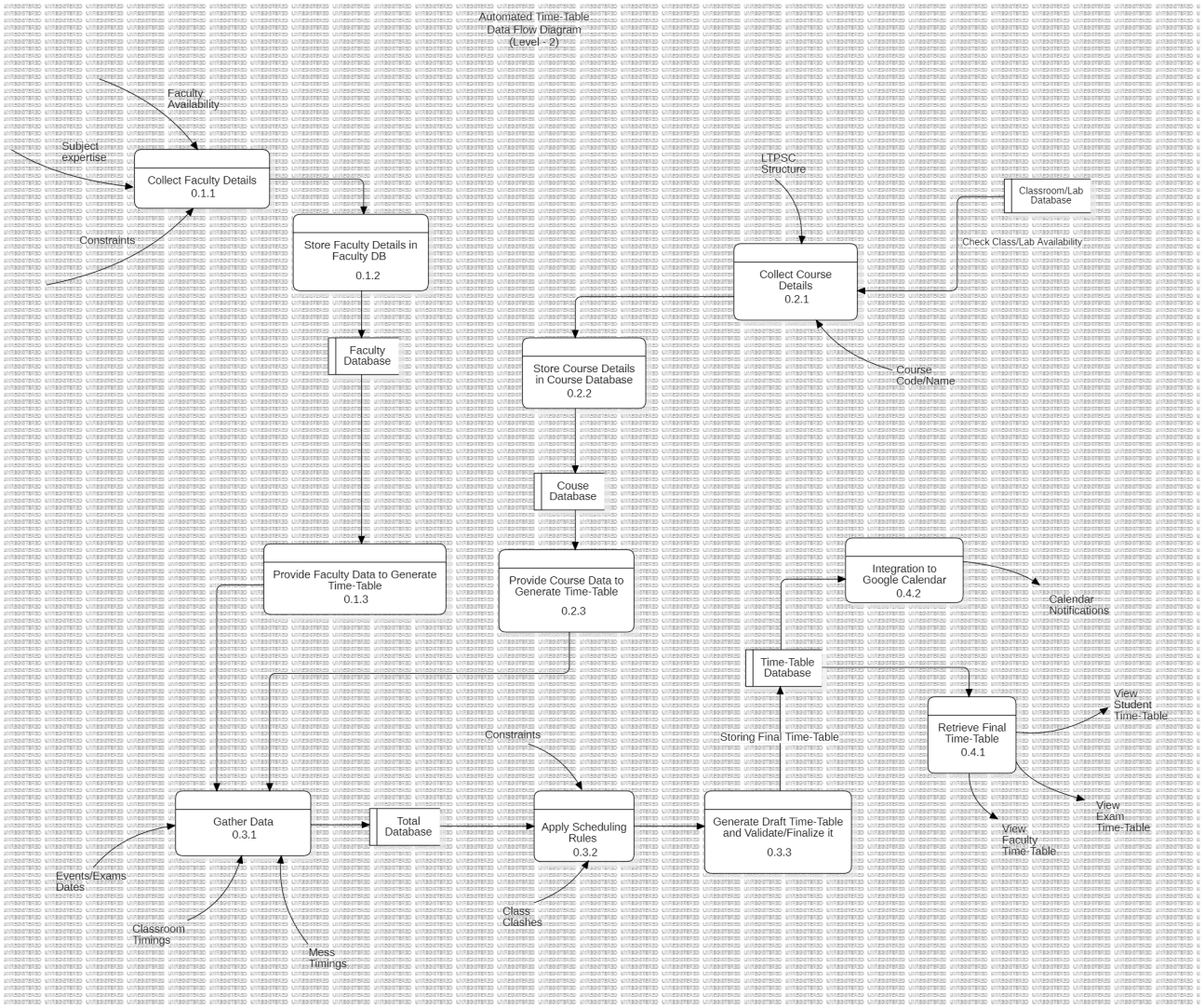


Figure 7. Level - 2 of the Automated Time-Table DFD

8.2 Data Dictionary

8.2.1 Level 0:

Field Name	Data Type	Data Format	Field Size	Description	Example
Student_ID	Integer	NNNNNN	6	Unique ID for each student	202301
Student_Name	Text	—	50	Full name of the student	Rahul Mehta
Faculty_Name	Text	—	50	Name of the faculty member	Dr. Sneha Verma
Avail._Days	Text	Day/Day	20	Days faculty is available	Mon/Wed/Fri
Avail._Time_Slots	Text	HH:MM-HH:MM	20	Time ranges when faculty is available	10:00-12:00
Subject_Code	Text	AAA999	6	Unique subject identifier	CSE101
Subject_Name	Text	—	50	Name of the subject	Data Structures
Classroom_ID	Text	ROOM-XXX	8	Identifier for the classroom	ROOM-102
Class_Capacity	Integer	NNN	3	Seating capacity of classroom	60
Exam_Date	Date	DDMMYYYY	10	Scheduled exam date	25/10/2025
Exam_Time	Time	HH:MM	5	Exam start time	09:00
Day	Text	—	10	Day of the week	Monday
Time_Slot	Text	HH:MM-HH:MM	11	Time block for class/exam	10:00-11:00

Table 1
Level 0 Data Dictionary

8.2.2 Level 1

- Faculty Record

Field	Type	Size	Description
Faculty_ID	Integer	6	Unique identifier for faculty
Name	Text	50	Full name
Department	Text	30	Faculty's department
Available_Days	Text	20	Days available
Available_Slots	Text	20	Time slots available

Table 2
Level 1 Faculty Record

- Course Record

Field	Type	Size	Description
Course_Code	Text	10	Unique identifier
Course_Name	Text	50	Name of the course
Semester	Integer	1	Semester number
Credits	Integer	1	Credit value

Table 3
Level 1 Course Record

- Exam Schedule

Field	Type	Size	Description
Course_Code	Text	10	Unique identifier
Course_Name	Text	50	Name of the course
Semester	Integer	1	Semester number
Credits	Integer	1	Credit value

Table 4
Level 1 Exam Schedule

- Room Availability Record

Field	Type	Size	Description
Room_ID	Text	10	Unique room/lab identifier
Capacity	Integer	3	Number of seats
Availability	Text	30	Days and slots available

Table 5
Level 1 Room Availability Record

- Mess Timing Record

Field	Type	Size	Description
Day	Text	10	Day of the week
Start_Time	Time	—	Mess open time
End_Time	Time	—	Mess close time

Table 6
Level 1 Mess Timing Record

- Time-Table Record

Field	Type	Size	Description
TT_ID	Integer	6	Time-table entry ID
Faculty_ID	Integer	6	Assigned faculty
Course_Code	Text	10	Course scheduled
Room_ID	Text	10	Room allocated
Day	Text	10	Day of the class/exam
Time_Slot	Text	20	Start-End time format

Table 7
Level 1 Time-Table Record

- Database

Field	**Type**	**Size**	**Description**
TT_ID	Integer	6	Time-table entry ID
Faculty_ID	Integer	6	Assigned faculty
Course_Code	Text	10	Course scheduled
Room_ID	Text	10	Room allocated
Day	Text	10	Day of the class/exam
Time_Slot	Text	20	Start-End time format

Table 8
Level 1 Database

8.3 Low-Level Design

8.3.1 Data Structures

The following data structures will be used to represent the key entities in the timetable management system.

Listing 1: Data Structures for the Timetable System

```
// Represents a student
struct Student {
    int studentId;
    char name[50];
    char department[30];
    int semester;
    int registeredCourses[10]; // course codes
};

// Represents a faculty member
struct Faculty {
    int facultyId;
    char name[50];
    char department[30];
    int availableSlots[20]; // times they are available
};

// Represents a course
struct Course {
    int courseCode;
    char name[50];
    int semester;
    int credits;
    int facultyId;
    int expectedEnrollment;
};
```

```
// Represents a classroom/exam hall
struct Room {
    int roomId;
    int capacity;
    char facilities[100];
};

// Represents a timetable entry
struct TimetableEntry {
    int entryId;
    int courseCode;
    int facultyId;
    int roomId;
    char day[10];
    char timeSlot[20];
    char type[10]; // "class" or "exam"
};
// Represents a conflict between timetable entries
struct ConflictDetail {
    int entryId1;
    int entryId2;
    char reason[100];        // description of conflict
};

// Represents a saved timetable version (for rollback)
struct TimetableVersion {
    char label[30];          // version label
    char timestamp[25];      // save time
    int entryIds[200];       // IDs included in snapshot
    int count;
};
```

8.3.2 Function Declarations

The following functions will provide the main functionality of the system.

Listing 2: Function Declarations for the Timetable System

```
// Initialize and load system config
void initSystem(const char *configPath);

// Free resources and shut down system
void shutdownSystem();

// Load all input data (students, faculty, courses, rooms)
void loadInputData();

// Load specific entities
void loadStudents(const char *path);
void loadFaculty(const char *path);
void loadCourses(const char *path);
void loadRooms(const char *path);

// Generate timetable for a semester (classes)
TimetableEntry* generateClassTimetable(int semester);

// Generate timetable for selected courses (exams)
TimetableEntry* generateExamTimetable(int courseCodes[], int count);

// Detect conflicts (student overlap, room clash, faculty double-booking)
int detectConflicts(TimetableEntry timetable[], int size, struct ConflictDetail
    conflicts[]);

// Attempt auto-resolution of conflicts
int autoResolveConflicts(TimetableEntry timetable[], int size);

// Allocate rooms to timetable entries based on capacity
```

```
void allocateRooms(TimetableEntry timetable[], int size, Room rooms[], int
    roomCount);

// Create personalized timetable for a specific student
TimetableEntry* buildStudentView(int studentId, TimetableEntry timetable[], int
    size);

// Create timetable view for a faculty member
TimetableEntry* buildFacultyView(int facultyId, TimetableEntry timetable[], int
    size);

// Export timetable to file (PDF/CSV/Image/iCal)
void exportTimetable(TimetableEntry timetable[], int size, char format[]);

// Save current timetable as a version (for rollback)
void saveVersion(TimetableEntry timetable[], int size, char label[]);

// Restore a previously saved version
TimetableEntry* rollbackTo(char label[]);

// Compute metrics (room utilization, student load, faculty load)
void computeMetrics(TimetableEntry timetable[], int size);

// Set log level for system (0=ERROR,1=INFO,2=DEBUG)
void setLogLevel(int level);
```

8.4 Module Structure

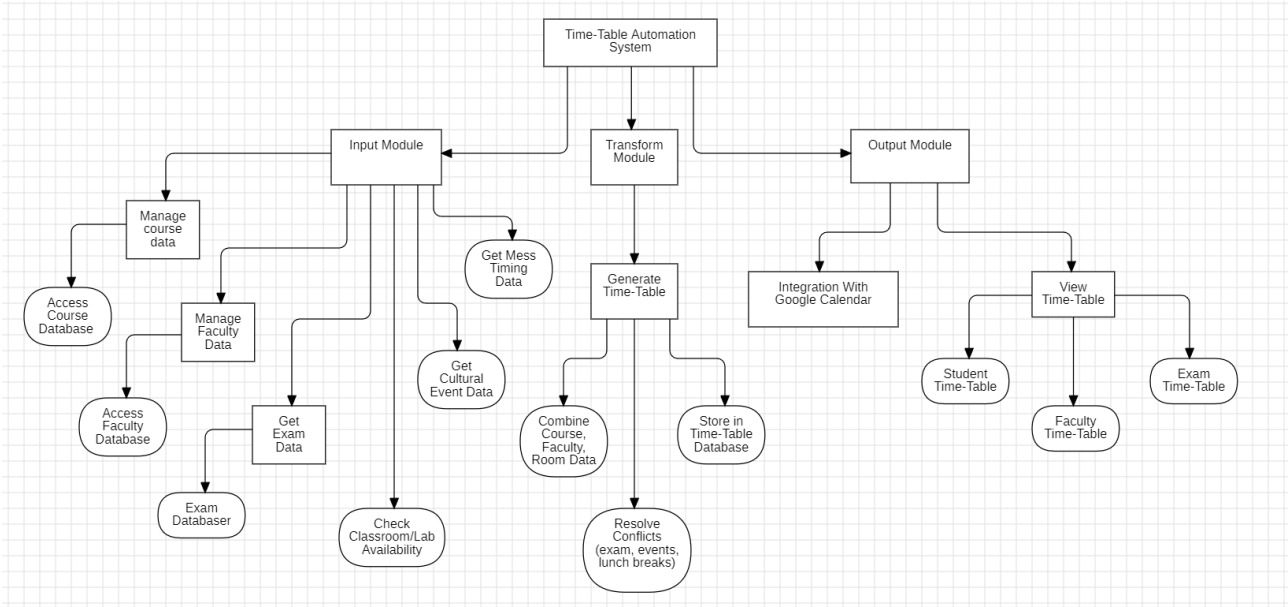


Figure 8. Module Structure

9 Coding/Implementation

This section outlines the proposed technology stack and implementation strategy for the Automated Timetable Scheduling system. The chosen technologies are selected to meet the functional and non-functional requirements outlined previously, emphasizing performance, scalability, and maintainability.

Technology Stack

- Programming Language: Python
- Data Handling : CSV files, Pandas, Numpy
- IDE/Editor: Visual Studio Code
- Frontend: React.js
- Backend: Python with Django
- Database: PostgreSQL
- Core Algorithm: Constraint Satisfaction Programming (CSP) or Genetic Algorithm
- Deployment: Docker on a cloud platform (AWS/GCP)

10 Conclusion

Timetabling is a fundamental aspect of academic institutions, encompassing both the normal everyday class schedules and the examination timetable. In top-tier national colleges, the complexity of these timetables is heightened due to a large student population, diverse courses, multiple departments, and elective choices. This documentation has explored the existing systems for both daily class schedules and examination timetables, highlighting their manual nature, the challenges they present, and the clear need for automation.

The everyday timetable, which dictates when and where classes occur, is traditionally prepared using manual methods such as spreadsheets, paper records, or ad-hoc software. Similarly, the examination timetable is often managed through manual scheduling and allocation processes. Both systems depend heavily on human coordination and data entry, requiring academic staff to balance numerous constraints, including course clashes, room availability, instructor schedules, and student enrollment patterns.

While manual timetabling may have sufficed for smaller or less complex institutions, it struggles to accommodate the dynamic and interconnected nature of modern academic programs. The existing manual systems for both class and exam timetables are prone to errors such as overlapping classes or exams, inefficient use of resources, and difficulties in communication. Students often face confusion when interpreting bulky, generic schedules that lack personalization or guidance. Faculty and administrative staff endure a substantial workload attempting to resolve clashes and update schedules manually, often under tight deadlines.

Moreover, manual processes lack real-time update capabilities, meaning any changes in schedules—whether due to room availability, instructor absence, or unforeseen circumstances—are slow to propagate. This results in misinformation, missed classes or exams, and general dissatisfaction across the campus community. The manual systems also do not provide easy access to critical data, such as optimized room allocations or invigilation duties, which further complicates planning and reduces overall efficiency.

The challenges outlined make a compelling case for the adoption of automated timetabling

systems. Automation harnesses advanced algorithms and databases to integrate diverse inputs such as course details, student enrollments, faculty availability, and room capacities. This integrated approach can simultaneously generate conflict-free daily class schedules and examination timetables that respect institutional rules and constraints.

Automated timetabling enhances accuracy and consistency, drastically reducing the likelihood of scheduling conflicts. It also significantly lightens the administrative burden by automating clash detection, room allocation, and notification processes. Students benefit from personalized, accessible schedules with clear venue guidance and timely alerts for any changes. Faculty members receive optimized invigilation assignments aligned with their availability, improving fairness and transparency.

Furthermore, automation enables quick adaptation to unexpected changes, ensuring that updated timetables are instantly shared with all stakeholders. This flexibility is critical in maintaining smooth academic operations and minimizing disruption. The ability to store and analyze historical scheduling data also supports better future planning and resource utilization.

In summary, the current manual methods of managing both normal everyday timetables and examination schedules are inadequate for the scale and complexity of modern top-tier colleges. They introduce inefficiencies, increase the risk of errors, and create communication gaps that impact the academic experience. Transitioning to automated timetabling systems represents a necessary evolution—one that leverages technology to improve operational efficiency, enhance student and staff satisfaction, and support the institution's commitment to academic excellence.

Implementing such automation requires institutional commitment and collaboration among academic departments, administration, and IT teams. However, the benefits far outweigh the initial effort and investment. As this documentation has demonstrated, the future of timetabling lies in automation, offering a smarter, more reliable, and student-centric approach to academic scheduling.

References

1. Goktug, A. N., Chai, S. C., Chen, T. (2013). A timetable organizer for the planning and implementation of screenings in manual or semi-automation mode. Journal of Biomolecular Screening, 18(8), 938–942. doi:10.1177/1087057113493720

2. Shah, M., Patel, K., Bhatt, C. (2018). Automated timetable generation using genetic algorithm. International Journal of Computer Applications, 182(18), 1–5. doi:10.5120/ijca2018917461