



TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
NATIONAL COLLEGE OF ENGINEERING

A
MAJOR PROJECT REPORT
ON
**“AUTOMATIC ROUTINE GENERATOR USING
BACKTRACKING ALGORITHM”**

SUBMITTED BY:

BIBEK ARYAL (NCE077BCT008)
MIJASH ACHARYA (NCE077BCT016)
PRERANA SUBEDI (NCE077BCT022)
SUSMITA DHAMALA (NCE077BCT037)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

LALITPUR, NEPAL

MARCH, 2024

1. Page of Approval

TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS
DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

The undersigned certify that they have read, and recommended to the Institute of Engineering for acceptance, a project report entitled "Travel Advisor Kathmandu" submitted by Bibek Aryal, Mijash Acharya, Prerana Subedi and Susmita Dhamala in partial fulfilment of the requirements for the Bachelor's degree in Electronics Communication / Computer Engineering.



Er. Anup Shrestha
Supervisor
Department of Computer Engineering

Er. Suroj Burlakoti
HOD
Department of Electronics and Computer Engineering

Name
External Examiner
Name of the coordinating committee

DATE OF APPROVAL: March.2025

2. Copyright

The author has agreed that the Library, Department of Electronics and Computer Engineering, National College of Engineering, and Institute of Engineering, may make this report freely available for inspection. Moreover, the author has agreed that permission for extensive copying of this project report for scholarly purpose may be granted by the supervisors who supervised the project work recorded herein or, in their absence, by the Head of the Department wherein the project report was done. It is understood that the recognition will be given to the author of this report and to the Department of Electronics and Computer Engineering, National College of Engineering, Institute of Engineering in any use of the material of this project report. Copying or publication or the other use of this report for financial gain without approval of to the Department of Electronics and Computer Engineering, National College of Engineering, Institute of Engineering and author's written permission is prohibited.

Request for permission to copy or to make any other use of the material in this report in whole or in part should be addressed to:

Head

Department of Electronics and Computer Engineering

National College of Engineering

Talchhikhel, Lalitpur

Nepal

3. Acknowledgements

We would like to express our sincerest gratitude and appreciation to all those who have contributed to the completion of this project. First and foremost, we extend our heartfelt thanks to Er.Anup Shrestha, our supervisor, for his invaluable guidance, support, and encouragement throughout the duration of this project. His expertise and insightful feedback have been instrumental in shaping the direction and quality of this work. Furthermore, we would also like to express our heartfelt gratitude towards all our department teachers for their constant guidance and input on the project. We are also thankful to the Department of Electronics and Computer Engineering, for providing us the necessary resources and guidance materials for the successful completion of this project. This project would not have been possible without the contributions and support of all those mentioned above. Thank you.

4. Abstract

Developing an automatic routine generator using back tracking algorithm is a web-based project designed for timetable creation for the National College of Engineering(NCE). The primary aim is to develop a system that automates the generation of a college timetable, considering the availability and workload of teachers, the distribution of subjects across different semesters, and the constraints specific to practical sessions. Leveraging Constraint Satisfaction Problem (CSP) techniques, the system integrates multiple constraints such as teacher availability, workload limits, qualified teacher assignment, no teacher overlap, and subject daily limits. The project uses a MongoDB database to store detailed information about subjects and teachers, categorized by department and semester. The timetable generation process starts with user inputs for the department, semester, college start and end times, and period length, and then fetches relevant data from the database to apply the CSP model and generate the initial timetable. Additionally, the system employs algorithms to prioritize part-time teachers, balance workloads, and schedule practical sessions effectively. We will also implement mechanisms to dynamically accommodate changes and updates to the generated timetable based on specific inputs from teachers for each alteration. By automating the timetable generation process, the project aims to reduce administrative overhead, ensure fair workload distribution among teachers, demonstrating flexibility in handling various constraints and adapting to departmental needs.

Keywords: web-based, ML, constraints, CSP, automatic, Mongoddb

Contents

1	Page of Approval	i
2	Copyright	ii
3	Acknowledgements	iii
4	Abstract	iv
	List of Figures	vii
	List of Tables	viii
	List of Abbreviations	ix
5	Introduction	1
5.1	Background	1
5.2	Problem statements	2
5.3	Aims and Objectives	3
5.4	Scope.....	3
6	Literature Review	4
6.1	Related work	4
6.2	Related theory	6
7	Proposed Methodology	7
7.1	System Block Diagram.....	7
7.2	Algorithms.....	8
7.2.1	Schedule Generation using Constraint Satisfaction Problem (CSP) and Backtracking Algorithm	8
7.3	Proposed System Design	9
7.3.1	Admin Functionalities	9
7.4	System Development Model	10

8	Tools Used	12
8.1	Frontend Development	12
8.2	Backend Development	13
9	Result and Discussion	14
9.1	Data Collection	14
9.2	Data Preprocessing	16
9.3	Output	16
9.4	Outcome	26
10	Future Enhancements	27
	References	27

List of Figures

7.1	System Block diagram	7
7.2	Proposed Diagram of System	10
9.1	Subject-Wise Workload Format	15
9.2	Teacher-Wise Workload Format	15
9.3	Teacher Data.....	16
9.4	Subject Data	16
9.5	Home Page	17
9.6	Login Page	17
9.7	Admin Dashboard	18
9.8	User Management Page.....	18
9.9	Teacher Management Page.....	19
9.10	Subject Management Page	19
9.11	Teacher Edit Page.....	20
9.12	Sign up Page.....	20
9.13	System Training Data Flow Graph.....	20

List of Tables

9.1	Workload distribution by designation.....	22
9.2	Room availability for different practical types	23

List of Abbreviations

AI	Artificial Intelligence
CRUD	Create Read Update Delete
CSP	Constraint Satisfaction Problem
CSS	Cascading Style Sheet
HOD	Head Of Department
HTML	HyperText Markup Language
ML	Machine Learning
MVT	Model View Template
RDBMS	Relational Database Management System
SQL	Structured Query Language

5. Introduction

The Automated Routine Generator represents a transformative difference in the management of scheduling processes within educational institutions, specifically universities. This system uses advanced algorithms and technologies to automate the tough task of generating class schedules, thereby advanced resource allocation, minimizing conflicts, and significantly reducing errors. By replacing labor-intensive manual scheduling methods with an efficient and precise automated approach, the Automated Routine Generator promises to redefine the academic landscape. Its intuitive interface and strong backend architecture offer a better user experience, ensuring optimal scheduling outcomes for students, faculty, and staff alike. With its potential to reduce administrative burdens, and enhance scheduling accuracy, the Automated Routine Generator emerges as a important tool in modernizing educational institutions and fostering a conducive learning environment.

5.1 Background

Before the concept of automatic routine generators was invented, people relied on manual scheduling methods. This involved using physical tools such as calendars, planners, and wall charts, along with basic arithmetic and logical reasoning to allocate tasks and resources. Schedulers, often university administrators or staff, manually assessed constraints like room availability, faculty schedules, and course requirements to create timetables. This process was labor-intensive, time-consuming, and inclined to human error, especially for complex scheduling needs such as university timetables or large-scale industrial operations.

Manual scheduling methods before the emergence of automatic routine generators were time-consuming, susceptible to human error, and inefficient, often resulting in sub optimal use of resources. These methods struggled with scalability, flexibility, and managing complex constraints, leading to inconsistent quality and difficulty in adapting to last-minute changes. Additionally, poor data management and lack of optimization

made decision-making slower and less effective, requiring significant human resources and increasing operational costs. Research on manual scheduling approaches highlights several key limitations: significant time investment, high error rates, and inefficiencies, particularly in complex environments like universities. Manual methods struggle with scalability, adapting to changes, and managing complex constraints, resulting in inconsistent quality and increased operational costs. Comparative studies consistently demonstrate the superiority of automated systems.

5.2 Problem statements

Manual routine generation for academic institutions often faces numerous challenges, which can lead to inefficiencies, errors, and overall dissatisfaction among faculty and students. Some of which are:

- The task of routine making can be tedious, time consuming and error prone when done manually. Slightest mistake can lead to a huge disruption in the educational process.
- Routine once made is difficult to alter.
- It can result in inconsistent time allocation which causes imbalanced workloads, inefficient resource use, and administrative challenges.
- Lack of real-time updates leads to miscommunication and delays in addressing the changes.
- Reduced accountability and tracking make it harder to monitor scheduling issues and resource allocation effectively.
- Lack of optimization results in inefficient scheduling and resource use.
- Maintaining historical records becomes challenging, hindering the ability to track past scheduling decisions and learn from previous experiences.

5.3 Aims and Objectives

The main aim of our project is to develop a system that automatically generates class routines using Machine Learning.

Objectives:

- To develop a system that can generate a printable routine file
- To implement improved resource management, including faculty and staff management.

5.4 Scope

An automated routine generator using machine learning in the context of Nepalese universities offers a wide scope of applications to address various challenges in academic scheduling. By adapting machine learning algorithms to handle the specific requirements of Nepalese universities, such as cultural events and examination schedules, the generator can produce schedules that meet the diverse needs of stakeholders. Additionally, the generator can enhance resource utilization by analyzing historical data to identify usage patterns and allocate resources more efficiently throughout the semester. It can also adaptive adjust schedules based on dynamic factors like faculty availability and unexpected events, ensuring flexibility and responsiveness. Furthermore, machine learning techniques can help detect and resolve scheduling conflicts while incorporating constraints unique to Nepalese universities, such as religious holidays and local festivals. By continuously analyzing feedback from teachers and faculty, the generator can tailor schedules to improve satisfaction and accommodate diverse conditions. Integration with existing university management systems streamlines data exchange and facilitates informed decision-making by providing administrators with actionable insights and recommendations based on data-driven analysis. Overall, an automated routine generator using machine learning has the potential to change academic scheduling in Nepalese universities, improving efficiency, fairness, and stakeholder satisfaction while addressing the unique challenges of the local context.

6. Literature Review

The development of automated scheduling systems has been a prime focus of research for decades, driven by the need to improve resource allocation and minimize scheduling conflicts in various domains, including educational institutions. This literature review explores the evolution of scheduling methods, the theoretical foundations of automated scheduling algorithms, and the practical applications of these technologies in university settings.

6.1 Related work

L. Ying, H. K. Mammi proposed a web application that used a genetic algorithm to find the most suitable schedule, referring to constraints such as subject, lecture hall and student group. The authors defined two types of constraints, i.e., soft constraints and hard constraints. But only strict restrictions are met. Due to the large number of electives offered by the program, there were conflicts between electives. This happened because the soft limits were not followed. It also takes about 5 minutes 34 seconds on average to build the schedule[1].

M. Bagul, Mayuri R and Chaudhari, explored various methods for timetable generation, including local search algorithms and constraint programming techniques. They implemented the project by constructing a dataset from the university's course timetables, room capacities, and time slots. Initial feasible solutions were generated using a heuristic algorithm, and then refined using a genetic algorithm. Limitations of the research: -The project may face limitations in handling the complexity of real-world timetabling scenarios, necessitating further algorithmic refinement to efficiently manage larger datasets and intricate constraints[2].

Ratul Prosad, Md. Ashikur Rahman Khan, and Ishtiaq Ahammad have designed a class routine and exam hall invigilation system using genetic algorithms and a greedy approach. The system automates schedule management and stores all information in a centralized database. It allows for the distribution of classrooms, adjustment of schedules under complex conditions, display of available schedules for exam hall surveil-

lance, and effective classroom utilization[3].

Savdekar, Mansi and Bornare, presented an efficient timing algorithm designed for an automated timetable system. This system could handle various challenges and generated schedules for classes and teachers. It considered each teacher's availability, resource usage, and specific rules for different classes, semesters, teachers, and grade levels. This allowed both teachers and students to view their schedules after a semester ends, even though they do not participate in the planning process[4].

Maryam Alwashahi discussed the investigation and optimization of course scheduling at Sohar University using a genetic algorithm. The research focuses on creating course timetables that allocate events (time, subject, and faculty) effectively, using available resources and avoiding conflicts. The paper introduced a suitable methodology and an efficient algorithm for solving scheduling problems. The proposed system met all strict constraints without any clashes between schedules. On average, it took 23 minutes to generate the best schedule for one faculty member[5].

They structured a dataset containing faculty, subjects, rooms, and time intervals to facilitate the generation of accurate timetables. Furthermore, they discussed potential enhancements such as refining the backtracking algorithm to handle larger datasets efficiently, integrating machine learning for adaptive optimization, and improving the user interface[6].

Rahman, Ashiqur Md and Giasuddin used two heuristic algorithms to address the NP-hard problems of academic routine generation and exam timetabling for universities that followed an open credit system where students could choose any course after completing prerequisites. The algorithms provided reasonably good solutions but had limitations in handling all constraints simultaneously and depended on the input data size. The key contributions lay in tackling the challenges of routine and exam scheduling in an open credit system environment through heuristic techniques[7].

Prof. Jyothi Patil and his team purposed a system to create an efficient and conflict-

free automatic timetable generator for colleges. The system was implemented using HTML and CSS for the front-end, and Python and MySQL for the back-end, with a genetic algorithm employed to generate the timetables. Several limitations were noted, including the challenges in handling complex constraints, the resource-intensive nature of the genetic algorithm, the need for manual adjustments in the event of last-minute changes, and difficulties in generating a diverse initial population[8].

Sohail Qureshi, Nikita Veer, Vaishnavi Ugale, and Priyanka Agrawal designed a project aimed to automate timetable generation using a genetic algorithm for optimization. Python libraries like tkinter and Django were used for the interface and web application. Limitations included the need for manual adjustments for absences and the resource-intensive nature of the genetic algorithm[9].

While aSc Timetables, a desktop software, significantly aids in creating conflict-free and equitable schedules, its specific features offer notable advantages. It offers features like:

1. Automated Scheduling: Efficient, conflict-free timetables.
2. Customization: Flexible scheduling options.
3. User-Friendly Interface: Easy adjustments and updates.
4. Mobile Access: Schedules available on mobile devices.

One significant limitation of aSc Timetables is its labor-intensive setup process. The software requires users to define over 30 constraints, making it time-consuming and demanding to configure. It is not efficient for the university scheduling.[10]

6.2 Related theory

Backtracking Algorithm:

Backtracking is a depth-first search algorithm that incrementally builds candidates for the solution and abandons a candidate/backtracks as soon as it determines that this candidate cannot lead to a valid solution.

Implementation in the System:

- Start with an empty schedule.
- Assign the first class to the first available time slot and room that satisfies all constraints.

- Recursively assign subsequent classes to available time slots and rooms.
- If a constraint is violated, backtrack to the previous step and try the next available option.

7. Proposed Methodology

7.1 System Block Diagram

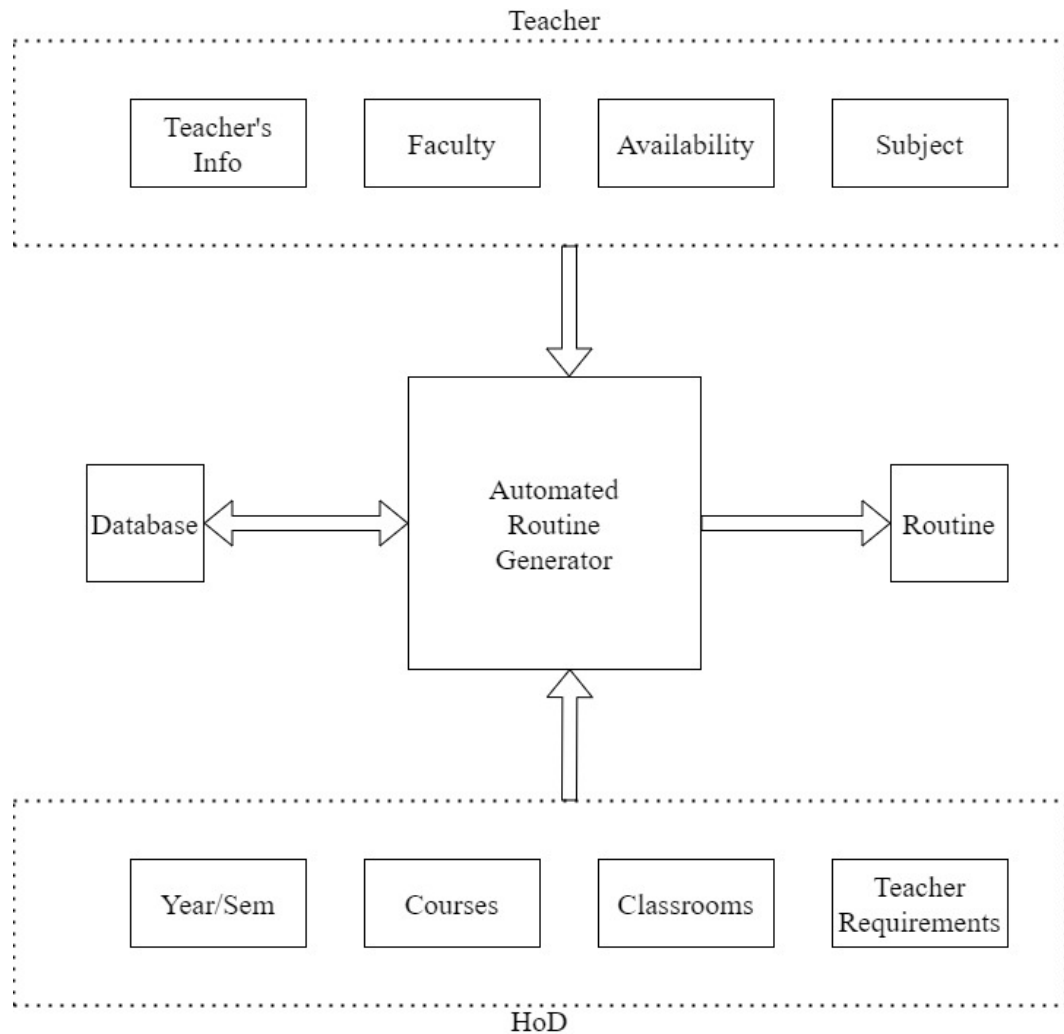


Figure 7.1: System Block diagram

The Automated Routine Generator system consists of four main components: Teacher's Input, Admin's/HOD's Input, Database, and Routine Generation. Teachers input their personal details, faculty, availability, and subjects they will teach, while admins or HODs provide information such as year/semester, courses, classrooms, and teacher requirements. All this information is recorded in a central database, which connects to the system. Based on these inputs, the system generates a comprehensive routine that

accommodates all scheduling constraints and preferences. The generated routine can then be viewed or downloaded by the HOD, teachers, and admin.

7.2 Algorithms

7.2.1 Schedule Generation using Constraint Satisfaction Problem (CSP) and Backtracking Algorithm

1. Initialization

- Start with an empty schedule matrix S where rows represent time slots and columns represent classrooms.
- Define constraints such as teacher availability T_i , room availability R_j , and subject requirements C_k .

2. Recursive Function

- **Base Case:** If all classes are scheduled, return the schedule S .
- **Recursive Case:** For each class c , try assigning it to each time slot t and room r :
 - Check if assigning c to t and r violates any constraints.
 - If no constraints are violated, place c in $S[t][r]$ and recursively call the function with the next class.
- **Backtracking:** If a constraints is violated, remove c from $S[t][r]$ (backtrack) and try the next available option.

Mathematical Explanation

This mathematical explanation defines the constraints for scheduling classes by ensuring that a teacher and a room are available for a given time slot. It introduces binary variables: T_i to indicate teacher availability and R_j for room availability. The goal is to assign a schedule $S[t][r]$ such that no teacher or room is double-booked at any time. The example demonstrates the step-by-step assignment of classes to time slots and rooms while adhering to these constraints, ensuring a conflict-free routine.

- Let T_i be a binary variable where $T_i = 1$ if teacher i is available at time t .

- Let R_j be a binary variable where $R_j = 1$ if room j is available at time t .
- The objective is to find $S[t][r]$ such that:

$$\sum_{i=1}^n T_i \cdot S[t][r] \leq 1$$

and

$$\sum_{j=1}^m R_j \cdot S[t][r] \leq 1$$

for all t and r .

7.3 Proposed System Design

System Overview:

The figure 5.2 represents a comprehensive **Automated Routine Generator System** designed to facilitate the efficient scheduling of academic classes for educational institutions. The system is composed of several key components, which are interconnected to streamline user management, teacher assignments, subject scheduling, and routine generation processes.

User Access and Management The system provides access to two primary roles:

- **HOD/DHOD (Head of Department/Deputy Head of Department)**
- **Admin**

Users can log in or sign up through a dedicated *Login/Signup Module*. Once authenticated, they are authorized to perform specific tasks based on their assigned roles.

7.3.1 Admin Functionalities

The **Admin** plays a central role in managing the system's core data. This includes:

- **User Management:** Adding, editing, or deleting users who can access the system.
- **Teacher Management:** Managing teacher records, including their assigned subjects, workloads, and availability.

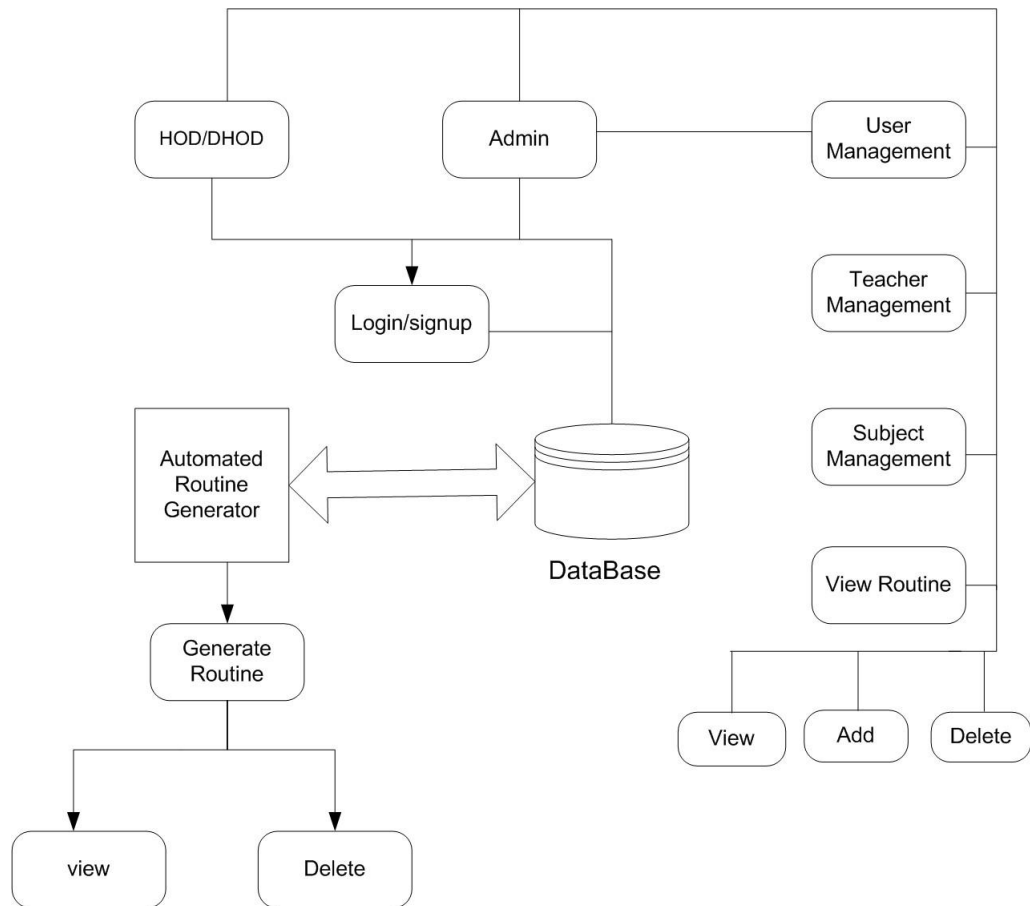


Figure 7.2: Proposed Diagram of System

- **Subject Management:** Maintaining details about subjects, including their credit hours and practical or theory classifications.

7.4 System Development Model

For the development of the "Automated routine generator using ML" system, we have opted for the Incremental Model as our system development approach. This method breaks down the project into manageable increments, allowing us to deliver functional subsets of the system at regular intervals. By doing so, we can prioritize essential features, provide early user feedback, and reduce project risks. The model's flexibility accommodates changes and user involvement throughout the development process, ensuring a close alignment with user needs. Each increment undergoes thorough testing and deployment, with subsequent iterations incorporating feedback and refining functionalities. This modular development approach enhances maintainability and scalability,

facilitating the integration of new features. The phases include requirements gathering, design, implementation, testing, and enhancement for subsequent increments.

8. Tools Used

8.1 Frontend Development

HTML

Hypertext Markup Language(HTML), serves as the backbone of web pages, defining their structure and content through a series of elements or tags. These elements encompass various components like headings, paragraphs, links, images, and forms, collectively shaping the visual and functional aspects of a web page. HTML provides the fundamental framework upon which other web technologies, such as CSS and JavaScript, build to create dynamic and interactive online experiences.

CSS

Cascading Style Sheets, complements HTML by controlling the presentation and layout of web documents. It allows developers to define styles for HTML elements, including attributes like colors, fonts, spacing, and positioning. By separating content from presentation, CSS enables consistent styling across multiple web pages and ensures a visually appealing and user-friendly interface across different devices and screen sizes.

Bootstrap

It stands as a robust front-end framework, providing developers with a comprehensive toolkit for building responsive and mobile-first websites and web applications. By offering pre-designed templates, components, and utilities, Bootstrap streamlines the development process, allowing for rapid prototyping and iteration. Its grid system and responsive design principles ensure that projects are accessible and contributing to a seamless user experience.

8.2 Backend Development

Python

Python is a high-level, interpreted programming language known for its readability and simplicity. Python emphasizes code readability with its use of significant indentation. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python's extensive standard library and vibrant ecosystem of third-party packages make it suitable for a wide range of applications, from web development and data analysis to artificial intelligence and scientific computing. Its simplicity and versatility have contributed to its widespread adoption in both industry and academia.

Mongodb

MongoDB is a NoSQL, document-oriented database designed for scalability, flexibility, and performance. Developed by MongoDB Inc., it stores data in flexible, JSON-like documents, which allows for varied and hierarchical data structures. This schema-less design makes it easier to handle changing data requirements and to scale horizontally. MongoDB supports a rich query language, indexing, and real-time aggregation, enabling efficient data retrieval and manipulation. Its distributed architecture provides high availability and fault tolerance, making it a popular choice for modern applications that require fast, iterative development and the ability to handle large volumes of data.

Flask

Flask is a lightweight and versatile Python web framework designed for building web applications and RESTful APIs. Known for its simplicity and flexibility, Flask uses WSGI for request handling and Jinja2 for dynamic HTML templating. It includes a built-in development server and debugger, making it ideal for rapid prototyping. Flask's modularity allows developers to start with a basic app and scale it by adding extensions for features like database integration, authentication, and more. Its ease of learning, active community, and scalability make it suitable for projects ranging from small websites to complex APIs and microservices.

9. Result and Discussion

Overview

We successfully completed the task of collecting and inserting data for courses, teachers, and classes into a MongoDB database. This foundational step ensures that the automatic timetable generation project has the necessary data for accurate and efficient timetable creation. Additionally, we collected data from different faculty heads of NCE to enhance our dataset and ensure thorough coverage of all course offerings and faculty availability.

Additionally, we expanded our dataset by gathering information from different faculty heads at NCE, ensuring that our system considers department-specific constraints, faculty preferences, and course dependencies. This comprehensive approach enhances the accuracy of the generated timetable and ensures that the system includes scheduling logic, conflict resolution mechanisms.

9.1 Data Collection

Courses:

1. Course Code: Unique identifier for each course.
2. Course Title: Descriptive title of the course.
3. L: Lecture Hours
4. T: Tutorial Hours
5. P: practical Hours
6. Total: Total credit hours per week.
7. Faculty: Faculty responsible for the course.

Teachers:

1. Teacher ID: Unique identifier for each teacher.

2. Name: Full name of the teacher.
3. Department: Department to which the teacher belongs.
4. Workload: Total teaching workload assigned.
5. Part/Full Time: Employment status.

6. Availability: Times when the teacher is available.

7. Designation: position held by an individual within an organization, defining their role and responsibilities.

Department of Electronics and Computer Engineering								
S.N	Subject	Year/ part	Prog	L/W	P/W	Group	Teachers Name	Full / Part Time
1	EX 451 Basic Electronics Engineering	I/II	BCT+BEL	5			Shahil Subedi	Full Time
					3	A1/A2	Shahil Subedi	Full Time
							Rajan Kusi	full time
			BCE	5			Dr. Kishor Kumar Adhikari	Full Time
					3	A1/A2	Dr. Kishor Kumar Adhikari	Full Time
							sahil subedi	full time
2	SH 553 Numerical Methods	II/II	BCT	4			Dr.kishore Adhikari	Part Time
					3	A1	Dr.kishore Adhikari	full time
							Sharmila Bista	full time
					3	A2	Sharmila Bista	Full Time
			BEI+BEL	5			Rajan Kusi	full time
							Binod Stoula	Full Time
							Subash Panday	Full Time
					3	BEL	Rajan Kusi	Full Time
							Subash Panday	Full Time
					3	BCT	rajan Kusi	full time
3	CT 551 Discrete Structure		BCT	5			Subash Panday	Full Time
			BEI	5			Anup Shrestha	full time
4	CT 552 Data Structure and Algorithm	II/II	BCT	5			Bibha sthapit	Full Time
					3	A1	Himal Rawal	Part Time
					3	A2	Himal Rawal	Part Time

Figure 9.1: Subject-Wise Workload Format

S.N	Teachers Name	Designation		Subject	Program	Year/Part	L	P	Total
1	Pradip Adhikaree	Jr. Associate Professor	Full-time	Advance Electronics	BEI	II/II	5		12.5
				Microprocessors	BEL	II/II		1.5	
				Communication Systems	BCT	III/I		3	
2	Anup Shrestha	Sr. Lecturer	Full-time	Discrete Structure	BEI	II/II	5		16
				Operating System	BCT	III/II	5	3	
				DSA	BEI	II/II		3	
3	Dr.Kishore Adhikari	Associate Professor	Full-time	Numerical Method	BCT	II/II	5	3	16
				Basic Electronics	BCE	I/II	5	3	
4	Suroj Burlakoti	Sr. Lecturer	Full-time	Microprocessors	BCT	II/II	6	6	21
				Microprocessors	BEL	I/II	6	3	
5	Subash Panday	Sr. Lecturer	Full-time	Database Management System	BCT	III/II	5	6	25
				Artificial Intelligence	BCT	III/II		3	
				Numerical Method	BEI+BEL			6	
				Discrete Structure	BEI	II/II	5		
6	Sharmila Bista	Sr. Lecturer	Full-time	Artificial Intelligence	BCT	III/II	5	3	28
				Operating System	BCT	III/II		3	
				Simulation and Modelling	BCT	IV/II	5	3	
				Numerical Method	BCT			6	
				Internet and Intranet	BCT	IV/II		3	

Figure 9.2: Teacher-Wise Workload Format

9.2 Data Preprocessing

In preparing our data for the automatic timetable generation project, we began by collecting raw data from various sources, including inputs from faculty heads at NCE. Our initial focus involved rigorous data cleaning to handle missing values, remove duplicates, and rectify inconsistencies. Subsequently, we transformed the cleaned data into JSON format, aligning it with MongoDB's document-oriented structure for efficient storage and retrieval. This process also included normalization steps to organize data logically and ensure compatibility with our database schema. Additionally, we integrated supplementary data from faculty heads to enrich our dataset, ensuring thorough coverage of course offerings and faculty availability. Through systematic validation, we ensured the accuracy and completeness of the converted JSON data. Choosing JSON offered advantages in flexibility and compatibility with our database technology, setting a solid foundation for future enhancements.

```
{
  "name": "Bibha Sthapit",
  "type": "Part Time",
  "designation": "Teacher",
  "Course_Code": [
    "CT 552"
  ],
  "availability": {
    "Monday": [ "07:00-10:00" ],
    "Tuesday": [ "10:00-14:00" ],
    "Wednesday": [ "08:00-11:00" ],
    "Friday": [ "12:00-13:00" ]
  }
},
{
  "name": "Santosh Giri",
  "type": "Part Time",
  "designation": "Teacher",
  "Course_Code": [ "CT 651" ],
  "availability": {
    "Monday": [ "07:00-10:00" ],
    "Thursday": [ "11:00-13:00" ]
  }
},
},
```

Figure 9.3: Teacher Data

```
{
  "Name": "Engineering Physics",
  "Code": "SH 402",
  "L": 4,
  "T": 1,
  "P": 2,
  "Total": 7,
  "Ptype": "Physics Lab",
  "Semester": 1,
  "Faculty": "Computer"
},
{
  "Name": "Applied Mechanics",
  "Code": "CE 401",
  "L": 3,
  "T": 2,
  "Total": 5,
  "Semester": 1,
  "Faculty": "Computer"
},
},
```

Figure 9.4: Subject Data

9.3 Output

Here is the output that we have achieved successfully to date:

Home Page

This page serves as the main entry point for users, providing an overview of the website and easy navigation to other sections such as login, signup, and available routines.

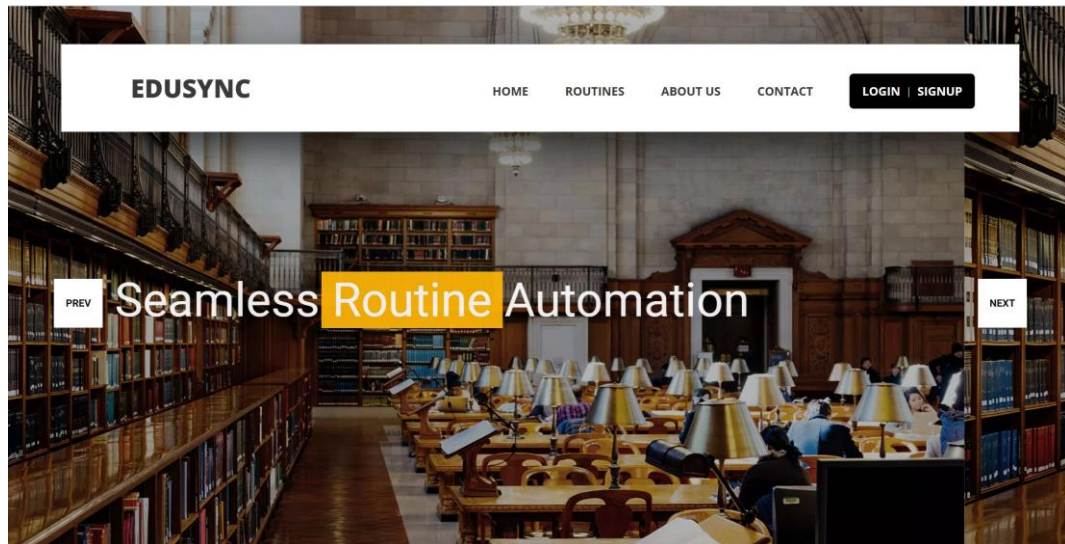


Figure 9.5: Home Page

Login

The login page allows users to securely input their email and password to gain access to their accounts, featuring options for password recovery and registration for new users.

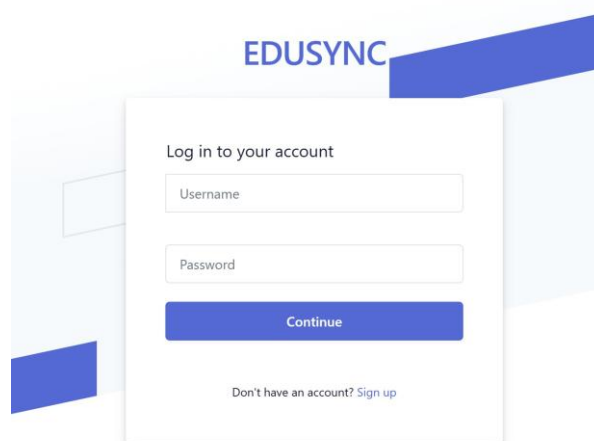


Figure 9.6: Login Page

Admin Dashboard

The admin dashboard provides administrators with an intuitive interface to manage users, blogs, and system settings, offering secure access and comprehensive functionality.

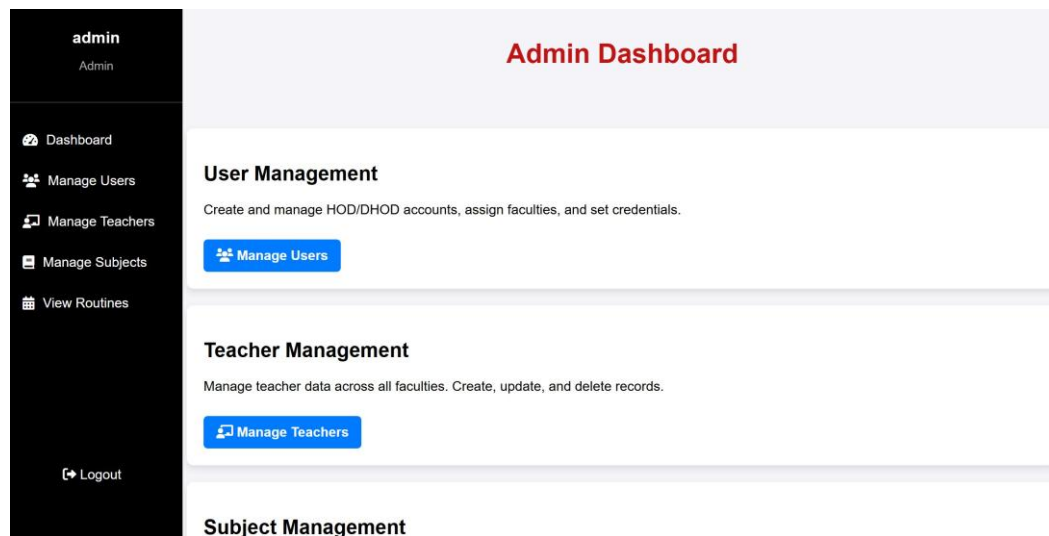


Figure 9.7: Admin Dashboard

User Management

The user management page enables administrators to add, update, or delete user profiles, assign roles, and manage various user details efficiently.

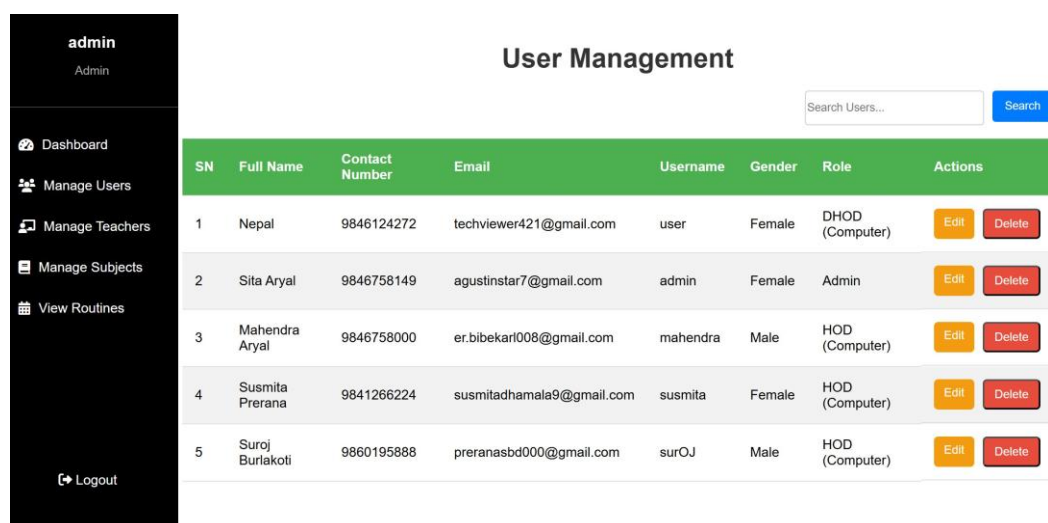


Figure 9.8: User Management Page

Teacher Management

The teacher management page gives administrators the ability to add, update, or remove teacher profiles, manage their availability, roles, and teaching preferences.

SN	Name	Designation	Subject	Actions
1	Anup Shrestha	Teacher	CT501	<button>Edit</button> <button>Delete</button>
2	Dr Kishore Adhikari(KA)	Teacher	SH553	<button>Edit</button> <button>Delete</button>
3	Madhu Krishna Shrestha(MKS)	Teacher	SH551	<button>Edit</button> <button>Delete</button>
4	Madhu Krishna Shrestha(MKS)	Teacher	SH201	<button>Edit</button> <button>Delete</button>
5	Madhu Krishna Shrestha(MKS)	Teacher	SH501	<button>Edit</button> <button>Delete</button>
6	PA	Teacher	EE 501	<button>Edit</button> <button>Delete</button>

Figure 9.9: Teacher Management Page

Subject Management

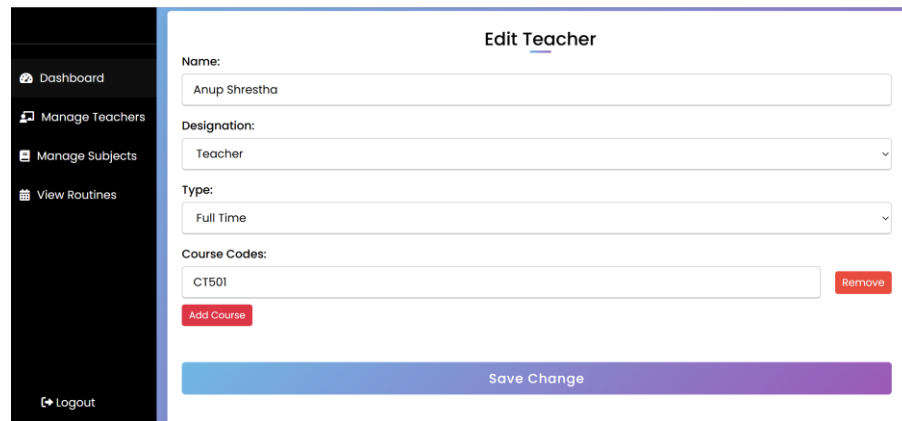
The subject management page allows administrators to add, edit, or remove subjects, assign credit hours, and oversee all subject-related details effectively.

SN	Name	Code	Faculty	Sem	L	T	P	Total	Actions
1	Applied Mechanics	CE 401	Electrical	1	3	2		5	<button>Edit</button> <button>Delete</button>
2	Applied Mechanics	CE 401	Electronics	1	3	2		5	<button>Edit</button> <button>Delete</button>
3	Basic Electrical Engineering	EE 401	Electrical	1	3	1	1.5	5.5	<button>Edit</button> <button>Delete</button>
4	Basic Electrical Engineering	EE 401	Computer	1	3	1	1.5	5.5	<button>Edit</button> <button>Delete</button>
5	Basic Electrical Engineering	EE 401	Electronics	1	3	1	1.5	5.5	<button>Edit</button> <button>Delete</button>
6	Computer Programming	CT 401	Civil	1	3		3	6	<button>Edit</button> <button>Delete</button>

Figure 9.10: Subject Management Page

Teacher Edit Page

The teacher edit page allows administrators to modify existing teacher profiles, update their details, and adjust their roles, availability, and preferences for effective teaching management.

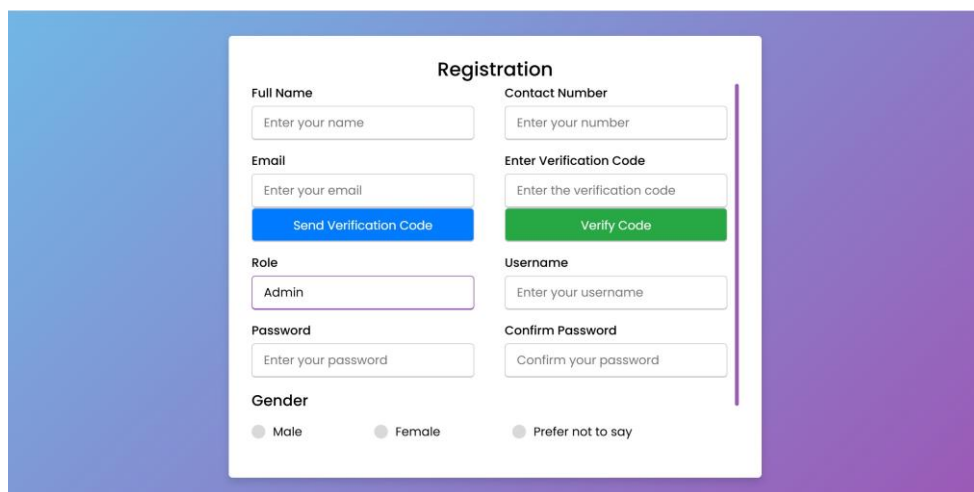


The screenshot shows the 'Edit Teacher' page. On the left is a dark sidebar with navigation links: 'Dashboard', 'Manage Teachers', 'Manage Subjects', 'View Routines', and 'Logout'. The main content area has a title 'Edit Teacher' and several form fields: 'Name' (text input with 'Anup Shrestha'), 'Designation' (dropdown menu with 'Teacher'), 'Type' (dropdown menu with 'Full Time'), and 'Course Codes' (text input with 'CT501'). There are 'Add Course' and 'Remove' buttons next to the course codes. At the bottom is a large blue 'Save Change' button.

Figure 9.11: Teacher Edit Page

Sign Up Page

The sign-up page enables users to register with a structured form. It captures essential details, including Full Name, Email, Role, Contact Number, and Password. Users verify emails via "Send Verification Code" and "Verify Code" buttons. The default role is "Admin," and a unique Username is required. Password confirmation and gender selection ensure security and customization.



The screenshot shows the 'Registration' form. It has two columns of fields. The left column includes: 'Full Name' (text input), 'Email' (text input with a 'Send Verification Code' button below it), 'Role' (dropdown menu with 'Admin'), 'Password' (text input), and 'Gender' (radio buttons for 'Male', 'Female', and 'Prefer not to say'). The right column includes: 'Contact Number' (text input), 'Enter Verification Code' (text input with a 'Verify Code' button below it), 'Username' (text input), and 'Confirm Password' (text input). The form is set against a blue and purple gradient background.

Figure 9.12: Sign up Page

Routine of Computer Department

This page displays the generated routine for the computer department, showing the allocated classes, times, and rooms for the courses in an organized and easy-to-read format.

Generated Timetable

Faculty: Computer Semester: 3											
Download Timetable as PDF											
Day	07:15 AM - 08:00 AM	08:00 AM - 08:45 AM	08:45 AM - 09:30 AM	09:30 AM - 10:15 AM	10:15 AM - 11:00 AM	11:00 AM - 11:45 AM	11:45 AM - 12:30 PM	12:30 PM - 01:15 PM	01:15 PM - 02:00 PM	02:00 PM - 02:45 PM	02:45 PM - 03:30 PM
Monday	Digital Logic Mr. Bimal	Digital Logic Mr. Bimal	Theory of Computation Prof. Kriti Bajracharya	Theory of Computation Prof. Kriti Bajracharya	BREAK	Electric Circuit Theory Pradip Adhikaree	Electric Circuit Theory Pradip Adhikaree	Digital Logic Ms. Rina Electric Circuit Theory Mr. Raj	Digital Logic Ms. Rina Electric Circuit Theory Mr. Raj	Digital Logic Ms. Rina Electric Circuit Theory Mr. Raj	Digital Logic Ms. Rina Electric Circuit Theory Mr. Raj
Tuesday	Digital Logic Mr. Bimal	Digital Logic Mr. Bimal	Theory of Computation Prof. Kriti Bajracharya	Theory of Computation Prof. Kriti Bajracharya	BREAK	Electric Circuit Theory Pradip Adhikaree	Electric Circuit Theory Pradip Adhikaree	Electromagnetics Mr. Mohan Object Oriented Programming Mr. Ram	Electromagnetics Mr. Mohan Object Oriented Programming Mr. Ram	Electromagnetics Mr. Mohan Object Oriented Programming Mr. Ram	Electromagnetics Mr. Mohan Object Oriented Programming Mr. Ram
Wednesday	Theory of Computation Prof. Kriti Bajracharya	Theory of Computation Prof. Kriti Bajracharya	Electric Circuit Theory Pradip Adhikaree	Electric Circuit Theory Pradip Adhikaree	BREAK	Object Oriented Programming Prof. Anup	Object Oriented Programming Prof. Anup	Digital Logic Ms. Rina Electronic Device & Circuits Ms. Anjali	Digital Logic Ms. Rina Electronic Device & Circuits Ms. Anjali	Digital Logic Ms. Rina Electronic Device & Circuits Ms. Anjali	Digital Logic Ms. Rina Electronic Device & Circuits Ms. Anjali
Thursday	Digital Logic Mr. Bimal	Digital Logic Mr. Bimal	Electric Circuit Theory MBD	Electric Circuit Theory MBD	BREAK	Library Hours	Theory of Computation Sharmila Bista	Object Oriented Programming Mr. Ram	Object Oriented Programming Mr. Ram	Object Oriented Programming Mr. Ram	Object Oriented Programming Mr. Ram
Friday	Object Oriented Programming Prof. Anup	Object Oriented Programming Prof. Anup	Digital Logic Mr. Bimal	Digital Logic Mr. Bimal	BREAK	Object Oriented Programming Dr. Ravi Kumar	Object Oriented Programming Dr. Ravi Kumar	Engineering Mathematics III GBJ	Digital Logic Ms. Rina	Digital Logic Ms. Rina	Library Hours

Figure 9.13: Generated Timetable

Mr. Bimal's Routine											
Back to Teacher List										Download as PDF	
Day	07:15 AM - 08:00 AM	08:00 AM - 08:45 AM	08:45 AM - 09:30 AM	09:30 AM - 10:15 AM	10:15 AM - 11:00 AM	11:00 AM - 11:45 AM	11:45 AM - 12:30 PM	12:30 PM - 01:15 PM	01:15 PM - 02:00 PM	02:00 PM - 02:45 PM	02:45 PM - 03:30 PM
Monday	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	-	-	-	-	-	-	-	-	-
Tuesday	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	-	-	-	-	-	-	-	-	-
Wednesday	-	-	-	-	-	-	-	-	-	-	-
Thursday	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	-	-	-	-	-	-	-	-	-
Friday	-	-	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	-	-	-	-	-	-	-

Figure 9.14.1: Individual Teacher Routine

Ms. Rina's Routine											
Back to Teacher List										Download as PDF	
Day	07:15 AM - 08:00 AM	08:00 AM - 08:45 AM	08:45 AM - 09:30 AM	09:30 AM - 10:15 AM	10:15 AM - 11:00 AM	11:00 AM - 11:45 AM	11:45 AM - 12:30 PM	12:30 PM - 01:15 PM	01:15 PM - 02:00 PM	02:00 PM - 02:45 PM	02:45 PM - 03:30 PM
Monday	-	-	-	-	-	-	-	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)
Tuesday	-	-	-	-	-	-	-	-	-	-	-
Wednesday	-	-	-	-	-	-	-	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)
Thursday	-	-	-	-	-	-	-	-	-	-	-
Friday	-	-	-	-	-	-	-	-	Ex 502 (3 - Computer)	Ex 502 (3 - Computer)	-

Figure 9.14.2: Individual Teacher Routine

System Training Data Flow Graph

A diagram representing how data is collected, processed, and utilized for timetable generation, ensuring continuous learning and improvements.

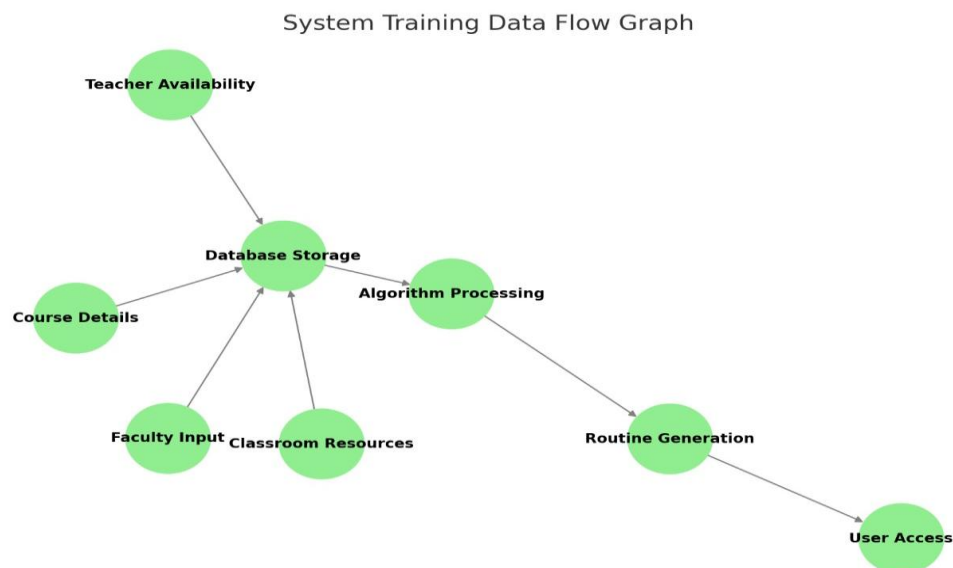


Figure 9.15: System Training Data Flow Graph

--	--	--	--	--	--	--	--

Constraints

The constraints that we treat are classified as hard or soft. Hard constraints are those to which a timetable has to adhere in order to be satisfied. A timetable is not viable if it does not satisfy them. Soft constraints are also those which should not be violated, but if this is not possible then the resulting timetable isn't optimal.

Hard Constraints

1. Teacher Workload Constraints

- **Maximum Workload Handling:** A teacher cannot exceed their maximum workload. If no other teacher is available, the admin must manually adjust.

- **Non-Classroom Activities Exclusion:** Non-classroom activities (such as supervision or meetings) are not considered in workload calculations.
- **Part-Time vs. Full-Time Availability:** Part-time teachers can override full-time teachers for both theory and practical classes, but only if they are regularly available.
- **Workload Formula:** The formula to calculate workload is:

$$\text{Workload} = (\text{Theory Classes} \times 1) + (\text{Practical Classes} \times 3 \times 0.8)$$

Project supervision and similar special cases are excluded from workload calculations.

Designation	Min Workload (period)	Max Workload (period)
CEO	2	3
Principal	2	3
Vice Principal	11	14
HOD	13	17
DHOD	15	19
Full-Time Teacher	19	25

Table 9.1: Workload distribution by designation

2. Subject Distribution Constraints

- **Theory Classes:** Combine Lecture (L) and Tutorial (T) periods to count as theory classes, and allocate one extra credit hour to theory classes (not to practical classes).
- **Optional Tutorials:** Tutorials are assigned lecture periods as theory classes.
- **Unallocatable Periods:** If extra credit hours cannot be scheduled, prioritize the explicit Lecture (L) and Tutorial (T) periods for allocation.

3. Practical Class Allocation Constraints

- **Practical Periods Allocation:**
 - For $P = 1.5$, assign one 3-period slot weekly.
 - For $P = 3$, assign two 3-period slots on different days.
- **Room Capacity:** Practical lab room capacity must match the subject's requirements for practical classes.
- **Multi-Batch Classes:** Block practical slots once a batch's session is assigned to avoid overlaps.
- **Teacher Pairing:** Pair one theory teacher with another eligible teacher for practical sessions, ensuring proper teacher coverage.

P type	Rooms Available
Hardware	2
Software	5
Field Visit	No room needed
English Lab	1
Thermodynamics Lab	1
Workshop	1
Chemistry Lab	1
Physics Lab	1
Drawing	2
Electrical Lab	2

Table 9.2: Room availability for different practical types

4. Scheduling and Rescheduling Constraints

- **Slot Blocking:** High-credit-hour subjects should be blocked first.
- **Conflict Resolution:** Prioritize rescheduling for important subjects and high teacher workloads. High-credit-hour subjects should not be displaced.
- **Cascading Conflicts:** During conflicts, high-credit-hour subjects should take precedence.

6. Special Cases and Constraints

- **Unresolved Constraints:** Any unresolved constraints must be flagged, notifying admins for manual intervention and suggesting reallocation strategies.
- **Frequent Violations:** Recurring violations should be flagged for long-term resolution.
- **Vacant Slots:** Attempts must be made to reallocate before leaving any slots vacant.
- **Practical Classes as Last Period:** Practical classes should ideally be scheduled as the last period of the day. If conflicts arise, they should be rescheduled.

7. Teacher Designation-Specific Constraints

- **Workload Variability by Designation:** Different teacher designations (e.g., HOD, Full-Time, Part-Time) have different workload limits.
- **Full-Time Teachers:** Must teach two subjects from different semesters within the same semester type.
- **Part-Time Teachers:** No workload cap applies. Their assignments are purely based on availability.

8. Error Handling Constraints

- **Unassigned Slots:** Unassigned slots should be logged for manual review or automatically adjusted if possible.

- **Teacher Absence:** If a teacher is absent, leave slots vacant and notify the admin for further action.
- **Practical Room Allocation:** Practical lab assignments must be rotated to manage room shortages, ensuring no single lab room is over-utilized.

10. Practical Scheduling Constraints

- **No Overlapping Lab Sessions:** Practical lab sessions of the same type must not overlap in the schedule.
- **Balanced Lab Room Utilization:** Practical rooms must be utilized efficiently across the week, avoiding room shortages or overbookings.

Soft Constraints

- (a) A classroom is not assigned to more than one teacher at the same time.
- (b) A class should be divided into groups and each groups is assigned with lab class in different time slot.
- (c) Lab class should be assigned after the theory class.
- (d) Lab class of 25 mark should be assigned in alternate weeks.
- (e) Lab class of 50 mark should be assigned in regular weeks.
- (f) More priority should be give to the part time teacher.
- (g) The subject should not be allocated to a time period inconvenient for a lecturer.

9.4 Outcome

The system offers advanced tools for managing teachers, subjects, and related information to ensure accurate routine generation across various departments. Through the Teacher Management Page, administrators can create or update teacher profiles, set availability, and define roles, while the Subject Management Page allows for managing subject details such as credit hours and categorization. This data is used to generate timetables for multiple departments, including Computer, Electronics, and Civil, as displayed in the respective routine pages, which present well-organized schedules for all relevant faculties. New users can easily register through the Sign Up Page, which features a secure email verification process to maintain data integrity and ensure only authorized users gain access. Additionally, the platform includes a User Management Page that enables administrators to manage user profiles by adding, editing, or deleting accounts, assigning roles, and overseeing system usage efficiently.

10. Future Enhancements

Up to this point, we have successfully developed and implemented the routine scheduling system for the Computer Department, but it is currently limited to only one semester. The same algorithm is designed to be scalable and will be applied to generate schedules for the remaining semesters in the future. However, one of the key limitations we are facing is the incomplete integration of practical lab constraints. As a result, the current implementation may not be able to handle all practical session requirements efficiently, which is an area that requires further refinement and improvement.

References

- [1] Lim Ying Ying and Hazinah Kutty Mammi. Timetable scheduling system using genetic algorithm (tsuga). *at IJIC (International Journal of Innovative Computing) Volume.*
- [2] Mayuri R Bagul, Sunil C Chaudhari, Sunita N Nagare, Pushkar R Patil, and KS Kumavat. A novel approach for automatic timetable generation. *International Journal of Computer Applications*, 975:8887, 2015.
- [3] Ratul Prosad, Md Ashikur Rahman Khan, and Ishtiaq Ahammad. Design of class routine and exam hall invigilation system based on genetic algorithm and greedy approach. *Asian Journal of Research in Computer Science*, 13(3):28–44, 2022.
- [4] Mansi Savdekar, Chaitali Bornare, Kailas Birari, and Kanchan Kolhe. Timetable generation system.
- [5] Maryam Alwashahi. Investigation and optimization of scheduling system in sohar university using genetic algorithm (ga). *International Journal of Computer Applications*, 126(11), 2015.
- [6] S Alagu Lakshmi, Ms S SUJITHA, and N DURGA ME. Time table automation system using backtracking technique. *International Journal of Advanced Research in Biology Engineering Science and Technology (IJARBEST) Vol, 2.*
- [7] Ashiqur Md Rahman, Sheik Shafaat Giasuddin, and Rashedur M Rahman. Decision tree based routine generation (drg) algorithm: A data mining advancement to generate academic routine and exam-time tabling for open credit system. *J. Comput.*, 5(1):12–22, 2010.
- [8] Sneha N T Sweta Jadhav Tahura Sadaf Prof. Jyothi Patil, Shambhavi V. Automatic timetable generator. *Ijrasnet Journal For Research in Applied Science and Engineering Technology*, 2023.

- [9] Nikita Veer Vaishanavi Ugale Priyanka Agrawal Rutuja Kavade, So-hail Qureshi. Smart timetable system using ai and ml. *International Journal of Creative Research Thoughts(IJCRT)*, 2023.
- [10] asc timetables. <https://www.asctimetables.com/>. Accessed: 2024-06-10.