MVJ COLLEGE OF ENGINEERING
Since 1982
An Autonomous Institute
(Affiliated to Visvesvaraya Technological University, Belagavi
Approved By AICTE, New Delhi,
Recognized by UGC under 2(f) & 12(B)
Accredited by NBA and NAAC)

PROJECT REVIEW REPORT ON "TEACHER'S AUTOMATIC TIMETABLE GENERATOR USING PHP"

Submitted in partial fulfilment of the requirements for the award of degree of Bachelor of Engineering in

COMPUTER SCIENCE & ENGINEERING

Submitted by

NITIN KUMAR JANGIR	1MJ19CS109
M R POOJA	1MJ19CS087
NIKETH N	1MJ19CS104
KAVYA M L	1MJ19CS077

Under the Guidance of
Mrs. K. L. Sujitha
Assistant Professor
Department of Computer Science & Engineering

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MVJ COLLEGE OF ENGINEERING
BANGALORE-67
2022-23

COLLEGE OF
ENGINEERING
Since 1982

Engineering A Better Tomorrow

An Autonomous Institute

MVJ COLLEGE OF ENGINEERING

Whitefield, Near ITPB, Bangre-67

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that phase-II of the project work, entitled **“TEACHER’S AUTOMATIC TIMETABLE GENERATOR USING PHP”** is a bonafide work carried out by

NITIN KUMAR JANGIR	**1MJ19CS109**
M R POOJA	**1MJ19CS087**
NIKETH N	**1MJ19CS104**
KAVYA M L	**1MJ19CS077**

in partial fulfilment for the award of degree of Bachelor of Engineering in Computer Science & Engineering of the Visvesvaraya Technological University, Belagavi during the academic year 2022-23. It is certified that all the corrections/suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements.

Signature of the Guide
(Mrs. K. L. Sujitha)

Signature of the HOD
(Dr. Kiran Babu T. S)

Signature of the Principal
(Dr. Suresh Babu V)

Name of the Examiners:

Signature with Date

1.
2.

An Autonomous Institute

MVJ COLLEGE OF ENGINEERING

Whitefield, Near ITPB, Bangre-67

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We,

NITIN KUMAR JANGIR	**1MJ19CS109**
M R POOJA	**1MJ19CS087**
NIKETH N	**1MJ19CS104**
KAVYA M L	**1MJ19CS077**

hereby declare that the entire Phase-II work of the project titled **“TEACHER’S AUTOMATIC TIMETABLE GENERATOR USING PHP”** embodied in this project report has been carried out by us during the 8th semester of B.E. degree at MVJCE, Bangalore under the esteemed guidance of **Mrs. K. L. Sujitha,** Assistant Professor, Dept. of CSE, MVJCE affiliated to Visvesvaraya Technological University, Belagavi. The work embodied in this dissertation work is original and it has not been submitted in part or full for any other degree in any University.

Place: MVJCE, Bangalore
Date:

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of people who made it possible, success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance. So, with gratitude we acknowledge all those whose guidance and encouragement served as beacon of light and crowned our effort with success.

We are thankful to our Principal **Dr. Suresh Babu V,** for his encouragement and support throughout the project work.

We are thankful to our Vice Principal **Dr. Brindha M**, for her encouragement and support throughout the project work.

We are thankful to our COE **Dr. M A Lourdu Anthony Raj**, for her encouragement and support throughout the project work.

We are also thankful to our HOD, **Dr. Kiran Babu T. S, Dept. of CSE** for his incessant encouragement & all the help during the project work.

We consider it a privilege and honour to express our sincere gratitude to our guide **Mrs. K. L. Sujitha, Assistant Professor, Dept. of CSE** for her valuable guidance throughout the tenure of this project work, and whose support and encouragement made this work possible.

It is also an immense pleasure to express our deepest gratitude to all faculty members of our department for their cooperation and constructive criticism offered, which helped us a lot during our project work.

Finally, we would like to thank all our family members and friends whose encouragement and support was invaluable.

Thanking you

ABSTRACT

Teacher's Automatic Timetable Generator is used to generate Timetable Automatically. At present we are using a manual system of preparing timetable. Here the manual system means the teacher needs to prepare the timetable which is very much time-consuming. So, this timetable generation system helps to manage all the periods automatically and also will be helpful for faculties to get automatic timetable in their phone by using this application. This timetable generator software also manages timetable when a teacher is absent, late coming or early going. Maximum and minimum workload for a faculty for a day, week and month will be specified for the efficient generation of timetable. This system has 2 modules namely Admin and Faculty. Admin has authority to manage course, faculty, classroom, create timetable, set lecture time slot and set subject time. Faculty has authority to login, view lectures, they get lecture reminder through SMS on their phone.

Table of Contents

List of Figures

CHAPTER 1

INTRODUCTION

Although the majority of faculty organization work has been mechanized, the lecture timetable preparation continues to be usually done manually taking lots of effort and time-consuming. It is widely used in schools as well as colleges and other fields of teaching. Other cases that can cause a problem is when the faculty number is less, the result's in rescheduling of time table or they need to fill on empty seats urgently. suppose a case in which a teacher of a particular class for some subject is absent, then the class advisor of that class will arrange some other lecture for that period. But at present in the existing system, she must check which teacher is free during that period which means all the work is done manually which leads to a waste of time of the class advisor as well as of students. While scheduling a timetable, even the smallest condition can take a lot of time and the case is even worse when the number of conditions/constraints or the amount of data to handle is more. In such cases Automated timetable, scheduling can be a very convenient method for managing such problems in computers with algorithms and proving to be eco-friendly for no paperwork. Most colleges have several different courses, and each course has several subjects. Now there are limited faculties and each faculty is teaching more than one subject. So now the timetable needed to b schedule the faculty at provided time slots in such a way that their timings do not overlap and the timetable schedule makes the best use of all faculty subject demands. We use a customized algorithm for this purpose. In our timetable generation algorithm, we proposed to utilize a timetable object.

CHAPTER 2

LITERATURE SURVEY

2.1 AUTOMATED TIMETABLE GENERATOR USING PARTICLE SWARM OPTIMIZATION

AUTHORS: Andeep Kumar, Kawaljeet Singh, Neeraj Sharma.

The timetabling problem at universities is an NP-hard problem under multiple constraints and limited resources. Thus a technique that can handle constraints is needed to optimize the problem. This focuses on Particle Swarm Optimization (PSO) for finding optimal solutions to the problem of course Timetabling. PSO is a promising scheme for solving NP - hard problems due to its fast convergence and fewer parameter settings. There are two objectives in this. First provide a detailed introduction to the topic of timetabling, Particle Swarm Optimization their method and their variations. The second objective is to apply them to the problem of Course Timetabling. The proposed algorithm is tested using the timetabling data from the College.

Methodology - Particle Swarm Optimization (PSO).

2.2 TIMETABLE GENERATOR USING TABU SEARCH

AUTHORS: Tanzila Islam, Zunayed Shahriar, Mohammad Anower Perves, Monirul Hasan.

This search is based on Timetable Generator by using Tabu Search algorithm. It helps to generate a course schedule and an exam schedule for a University. Every university faces a different set of problem while preparing course schedule and exam schedule. There are lots of constraints while making a scheduler. And for this reason, students suffer much as well as faculties. This report is based on discussion about an automated timetable generator for a university by using Tabu Search algorithm. Tabu Search is a meta-heuristic procedure for solving optimization problems. Tabu Search deals with a sub-optimal initial solution. By analysing the search space and averts inessential exploration, it optimists this solution and keeps the list of recently visited area in a Tabu list. This helps to solve these problems within a reasonable time and gives a feasible solution than any manual system. For a University, we have found that preparing exam schedule, course schedule, student assessment, room assignment with required resources are quite complex. But for all of them, we analyse that the Tabu Search technique is an essential method for getting a feasible solution. In this paper, we describe how Tabu Search works and how to get a feasible solution by using this algorithm.

Methodology – Tabu Search Algorithm (TSA).

2.3 A NOVEL APPROACH FOR AUTOMATIC TIMETABLE GENERATION

AUTHORS: Mayuri R. Bagul, Sunil C. Chaudhari, Sunita N. Nagare, Pushkar R. Patil, K.S. Kumavat.

The manual system of preparing time table in colleges is very time consuming and tedious task which usually ends up with various classes clashing either at identical room or with same teachers having more than one class at a time. Due to manual approach, proper use of resources is neither effective nor efficient. To overcome all these problems we propose to make an automated system with computer assisted timetable generator. The system will take various inputs like number of subjects, teachers, maximal lectures a teacher can conduct, priority of subject and topics to be covered in a week or a lecture, depending upon these inputs it will generate possible timetables for working days of the week, making optimal use of all resources in a way that will best suit the constraints. An appropriate timetable is then chosen from the optimal solutions generated.

Methodology - Resource Scheduling Algorithm (RSA).

2.4 TIMETABLE GENERATOR USING HEURISTIC TECHNIQUE

AUTHORS: Utkarsh Kumar, Yash Kamboj, Vaibhav Kumar , Rajesh Singh.

The manual means of getting ready timetable in schools is exceptionally tedious and drawn- out task which typically winds up with different classes conflicting either at indistinguishable room or with same educators having more than each class in turn. Because of manual methodology, legitimate utilization of assets is neither viable nor productive. To conquer this large number of issues we propose to make a mechanized framework with PC helped plan generator. The framework will take different sources of info like number of subjects, instructors, maximal talks an educator can direct, need of subject and points to be canvassed in a week or a talk, contingent on these information sources it will produce conceivable timetables for working days of the week, utilizing all assets such that will best suit the limitations. A suitable schedule is then looked over the ideal arrangements generated. Timetable creation is an extremely laborious and tedious assignment. To make plan it takes loads of persistence and worker hours. Timetable is made for different purposes like to arrange addresses in school and universities, to make timing diagrams for train and transport plan and some more. To make schedule it requires loads of time and labour supply. In our paper we have attempted to diminish these hardships of producing plan by Heuristic Algorithm.

Methodology – Heuristic Algorithm (HA).

CHAPTER 3

PROBLEM IDENTIFICATION & PROPOSED SOLUTION

3.1 Existing System

- In the Existing System, the whole process of making a timetable is done manually by taking care of all the possible constraints high as well as small constraints. The most tedious task of preparing a timetable is from the educational system, especially the University Timetabling. Here a specific teacher/lecturer is charged with the responsibility of creating an optimal timetable manually, the teacher/lecturer needs to take care of all possible constraints like should have the proper information of the total count of students, faculty members, proper time arrangement, etc. In case of teacher absence, he/she who is responsible for timetabling needs to arrange another lecture at that time, which means a single person individually is handling so much responsibility that is very much time consuming as well as a hectic work for the lecturer.

3.1.1 Demerits of Existing System

The existing system has a lot of disadvantages as it is done manually. Few disadvantages are listed below.

- Done manually.
- Increases paperwork.
- High chances of errors.
- Time taking process.
- Very Confusing.

3.2 Proposed System

3.2.1 Description

The process starts with the collection of datasets. For the dataset, it will first take the input from the user regarding the information related to the timetable such as "Courses, Subjects, labs, Semester" etc. The next step is to apply certain Rules/Constraints to the user input. E.g.- If we are taking input for lectures so the system should look after the clashes issues such that there should not be the same lectures for the same faculty at the same time. After this, all the constraints and possibilities are being verified which further leads to timetable generation. Now, the final timetable will get generated. In the further process, the user will review the generated timetable and suppose wants to edit the timetable as he/she is not satisfied with the generated one then the user can regenerate the timetable again. After this, a final regenerated timetable will get generated. And in the final process, the timetable will be viewed by the user. Even if the teacher is absent, we can generate the timetable according to that. Messages will also be sent to the respective teachers for the class they have.

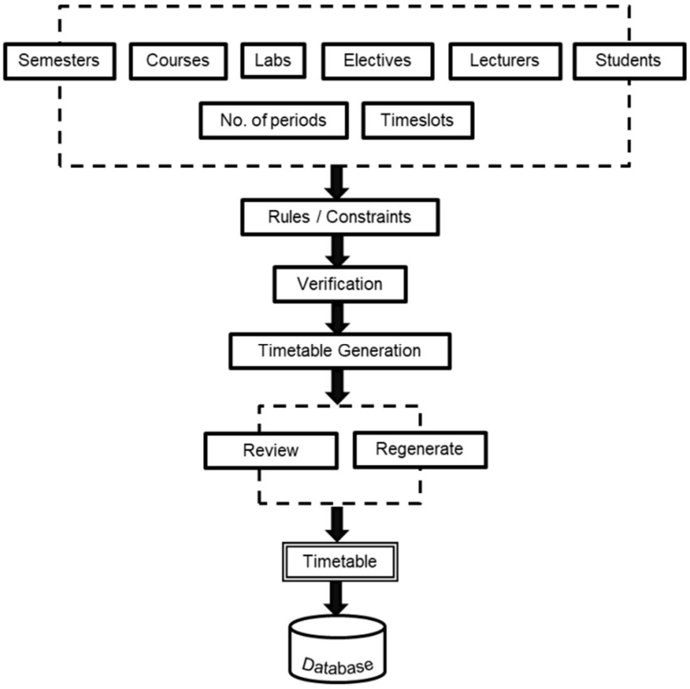


Fig.3.1 System Architecture

Structure of timetable generator consists of input data, relation between the input data, system constraints and application of genetics algorithm.

A. Input Data The input data contains:

1. Professor: Data describes the name of lecturers along with their identification number.
2. Subject: Data describes the name of courses in the current term.
3. Room: Data describes the room number and their capacity.
4. Time intervals: It indicates starting time along with duration of a lecture.

B. System Constraints System constraints are divided into 2 categories:

1. Hard Constraints: The timetable is subjected to the following four types of hard constraints, which must be satisfied by a solution to be considered as a valid one:

    a. A student should have only one class at a Time.

    b. A Teacher should have only one class at a time.

    c. A room should be booked only for one class at a time.

2. Soft Constraints: These are the constraints that are of no great concern but are still taken into contemplation. They don't need to be satisfied but the solutions are generally considered to be good if they are satisfied.

    a. Courses must be eventually distributed.

    b. Scheduling of teachers should be well spread over the week.

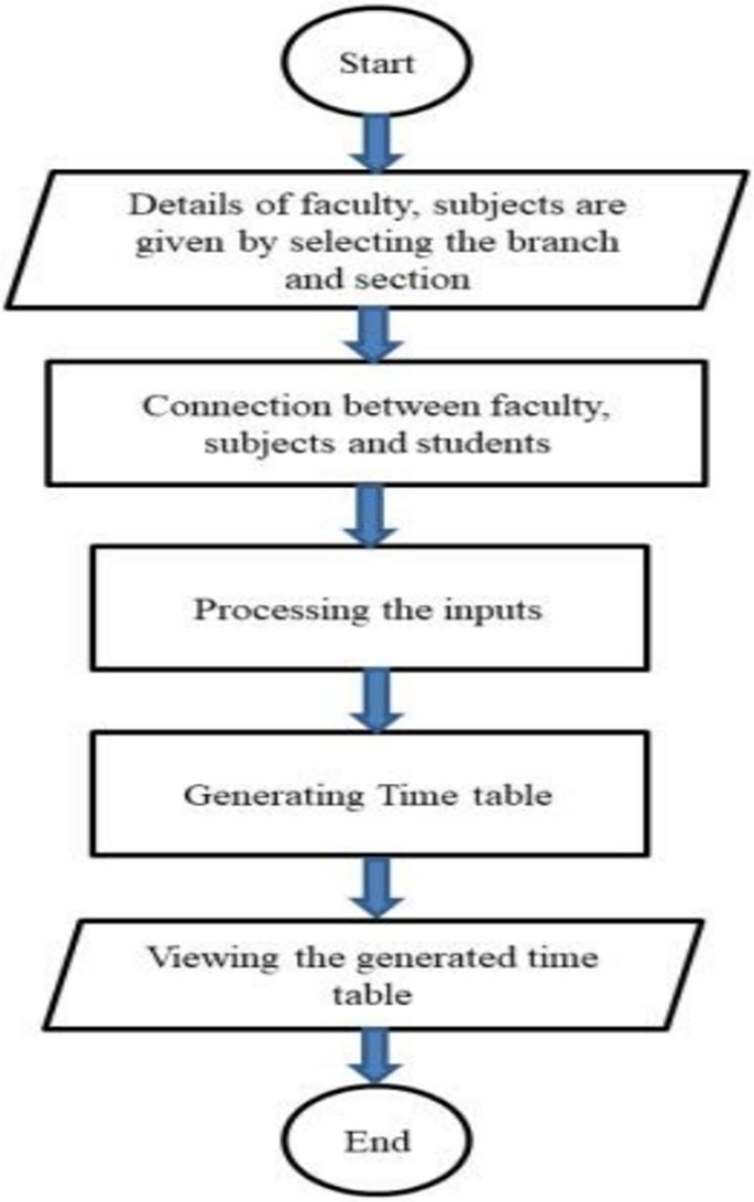

Start
Details of faculty, subjects are given by selecting the branch and section
Connection between faculty, subjects and students
Processing the inputs
Generating Time table
Viewing the generated time table
End

Fig 3.2 Flow Chart Diagram

3.2.2 Advantages of Proposed System

- Easier slot assigning.
- Less time consumption.
- No slot clashes.
- Always considers the other department slots first.
- Various possible slot combinations can be acquired.
- User friendly.

CHAPTER 4

OBJECTIVES & EXPECTED OUTCOME

4.1 Objective

- To develop an efficient and effective system that generates an accurate timetable based on user inputs.
- To save time and effort and reduce errors.
- To develop a User interface (UI) that provides smooth integration as well as a paperless environment, which is user-friendly, takes input from the user, and generates the timetable.

4.2 Scope

- To generate timetable for the institute which will be less time consuming and free of human errors along with high level of efficiency and precision.
- To improve the overall process of timetable generation with help of genetics algorithm along with the assistance of technology.

4.3 Methodology

As the flow chart explains the overall working of the TEACHER'S AUTOMATIC TIMETABLE GENERATOR System. It includes soft and hard constraints, input as student details, teacher details, and subject details. By using this collection of input from using the system will generate an optimized timetable Firstly the system contains Login Page. The User has to login first with the valid user-id and password which in turn opens the Home Page which contains the information about the TEACHER'S AUTOMATIC TIMETABLE GENERATOR System. It also has the various link tabs on the menu bar to navigate to other pages, but it will work if and only if the user is logged in to the system. Once the user gets logged in, he/she will get the approval to use the next page which is to add subjects where the subject name, semester, Timings, and other such required details input will be given by the user. As the details related to subjects and semester is filled by the user it will get reflected and saved into the database. Once the system gets the subject detail it will take the user to the next page where the user needs to add the faculty information i.e. the faculty name, which faculty will be taking which subject, and whether it will be a theory session or practical, etc. And these details will also get saved into our dataset. Now the system has all information related to subjects, teachers, semesters, Labs, Theory, timings, etc. So now it will ask the user from the given semester details for which semester the user wants to generate a timetable i.e., ODD Semester / EVEN Semester. All these data inputs will get saved and collected in our dataset and after this, the user will click on the generate button of the system, based on all the information and constraints the Automated timetable generating System will generate an optimized timetable in Excel Sheet and the user can download the sheet from the system into their device.

4.4 Expected Outcome

- To have a user-friendly Timetable Management System solution for teachers which helps in institutions.
- Making sure that the application acts responsive throughout the working process.
- Faculty details in department tells the details of the respective faculty.
- Subject details and classroom details are stored.
- It will also manage the time when the faculty is absent.
- There is no need for teachers to worry about Timings.

CHAPTER 5

ABOUT THE ALGORITHM

5.1 Genetic Algorithm

Genetic algorithms(GAs) are adaptive heuristic search algorithms that belong to the larger part of evolutionary algorithms. Genetic algorithms are based on the ideas of natural selection and genetics. These are intelligent exploitation of random search provided with historical data to direct the search into the region of better performance in solution space. They are commonly used to generate high-quality solutions for optimization problems and search problems.

Genetic algorithms simulate the process of natural selection which means those species who can adapt to changes in their environment are able to survive and reproduce and go to next generation. In simple words, they simulate "survival of the fittest" among individuals of consecutive generation for solving a problem. Each generation consists of a population of individuals and each individual represents a point in search space and possible solution. Each individual is represented as a string of character/integer/float/bits. This string is analogous to the Chromosome.

To summarise, the 5 main phases of a genetic algorithm are,

- Initial population - The first step in the functioning of a GA is the generation of an initial population. Each member of this population encodes a possible solution to a problem. After creating the initial population, each individual is evaluated and assigned a fitness value according to the fitness function. It has been recognized that if the initial population to the GA is good, then the algorithm has a better possibility of finding a good solution and that, if the initial supply of building blocks is not large enough or good enough, then it would be difficult for the algorithm to find a good solution.

- Fitness function - The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent, in particular, in the fields of genetic programming and genetic algorithms, each design solution is commonly represented as a string of numbers referred to as a chromosome. After each round of testing, or simulation, the idea is to delete the 'n' worst

- design solutions, and to breed 'n' new ones from the best design solutions. Each design solution, therefore, needs to be awarded a figure of merit, to indicate how close it came to meeting the overall specification, and this is generated by applying the fitness function to the test, or simulation, results obtained from that solution.

- Selection - This operator selects chromosomes in the population for reproduction. The fitter the chromosome, the more times it is likely to be selected to reproduce.

- Crossover - In genetic algorithms, crossover is a genetic operator used to vary the programming of a chromosome or chromosomes from one generation to the next. It is analogous to reproduction and biological crossover, upon which genetic algorithms are based. Cross over is a process of taking more than one parent solutions and producing a child solution from them. There are methods for selection of the chromosomes. This operator randomly chooses a locus and exchanges the sub sequences before and after that locus between two chromosomes to create two offspring.

- Mutation - Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. It is analogous to biological mutation. Mutation alters one or more gene values in a chromosome from its initial state. In mutation, the solution may change entirely from the previous solution. Hence GA can come to better solution by using mutation. This operator randomly flips some of the bits in a chromosome.

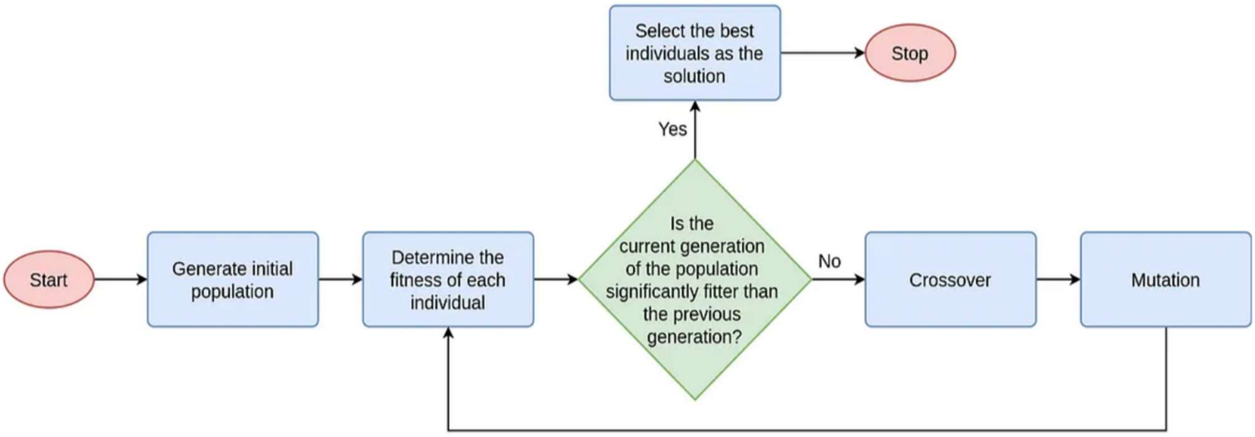

Select the best individuals as the solution
Stop
Yes
Start
Generate initial population
Determine the fitness of each individual
Is the current generation of the population significantly fitter than the previous generation?
No
Crossover
Mutation

Figure 5.1 Genetic Algorithm Flow Diagram

5.2 Working of Genetic Algorithm

Consider you are trying to come up with a weekly timetable for classes in a college for different groups of students. We have to arrange classes and come up with a timetable so that there are no clashes between classes. Here, our task is to search for the optimum timetable schedule.

Creating Chromosomes

An individual is characterized by a set of parameters (variables) known as genes. Genes are joined into a string to form a chromosome (solution). We model our solutions as chromosomes.

In timetable generation, chromosomes will be different lecture sessions for students taking different modules. Consider that we have to coordinate student groups, modules, lecture halls, the days of the week and time. You can represent a lecture session as,

<Module, Student Group, Day, Location, Lecture Time>

We can encode the lecture session as a binary pattern to a chromosome.

You can give binary values for each value in each entity. You can change the encoding pattern as you wish. Given below is an example way you can encode the lecture session.

Get the list of modules and assign binary values.

Data Mining - 0000,

Machine Learning - 0001,

Biology - 0010,

...

Get the list of student groups and give binary values.

STG0 - 00000,

STG1 - 00001,

STG2 - 00010,

STG3 - 00011,...

Similarly, you can come up with coding schemes for every entity in the lecture session.

Given below is a sample encoding of a lecture session.

<Data Mining, STG3, Monday, Hall D, 8.00AM>

Data Mining - 0000

STG3 - 00011

Monday - 000

Hall D - 1010

8.00AM - 1000

Chromosome - 00000001100010101000

Individual bits are called genes. This chromosome has 20 genes.

Creating an Initial Population

Different student groups take different classes within a week. Hence, you have to come up with different class combinations and create the initial population. You can decide upon the size of the population (number of classes).

<Data Mining, STG3, Monday, Hall A, 8.00AM>

<Machine Learning, STG2, Tuesday, Hall B, 8.00AM>

<Computational Biology, STG8, Tuesday, Hall A, 10.00AM>

...

You have to encode these classes into chromosomes as mentioned before.

Coming up with an Evaluation Function

As an example, you can formulate the evaluation function as the inverse of the number of class conflicts for student groups. Lesser the number of conflicts, more fit the lecture session is. You can pick a suitable evaluation function as you wish.

Now you can perform crossover and mutation operations to maximize the fitness value for each lecture session.

Termination

You can terminate the process when the population has reached the maximum fitness value. In our example, that is when the lecture sessions have no conflicts.

Chapter 6

SYSTEM REQUIREMENTS

6.1 Hardware Requirements

➢ Processors	:	Pentium IV 2.4 GHz
➢ RAM	:	4 GB (min)
➢ Storage	:	20GB
➢ Standard Devices	:	Webcam, Keyboard, Monitor and Mouse

6.2 Software Requirements

For developing the application, the following are the Software Requirements:

- Xampp Control Panel.
- Visual Studio Code.
- MySQL.
- Operating Systems supported: Windows 7/ 8/9/10/11.

6.3 Technologies and Languages used to Develop

- HTML.
- CSS.
- JavaScript.
- MySQL.
- PHP.

CHAPTER 7

CODE SNIPPETS

7.1 Login Page

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Styles -->
    @include('partials.styles')
    @yield('styles')

    <title>Sign In | {{env("APP_NAME")}}</title>
  </head>

  <body class="login-page">
    <div class="container">
      <div class="row">
        <div class="col-xs-12 col-md-4 col-sm-8 col-lg-4 col-md-offset-4 col-sm-offset-2 col-
lg-offset-4">
          <div id="login-form-container">
            <div class="login-form-header">
              <h3 class="text-center">Login</h3>
            </div>
            <div class="login-form-body">
              <div class="row">
```

```
<div class="col-xs-12 col-sm-12 col-md-12 col-lg-12">
    <form method="POST" action="{{ URL::to('/login') }}">
      {!! csrf_field() !!}
      @include('errors.form_errors')
      <div class="form-group">
         <label>Username</label>
             <input type="text" class="form-control" placeholder="Username" name="username">
      </div>

      <div class="form-group">
         <label>Password</label>
       <input type="password" class="form-control" placeholder="Password" name="password">
      </div>

      <div class="form-group">
         <input type="submit" name="submit" value="Sign in" class="btn btn-lg btn-block btn-custom">
      </div>

      <div class="form-group">
            <input type="submit" name="submit" value="Register" class="btn btn-lg btn-block btn-custom">
      </div>

      <div class="form-group">
     <a href="/request_reset" class="btn btn-lg btn-block btn-primary">Forgot Password?</a>
      </div>
   </form>
</div>
```

```
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <!-- Scripts -->
    @include('partials.scripts')
    @yield('scripts')
</body>
</html>
```

7.2 GeneticAlgorithm

```
<?php
namespace App\Services\GeneticAlgorithm;
class GeneticAlgorithm
{
    /**
     * This is the number of individuals in the population
     *
     * @var int
     */
    private $populationSize;

    /**
     * This is the probability in which a specific gene in a solution's
     * chromosome will be mutated
     *
     * @var double
     */
    private $mutationRate;
```

```
/**
 * This is the frequency in which crossover is applied
 *
 * @var double
 */
private $crossoverRate;

/**
 * This represents the number of individuals to be
 * considered as elite and skipped during crossover
 *
 * @var integer
 */
private $elitismCount;

/**
 * Size of the tournament
 *
 * @var int
 */
private $tournamentSize;

/**
 * Temperature for simulated annealing
 *
 * @var int
 */
private $temperature;

/**
 * Cooling rate for simulated annealing
```

```
     *
     * @var int
     */
    private $coolingRate;

    /**
     * Create a new instance of this class
     */
    public function __construct($populationSize, $mutationRate, $crossOverRate, $elitismCount,
$tournamentSize)
    {
        $this->populationSize = $populationSize;
        $this->mutationRate = $mutationRate;
        $this->crossoverRate = $crossOverRate;
        $this->elitismCount = $elitismCount;
        $this->tournamentSize = $tournamentSize;
        $this->temperature = 1.0;
        $this->coolingRate = 0.001;
    }

    /**
     * Initialize a population
     *
     * @param Timetable $timetable Timetable for generating individuals
     */
    public function initPopulation($timetable)
    {
        $population = new Population($this->populationSize, $timetable);

        return $population;
    }
```

```
/**
 * Get the temperature
 */
public function getTemperature()
{
    return $this->temperature;
}

/**
 * Cool temperature
 *
 */
public function coolTemperature()
{
    $this->temperature *= (1 - $this->coolingRate);
}

/**
 * Calculate the fitness of a given individual
 *
 * @param Individual $individual The individual
 * @param Timetable $timetable A timetable
 * @return double The fitness of the individual
 */
public function calculateFitness($individual, $timetable)
{
    $timetable = clone $timetable;

    $timetable->createClasses($individual);
    $clashes = $timetable->calcClashes();
    $fitness = 1.0 / ($clashes + 1);
    $individual->setFitness($fitness);
    return $fitness;
}

/**
 * Evaluate a given population
 *
 * @param Population $population The population to evaluate
 * @param Timetable $timetable Timetable data
 */
public function evaluatePopulation($population, $timetable)
{
    $populationFitness = 0;

    $individuals = $population->getIndividuals();

    foreach ($individuals as $individual) {
        $populationFitness += $this->calculateFitness($individual, $timetable);
    }

    $population->setPopulationFitness($populationFitness);
}

/**
 * Determine whether the termination condition has been met
 * For this problem, this occurs when we get an individual with
 * a fitness of 1.0
 *
 * @param Population $population Population we are evaluating
 * @return boolean The truth value of this check
 */
public function isTerminationConditionMet($population)
```

```
{
   return $population->getFittest(0)->getFitness() == 1;
}

/**
 * Determine whether we have reached the max generations we want to
 * iterate through
 *
 * @param int $generations Number of generations
 * @param int $maxGenerations Max generations
 */
public function isGenerationsMaxedOut($generations, $maxGenerations)
{
   return $generations > $maxGenerations;
}

/**
 * Select a parent from a population to be used in a crossover
 * with some other individual
 *
 * The technique used here is tournament selection method
 *
 * @param Population $population The population
 * @return Individual The selected parent
 */
public function selectParent($population)
{
   $tournament = new Population();

   $population->shuffle();

   for ($i = 0; $i < $this->tournamentSize; $i++) {
        $participant = $population->getIndividual($i);
        $tournament->setIndividual($i, $participant);
    }

    return $tournament->getFittest(0);
}

/**
 * Perform  a crossover on a population's individuals
 *
 * @param Population $population The population
 * @return Population $newPopulation The resulting population
 */
public function crossoverPopulation($population)
{
    $newPopulation = new Population($population->size());

    for ($i = 0; $i < $population->size(); $i++) {
        $parentA = $population->getFittest($i);

        $random = mt_rand() / mt_getrandmax();
        if (($this->crossoverRate > $random) && ($i > $this->elitismCount)) {
            // Initialise offspring
            $offspring = Individual::random($parentA->getChromosomeLength());

            $parentB = $this->selectParent($population);

            $swapPoint = mt_rand(0, $parentB->getChromosomeLength());

            for ($j = 0; $j < $parentA->getChromosomeLength(); $j++) {
                if ($j < $swapPoint) {
                    $offspring->setGene($j, $parentA->getGene($j));
                } else {
                    $offspring->setGene($j, $parentB->getGene($j));
                }
            }
            $newPopulation->setIndividual($i, $offspring);
        } else {
            // Add to population without crossover
            $newPopulation->setIndividual($i, $parentA);
        }
    }
    return $newPopulation;
}
/**
 * Perform a mutation on the individuals of the given population
 *
 * @param Population $population The population to mutate
 */
public function mutatePopulation($population, $timetable)
{
    $newPopulation = new Population();
    $bestFitness = $population->getFittest(0)->getFitness();

    for ($i = 0; $i < $population->size(); $i++) {
        $individual = $population->getFittest($i);
        $randomIndividual = new Individual($timetable);
        // Calculate adaptive mutation rate
        $adaptiveMutationRate = $this->mutationRate;

        if ($individual->getFitness() > $population->getAvgFitness()) {
            $fitnessDelta1 = $bestFitness - $individual->getFitness();
            $fitnessDelta2 = $bestFitness - $population->getAvgFitness();
            $adaptiveMutationRate = ($fitnessDelta1 / $fitnessDelta2) * $this->mutationRate;
            }

            if ($i > $this->elitismCount) {
                for ($j = 0; $j < $individual->getChromosomeLength(); $j++) {
                    $random = mt_rand() / mt_getrandmax();

                    if (($adaptiveMutationRate * $this->temperature) > $random) {
                        $individual->setGene($j, $randomIndividual->getGene($j));
                    }
                }
            }

            $newPopulation->setIndividual($i, $individual);
        }

        return $newPopulation;
    }
}
```

7.3 Timetables Controller

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Services\TimetableService;
use App\Events\TimetablesRequested;
use App\Models\CollegeClass;
use App\Models\Course;
use App\Models\Day;
use App\Models\Professor;
use App\Models\Room;
use App\Models\Timeslot;
use App\Models\Timetable;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Response;
use Illuminate\Support\Facades\Storage;

class TimetablesController extends Controller
{
    /**
     * Create a new instance of this controller and set up
     * middlewares on this controller methods
     */
    public function __construct(TimetableService $service)
    {
        $this->service = $service;
        $this->middleware('auth', ['except' => ['export']]);
        $this->middleware('activated', ['except' => ['export']]);
    }

    /**
     * Handle ajax request to load timetable to populate
     * timetables table on dashboard
     *
     */
    public function index()
    {
        $timetables = Timetable::orderBy('created_at', 'DESC')->paginate(10);

        return view('dashboard.timetables', compact('timetables'));
    }

    /**
     * Create a new timetable object and hand over to genetic algorithm
```

```
 * to generate
 *
 * @param Illuminate\Http\Request $request The HTTP request
 */
public function store(Request $request)
{
    $rules = [
        'name' => 'required',
        'academic_period_id' => 'required'
    ];

    $messages = [
        'academic_period_id.required' => 'An academic period must be selected'
    ];

    $this->validate(request(), $rules, $messages);

    $errors = [];
    $dayIds = [];

    $days = Day::all();

    foreach ($days as $day) {
        if ($request->has('day_' . $day->id)) {
            $dayIds[] = $day->id;
        }
    }

    if (!count($dayIds)) {
        $errors[] = 'At least one day should be selected';
    }
    if (count($errors)) {
        return Response::json(['errors' => $errors], 422);
    }

    $otherChecks = $this->service->checkCreationConditions();

    if (count($otherChecks)) {
        return Response::json(['errors' => $otherChecks], 422);
    }

    $timetable = Timetable::create([
        'user_id' => Auth::user()->id,
        'academic_period_id' => $request->academic_period_id,
        'status' => 'IN PROGRESS',
        'name' => $request->name
    ]);

    if ($timetable) {
        $timetable->days()->sync($dayIds);
    }

    event(new TimetablesRequested($timetable));

   return Response::json(['message' => 'Timetables are being generated.Check back later'], 200);
}

/**
 * Display a printable view of timetable set
 *
 * @param int $id
 */
public function view($id)
```

```
    {
        $timetable = Timetable::find($id);

        if (!$timetable) {
            return redirect('/');
        } else {
            $path = $timetable->file_url;
            $timetableData =  Storage::get($path);
            $timetableName = $timetable->name;
            return view('timetables.view', compact('timetableData', 'timetableName'));
        }
    }

    public function export()
    {

        $academicPeriodId = 1;
                        $groupIds     =     DB::table("courses_classes")->where("academic_period_id",
$academicPeriodId)->select("class_id")->get()->pluck("class_id");
                        $moduleIds     =     DB::table("courses_classes")->where("academic_period_id",
$academicPeriodId)->select("course_id")->get()->pluck("course_id");

        $professors = Professor::query()->with(['unavailable_timeslots'])->get()
            ->map(function ($item) {
                return [
                    "professorId" => $item->id,
                    "professorName" => $item->name,
                    "unavailableTimeslotIds" => $item->unavailable_timeslots->pluck('id')
                ];
            });
        $rooms = Room::query()->get()
            ->map(function ($item) {
```

```
        return [
            "roomId" => $item->id,
            "roomName" => $item->name,
            "capacity" => $item->capacity
        ];
    });
$timeslots =  Timeslot::query()->get()
    ->map(function ($item) {
        return [
            "timeslotId" => $item->id,
            "timeslot" => $item->time
        ];
    });
$groups = CollegeClass::query()->with([
    'unavailable_rooms',
    'courses' => function ($query)  use ($moduleIds) {
        return $query->whereIn("courses.id", $moduleIds);
    }
])->whereIn("id", $groupIds)->get()
    ->map(function ($item) {
        return [
            "groupId" => $item->id,
            "groupSize" => $item->size,
            "moduleIds" => $item->courses->pluck('id'),
            "unauthorizedRoomIds" => $item->unavailable_rooms->pluck('id'),
        ];
    });
$modules = Course::query()->with(['professors'])->whereIn("id", $moduleIds)->get()
    ->map(function ($item) {
        return [
            "moduleId" => $item->id,
            "moduleName" => $item->name,
                "moduleCode" => $item->course_code,
                "professorIds" => $item->professors->pluck('id')
            ];
        });
        return [
            "professors" => $professors,
            "rooms" => $rooms,
            "timeslots" => $timeslots,
            "groups" => $groups,
            "modules" => $modules
        ];
    }
}
```

7.4 Users Controller

```
<?php
namespace App\Http\Controllers;

use DB;
use Auth;
use Hash;
use Carbon\Carbon;
use App\Services\Helpers;
use Illuminate\Http\Request;

use App\Events\PasswordResetRequested;

use App\Models\User;
use App\Models\SecurityQuestion;

class UsersController extends Controller
{
    public function __construct()
    {
            $this->middleware('auth', ['only' => ['showAccountPage', 'showActivationPage',
'updateAccount' ]]);
    }
    /**
```

```
 * Show page for logging user in
 */
public function showLoginPage()
{
    return view('auth.login');
}

/**
 * Log in a user
 *
 * @param Illuminate\Http\Request $request The HTTP request
 */
public function loginUser(Request $request)
{
    $rules = [
        'password' => 'required'
    ];

    $this->validate($request, $rules);

    $user = User::first();

    if (!$user) {
        return redirect()->back()->withErrors(['No user account has been set up yet']);
    }

    if (!Hash::check($request->password, $user->password)) {
        return redirect()->back()->withErrors(['Password is invalid']);
    }

    Auth::login($user);

    return redirect('/');
}

/**
 * Show account activation page where new user can set up his
 * account
 *
 * @return Illuminate\Http\Response Account activation view
 */
public function showActivationPage()
{
    $user = Auth::user();
    $questions = SecurityQuestion::all();

    return view('users.activate', compact('user', 'questions'));
}
```

```
/**
 * Activate and set up account for user
 *
 * @param Illuminate\Http\Request $request The HTTP request
 * @return Illuminate\Http\Response Redirect to home page
 */
public function activateUser(Request $request)
{
    $user = Auth::user();

    if ($user->activated) {
        return redirect()->back()->withError('Your account is already activated');
    }

    $rules = [
        'name' => 'required',
        'password' => 'required|confirmed',
        'security_question_id' => 'required|exists:security_questions,id',
        'security_question_answer' => 'required'
    ];

    $messages = [
        'security_question_id.required' => 'A security question must be selected.',
        'security_question_answer.required' => 'Add an answer for security question.'
    ];

    $this->validate($request, $rules, $messages);

    $user->update([
        'name' => $request->name,
        'password' => bcrypt($request->password),
        'security_question_id' => $request->security_question_id,
        'security_question_answer' => $request->security_question_answer,
        'activated' => true
    ]);

    return redirect('/');
}

/**
 * Show the page to reuqest new password
 */
public function showPasswordRequestPage()
{
    $user = User::first();

    return view('users.password_request', compact('user'));
}

/**
 * Handle the request to reset password after user forgets
 * password
 *
 * @param Illuminate\Http\Request $request The HTTP request
 */
public function requestPassword(Request $request)
{
    $rules = [
        'email' => 'required|email',
        'security_question_answer' => 'required'
    ];

    $this->validate($request, $rules);

    $user = User::first();

    if ($user->security_question_answer != $request->security_question_answer) {
        return redirect()->back()->withErrors(['Your answer is not valid']);
    }

    $token = Helpers::generateRandomString();

    DB::table('password_resets')->delete();
    DB::table('password_resets')->insert([
        'user_id' => $user->id,
        'token' => $token,
        'expiry_date' => Carbon::now()->addDay()->toDateTimeString()
    ]);

    event(new PasswordResetRequested($token, $request->email));

    return redirect('/reset_password');
}

/**
 * Show page for password reset
 *
 */
public function showResetPassword()
{
    return view('users.password_reset');
}

/**
 * Handle reset of password
 */
public function resetPassword(Request $request)
{
    $token = DB::table('password_resets')->first();

    $rules = [
        'token' => 'required'
    ];

    $this->validate($request, $rules);

    if (!$token || ($token && $token->token != $request->token)) {
        return redirect()->back()->withErrors(['Invalid token']);
    }

    if (Carbon::now()->gt(Carbon::parse($token->expiry_date))) {
        return redirect()->back()->withErrors(['Token has expired.Please request for new token']);
    }

    $user = User::first();
    $user->update([
        'activated' => false
    ]);

    Auth::login($user);
    return redirect('/');
}

/**
 * Show account settings page
 *
 */
public function showAccountPage()
{
    $user = Auth::user();
    $questions = SecurityQuestion::all();

    return view('users.account', compact('user', 'questions'));
}

/**
 * Update user account
 *
 * @param Illuminate\Http\Request $request The HTTP request
 */
public function updateAccount(Request $request)
{
    $rules = [
            'name' => 'required',
            'security_question_id' => 'required',
            'security_question_answer' => 'required'
        ];

        if ($request->has('password') && $request->password) {
            $rules['password'] = 'confirmed';
            $rules['old_password'] = 'required';
        };

        $this->validate($request, $rules);

        $user = Auth::user();
        $data = [
            'name' => $request->name,
            'security_question_id' => $request->security_question_id,
            'security_question_answer' => $request->security_question_answer
        ];

        if ($request->has('password') && $request->password) {
            if (!Hash::check($request->old_password, $user->password)) {
                return redirect()->back()->withErrors(['Current password is invalid']);
            }

            $data['password'] = bcrypt($request->password);
        }

        $user->update($data);

        return redirect()->back()->with('status', 'Your account has been updated');
    }
}
```

CHAPTER 8

RESULTS

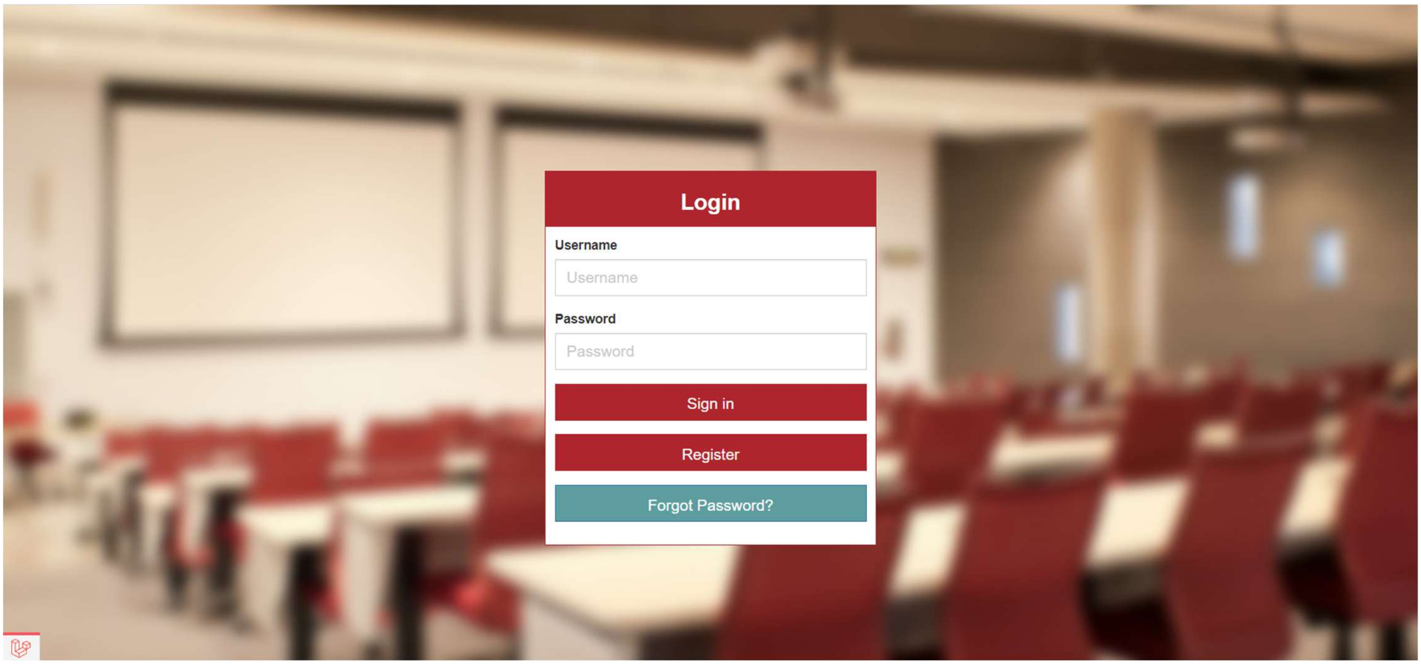
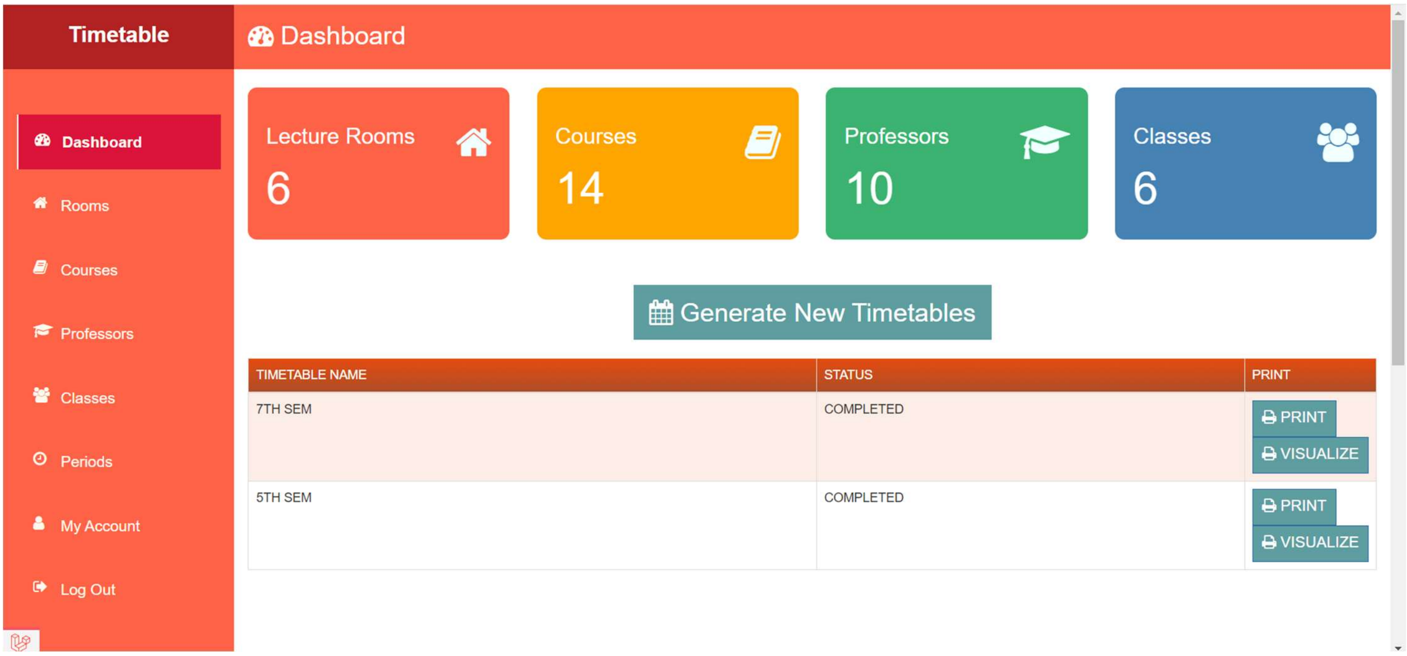
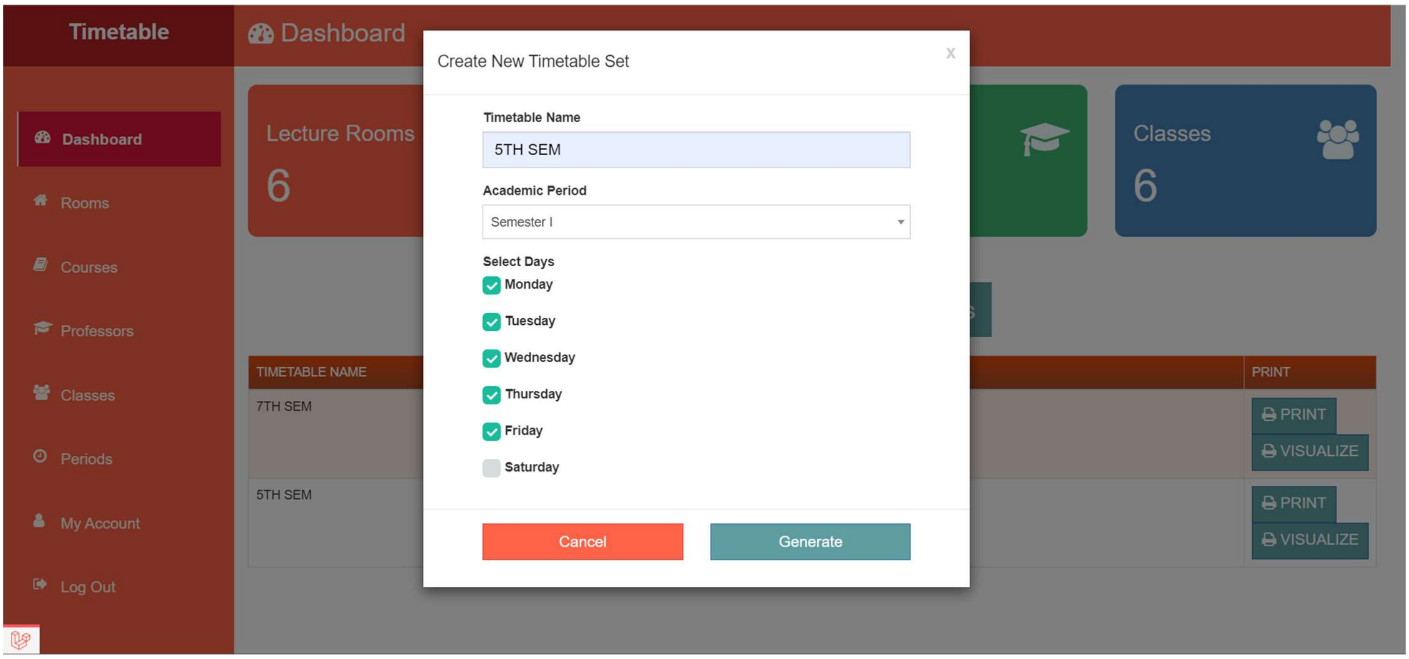


Figure 8.1 Login Page



TIMETABLE NAME	STATUS	PRINT
7TH SEM	COMPLETED	PRINT VISUALIZE
5TH SEM	COMPLETED	PRINT VISUALIZE

Figure 8.2 Admin Dashboard



Create New Timetable Set

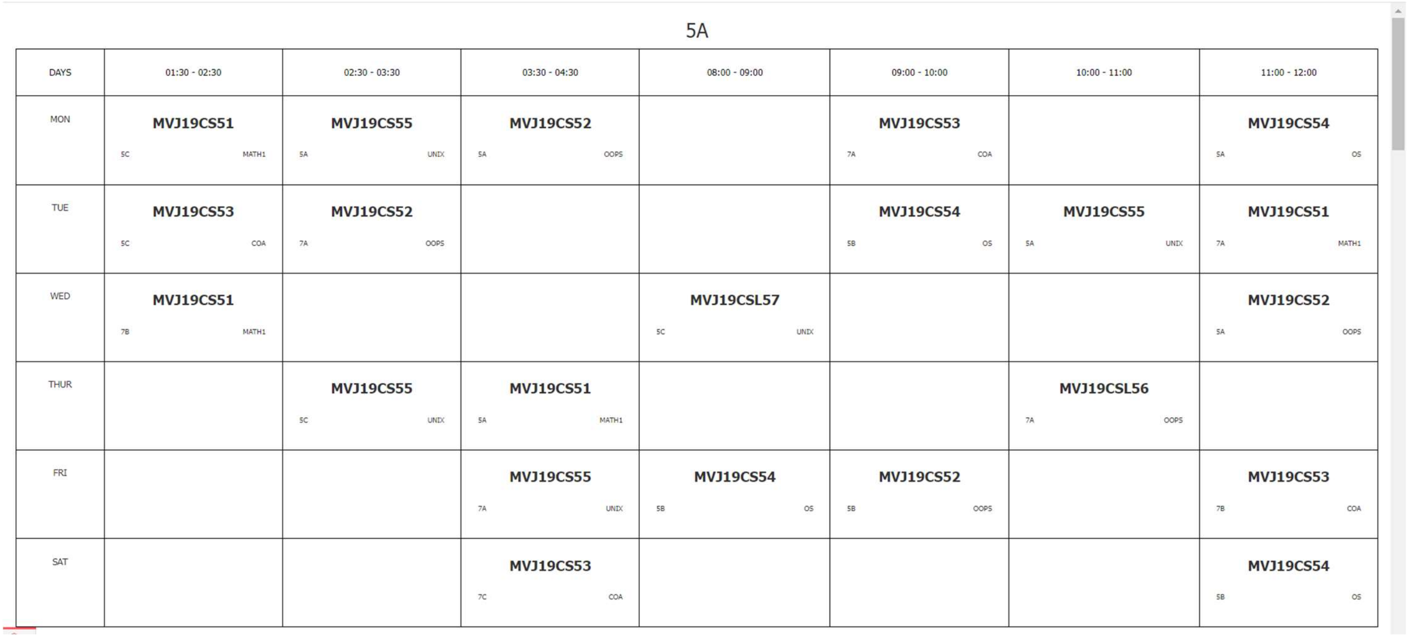
Timetable Name: 5TH SEM

Academic Period: Semester I

Select Days

- ☑ Monday
- ☑ Tuesday
- ☑ Wednesday
- ☑ Thursday
- ☑ Friday
- □ Saturday

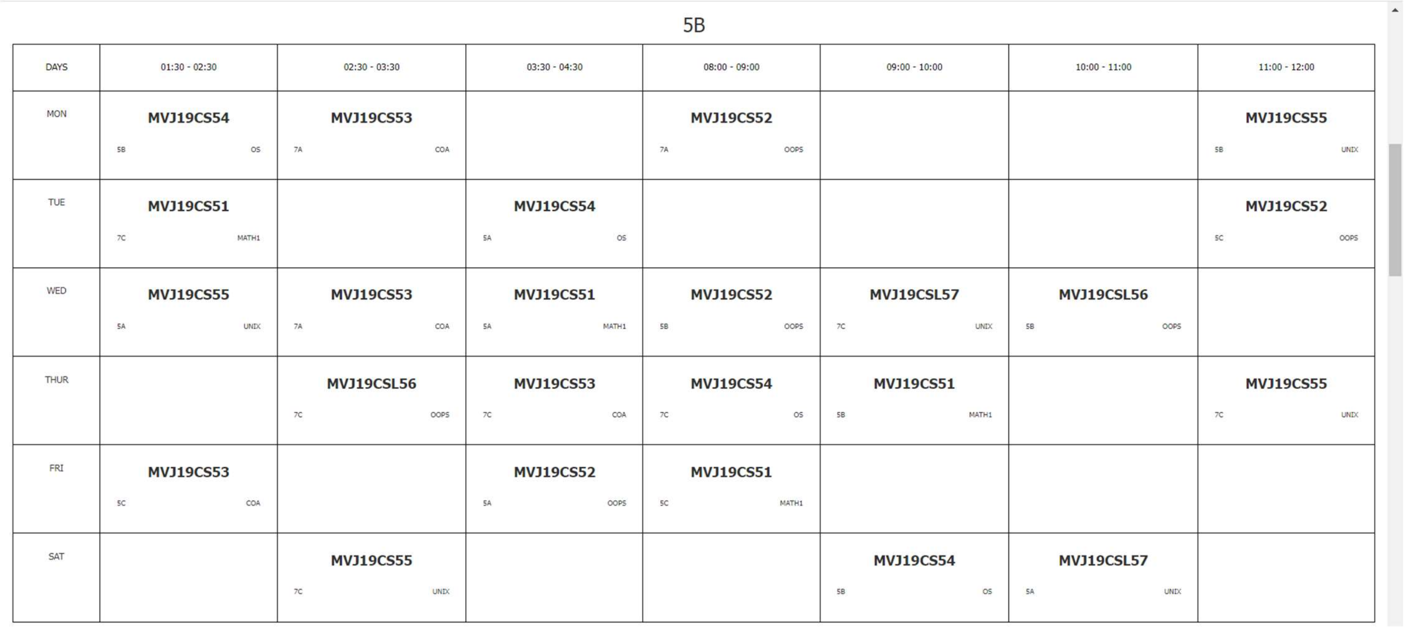
Figure 8.3 Generating New Timetable



5A

DAYS	01:30 - 02:30	02:30 - 03:30	03:30 - 04:30	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 12:00
MON	MVJ19CS51 5C MATH1	MVJ19CS55 5A UNIX	MVJ19CS52 5A OOPS		MVJ19CS53 7A COA		MVJ19CS54 5A OS
TUE	MVJ19CS53 5C COA	MVJ19CS52 7A OOPS			MVJ19CS54 5B OS	MVJ19CS55 5A UNIX	MVJ19CS51 7A MATH1
WED	MVJ19CS51 7B MATH1			MVJ19CSL57 5C UNIX			MVJ19CS52 5A OOPS
THUR		MVJ19CS55 5C UNIX	MVJ19CS51 5A MATH1			MVJ19CSL56 7A OOPS	
FRI			MVJ19CS55 7A UNIX	MVJ19CS54 5B OS	MVJ19CS52 5B OOPS		MVJ19CS53 7B COA
SAT			MVJ19CS53 7C COA				MVJ19CS54 5B OS

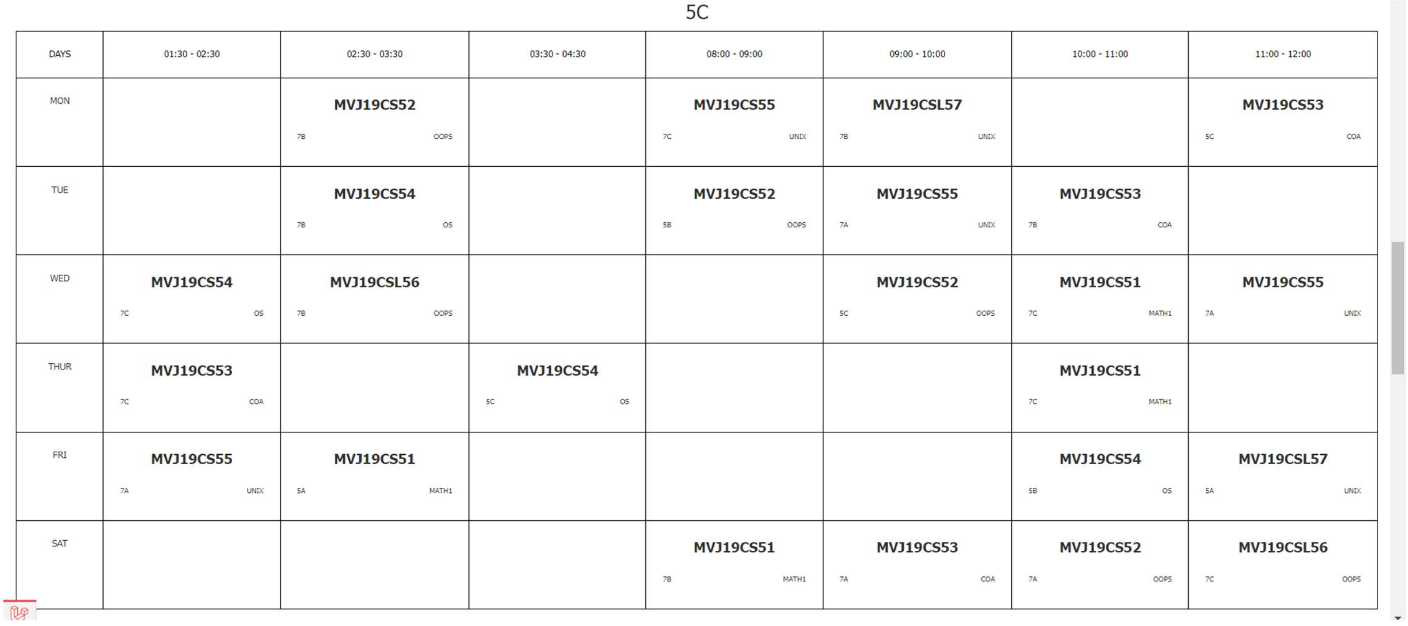
Figure 8.4 Class Timetable 5A



5B

DAYS	01:30 - 02:30	02:30 - 03:30	03:30 - 04:30	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 12:00
MON	MVJ19CS54 5B OS	MVJ19CS53 7A COA		MVJ19CS52 7A OOPS			MVJ19CS55 5B UNIX
TUE	MVJ19CS51 7C MATH1		MVJ19CS54 5A OS				MVJ19CS52 5C OOPS
WED	MVJ19CS55 5A UNIX	MVJ19CS53 7A COA	MVJ19CS51 5A MATH1	MVJ19CS52 5B OOPS	MVJ19CSL57 7C UNIX	MVJ19CSL56 5B OOPS	
THUR		MVJ19CSL56 7C OOPS	MVJ19CS53 7C COA	MVJ19CS54 7C OS	MVJ19CS51 5B MATH1		MVJ19CS55 7C UNIX
FRI	MVJ19CS53 5C COA		MVJ19CS52 5A OOPS	MVJ19CS51 5C MATH1			
SAT		MVJ19CS55 7C UNIX			MVJ19CS54 5B OS	MVJ19CSL57 5A UNIX	

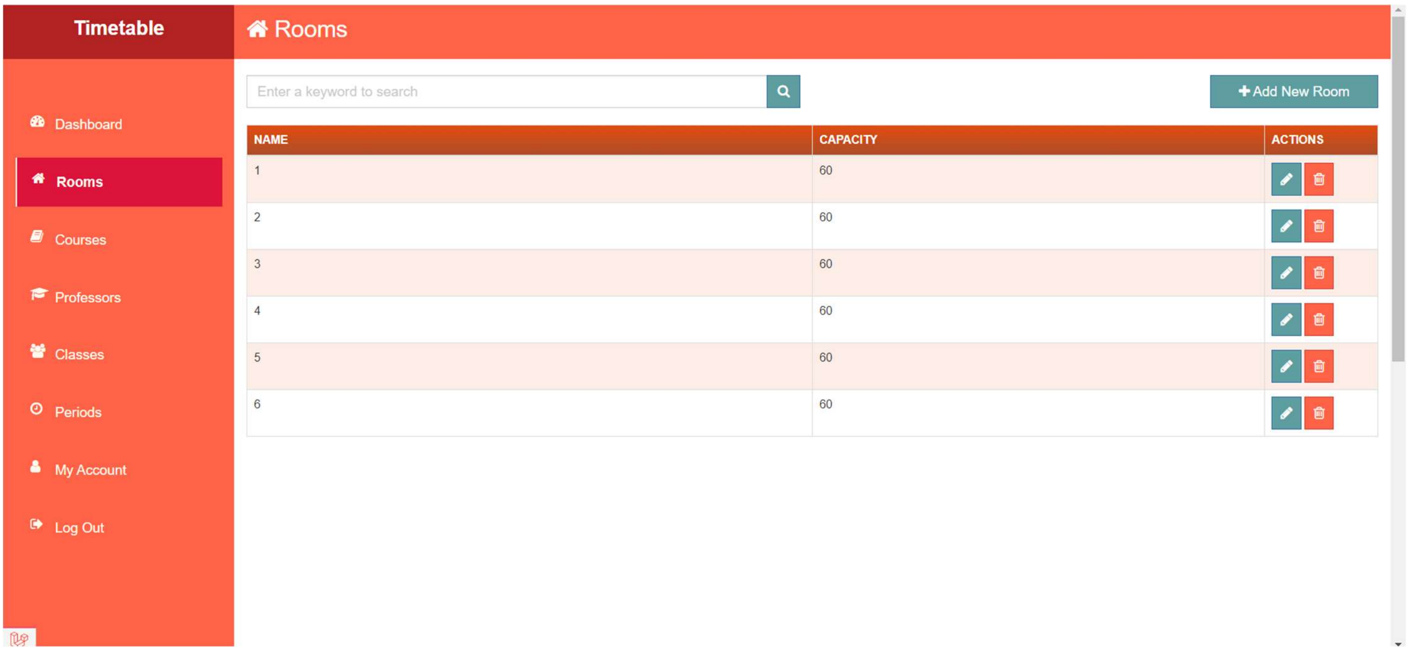
Figure 8.5 Class Timetable 5B



5C

DAYS	01:30 - 02:30	02:30 - 03:30	03:30 - 04:30	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 12:00
MON		MVJ19CS52 7B OOPS		MVJ19CS55 7C UNIX	MVJ19CSL57 7B UNIX		MVJ19CS53 5C COA
TUE		MVJ19CS54 7B OS		MVJ19CS52 5B OOPS	MVJ19CS55 7A UNIX	MVJ19CS53 7B COA	
WED	MVJ19CS54 7C OS	MVJ19CSL56 7B OOPS			MVJ19CS52 5C OOPS	MVJ19CS51 7C MATH1	MVJ19CS55 7A UNIX
THUR	MVJ19CS53 7C COA		MVJ19CS54 5C OS			MVJ19CS51 7C MATH1	
FRI	MVJ19CS55 7A UNIX	MVJ19CS51 5A MATH1				MVJ19CS54 5B OS	MVJ19CSL57 5A UNIX
SAT				MVJ19CS51 7B MATH1	MVJ19CS53 7A COA	MVJ19CS52 7A OOPS	MVJ19CSL56 7C OOPS

Figure 8.6 Class Timetable 5C



NAME	CAPACITY	ACTIONS
1	60	
2	60	
3	60	
4	60	
5	60	
6	60	

Figure 8.7 Rooms Page

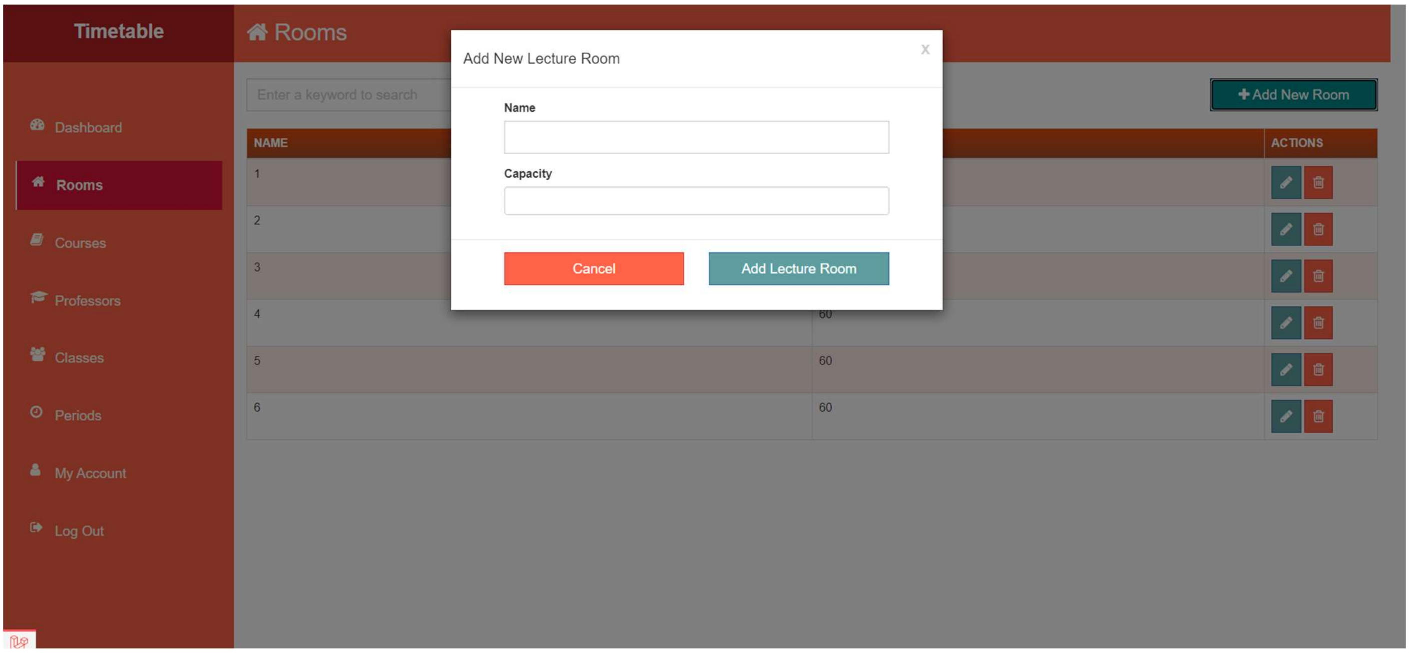
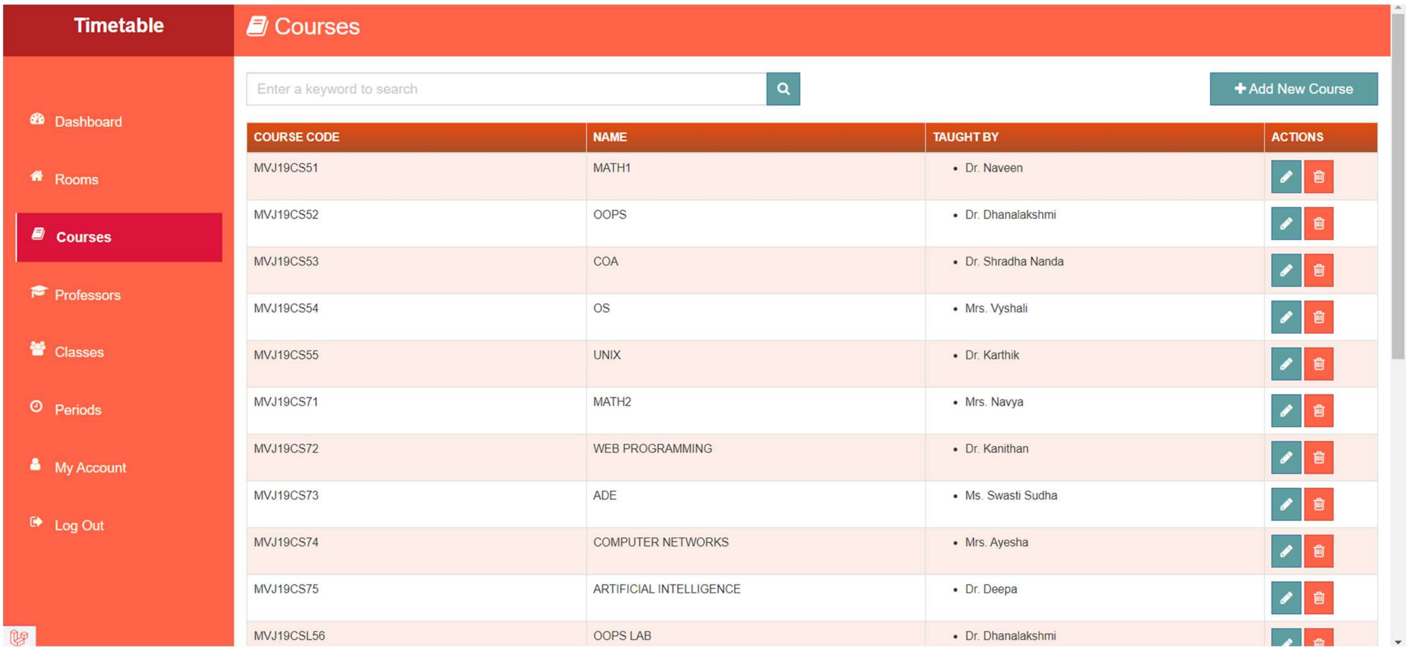
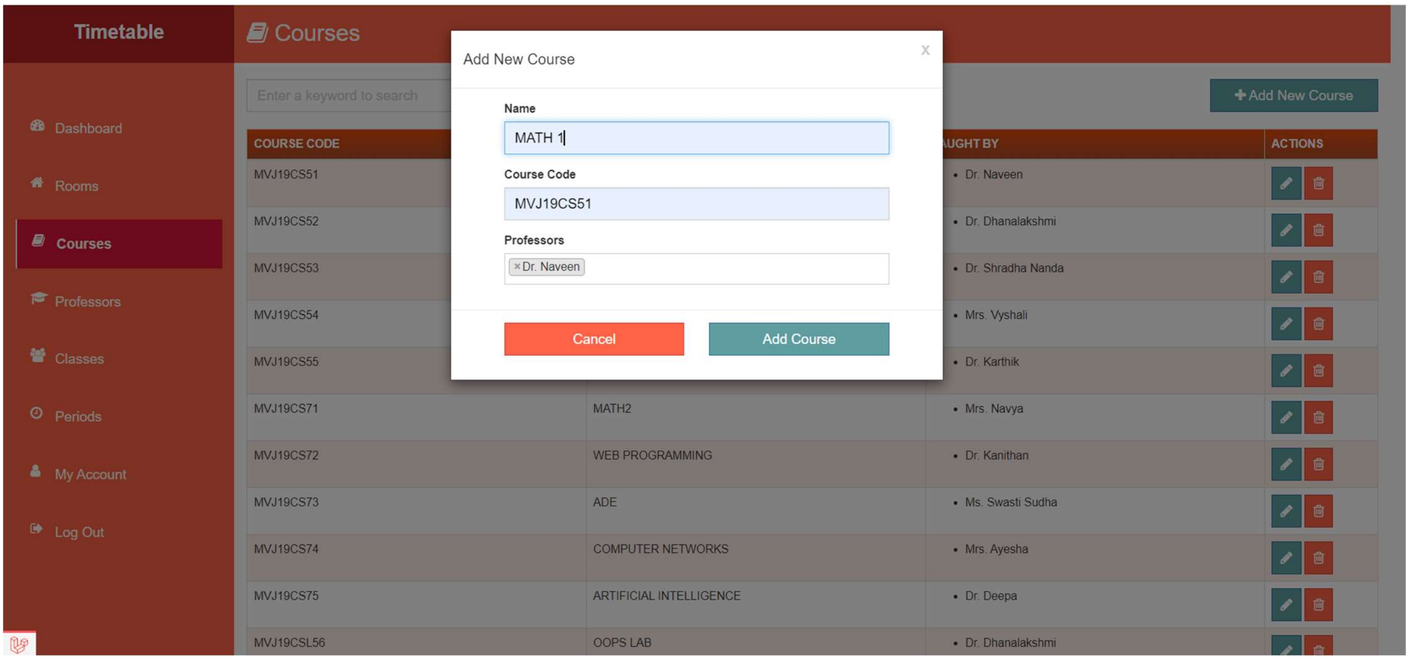


Figure 8.8 Add New Rooms



COURSE CODE	NAME	TAUGHT BY	ACTIONS
MVJ19CS51	MATH1	• Dr. Naveen	
MVJ19CS52	OOPS	• Dr. Dhanalakshmi	
MVJ19CS53	COA	• Dr. Shradha Nanda	
MVJ19CS54	OS	• Mrs. Vyshali	
MVJ19CS55	UNIX	• Dr. Karthik	
MVJ19CS71	MATH2	• Mrs. Navya	
MVJ19CS72	WEB PROGRAMMING	• Dr. Kanithan	
MVJ19CS73	ADE	• Ms. Swasti Sudha	
MVJ19CS74	COMPUTER NETWORKS	• Mrs. Ayesha	
MVJ19CS75	ARTIFICIAL INTELLIGENCE	• Dr. Deepa	
MVJ19CSL56	OOPS LAB	• Dr. Dhanalakshmi	

Figure 8.9 Courses Page



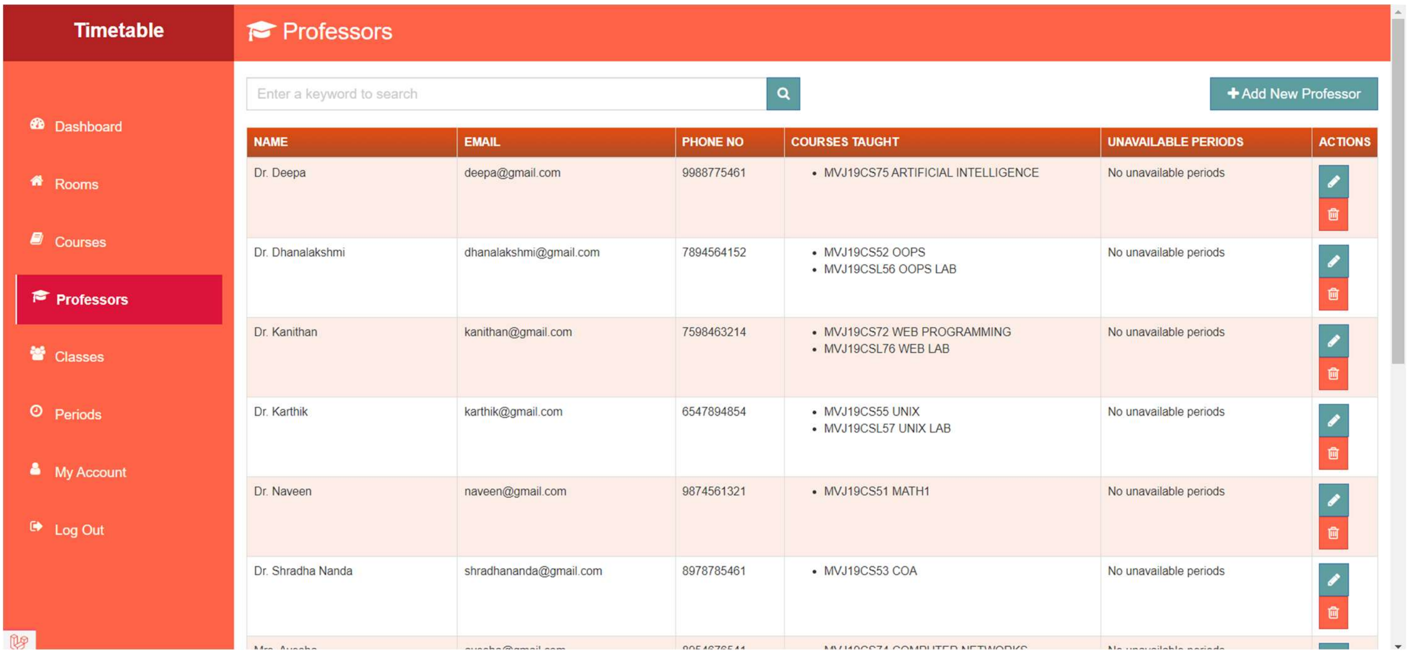
Name: MATH 1

Course Code: MVJ19CS51

Professors: ×Dr. Naveen

COURSE CODE	NAME	TAUGHT BY	ACTIONS
MVJ19CS51		• Dr. Naveen	
MVJ19CS52		• Dr. Dhanalakshmi	
MVJ19CS53		• Dr. Shradha Nanda	
MVJ19CS54		• Mrs. Vyshali	
MVJ19CS55		• Dr. Karthik	
MVJ19CS71	MATH2	• Mrs. Navya	
MVJ19CS72	WEB PROGRAMMING	• Dr. Kanithan	
MVJ19CS73	ADE	• Ms. Swasti Sudha	
MVJ19CS74	COMPUTER NETWORKS	• Mrs. Ayesha	
MVJ19CS75	ARTIFICIAL INTELLIGENCE	• Dr. Deepa	
MVJ19CSL56	OOPS LAB	• Dr. Dhanalakshmi	

Figure 8.10 Add New Course



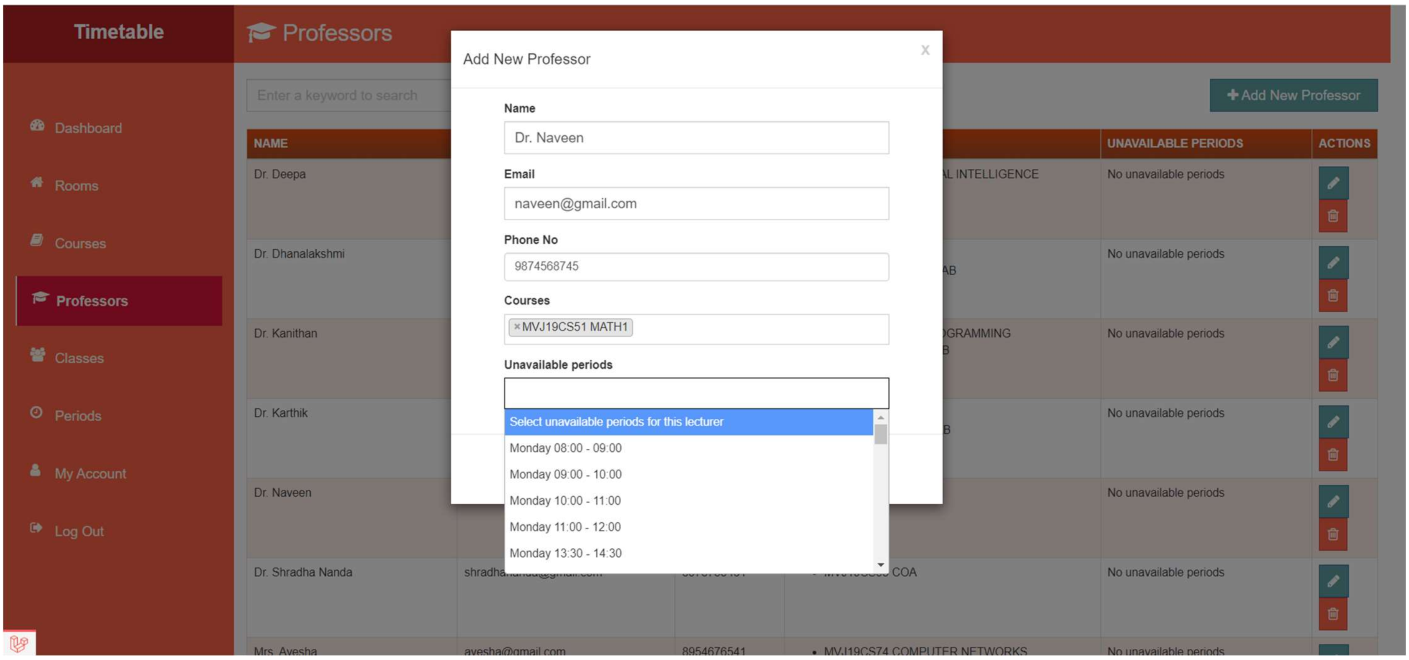
Professors

Enter a keyword to search

+ Add New Professor

NAME	EMAIL	PHONE NO	COURSES TAUGHT	UNAVAILABLE PERIODS	ACTIONS
Dr. Deepa	deepa@gmail.com	9988775461	• MVJ19CS75 ARTIFICIAL INTELLIGENCE	No unavailable periods	
Dr. Dhanalakshmi	dhanalakshmi@gmail.com	7894564152	• MVJ19CS52 OOPS • MVJ19CSL56 OOPS LAB	No unavailable periods	
Dr. Kanithan	kanithan@gmail.com	7598463214	• MVJ19CS72 WEB PROGRAMMING • MVJ19CSL76 WEB LAB	No unavailable periods	
Dr. Karthik	karthik@gmail.com	6547894854	• MVJ19CS55 UNIX • MVJ19CSL57 UNIX LAB	No unavailable periods	
Dr. Naveen	naveen@gmail.com	9874561321	• MVJ19CS51 MATH1	No unavailable periods	
Dr. Shradha Nanda	shradhananda@gmail.com	8978785461	• MVJ19CS53 COA	No unavailable periods	

Figure 8.11 Professors Page



Add New Professor

Name

Dr. Naveen

Email

naveen@gmail.com

Phone No

9874568745

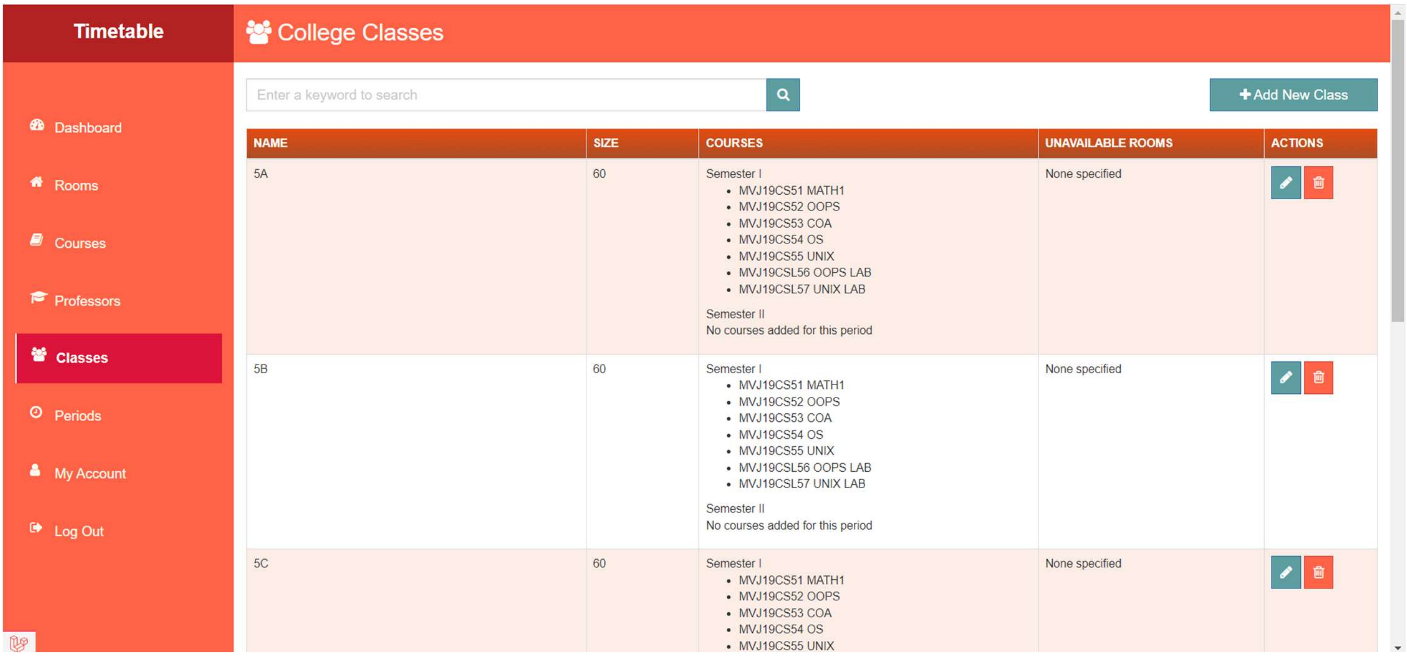
Courses

× MVJ19CS51 MATH1

Unavailable periods

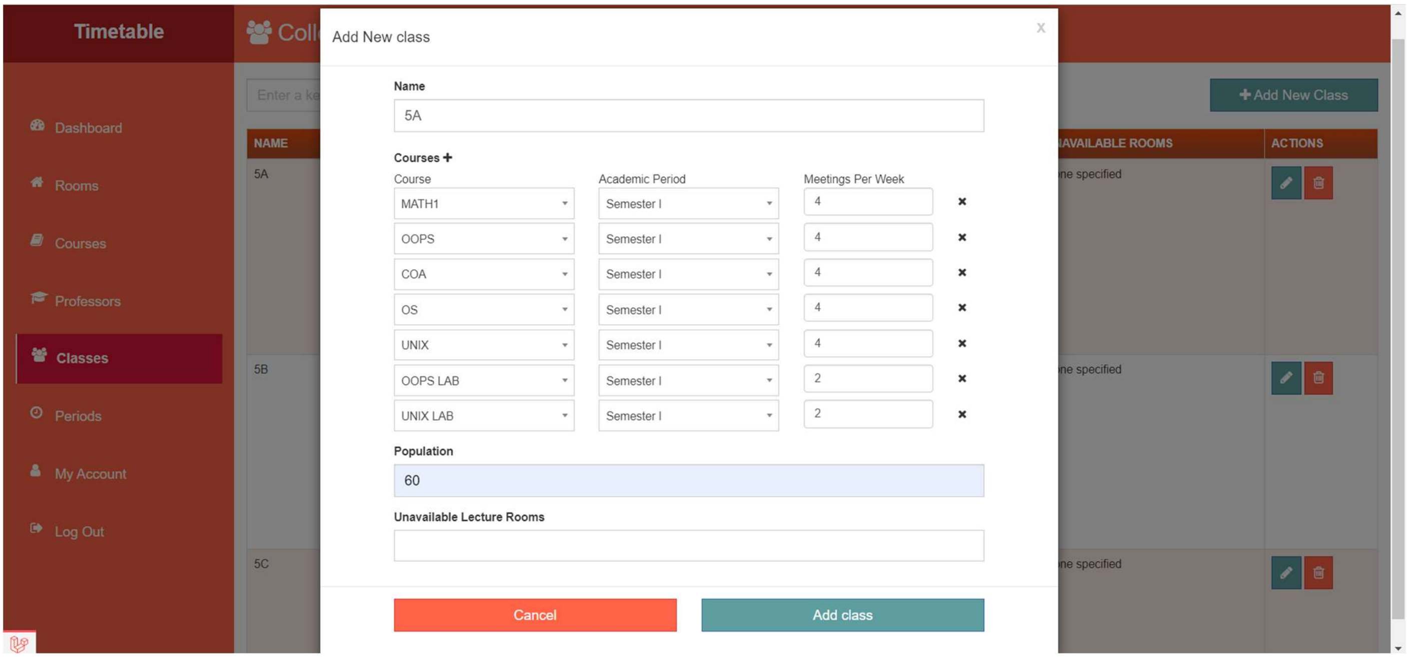
Select unavailable periods for this lecturer
Monday 08:00 - 09:00
Monday 09:00 - 10:00
Monday 10:00 - 11:00
Monday 11:00 - 12:00
Monday 13:30 - 14:30

Figure 8.12 Add New Professor



NAME	SIZE	COURSES	UNAVAILABLE ROOMS	ACTIONS
5A	60	Semester I • MVJ19CS51 MATH1 • MVJ19CS52 OOPS • MVJ19CS53 COA • MVJ19CS54 OS • MVJ19CS55 UNIX • MVJ19CSL56 OOPS LAB • MVJ19CSL57 UNIX LAB Semester II No courses added for this period	None specified	
5B	60	Semester I • MVJ19CS51 MATH1 • MVJ19CS52 OOPS • MVJ19CS53 COA • MVJ19CS54 OS • MVJ19CS55 UNIX • MVJ19CSL56 OOPS LAB • MVJ19CSL57 UNIX LAB Semester II No courses added for this period	None specified	
5C	60	Semester I • MVJ19CS51 MATH1 • MVJ19CS52 OOPS • MVJ19CS53 COA • MVJ19CS54 OS • MVJ19CS55 UNIX	None specified	

Figure 8.13 Classes Page



Add New class

Name: 5A

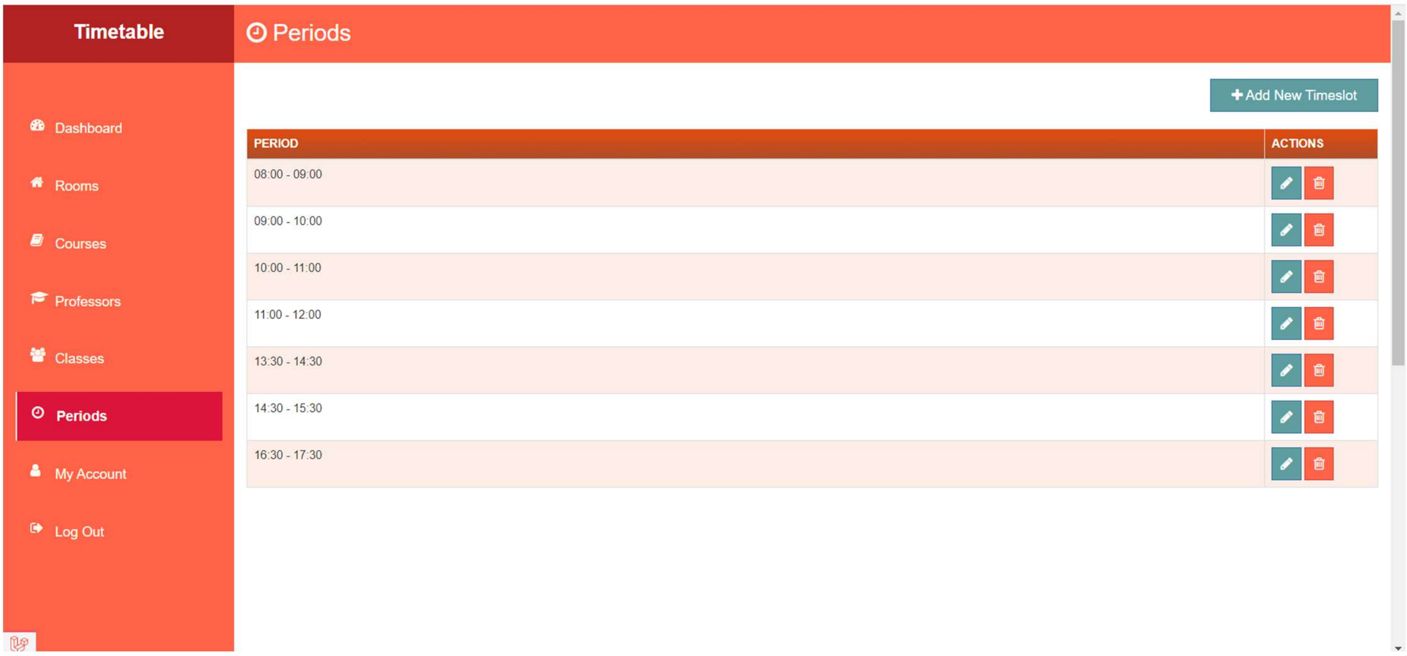
Courses +

Course	Academic Period	Meetings Per Week
MATH1	Semester I	4
OOPS	Semester I	4
COA	Semester I	4
OS	Semester I	4
UNIX	Semester I	4
OOPS LAB	Semester I	2
UNIX LAB	Semester I	2

Population: 60

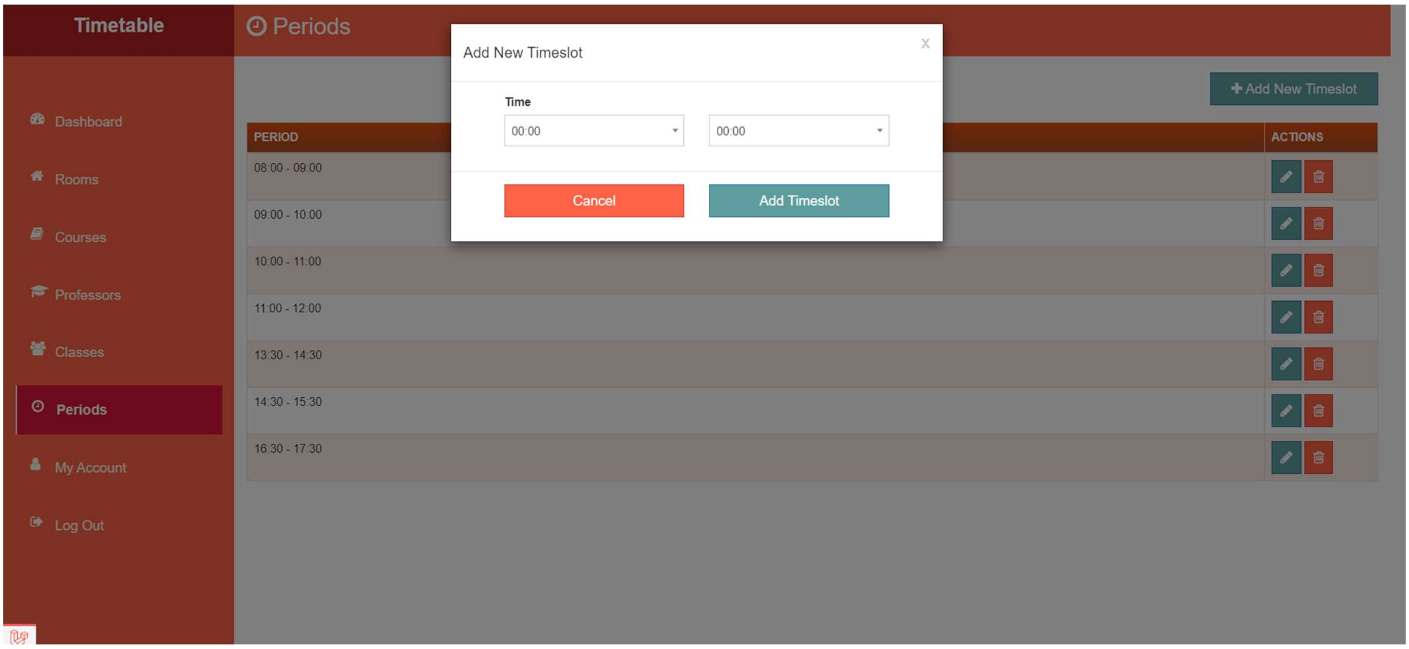
Unavailable Lecture Rooms:

Figure 8.14 Add New Classes



PERIOD	ACTIONS
08:00 - 09:00	
09:00 - 10:00	
10:00 - 11:00	
11:00 - 12:00	
13:30 - 14:30	
14:30 - 15:30	
16:30 - 17:30	

Figure 8.15 Timeslot Page

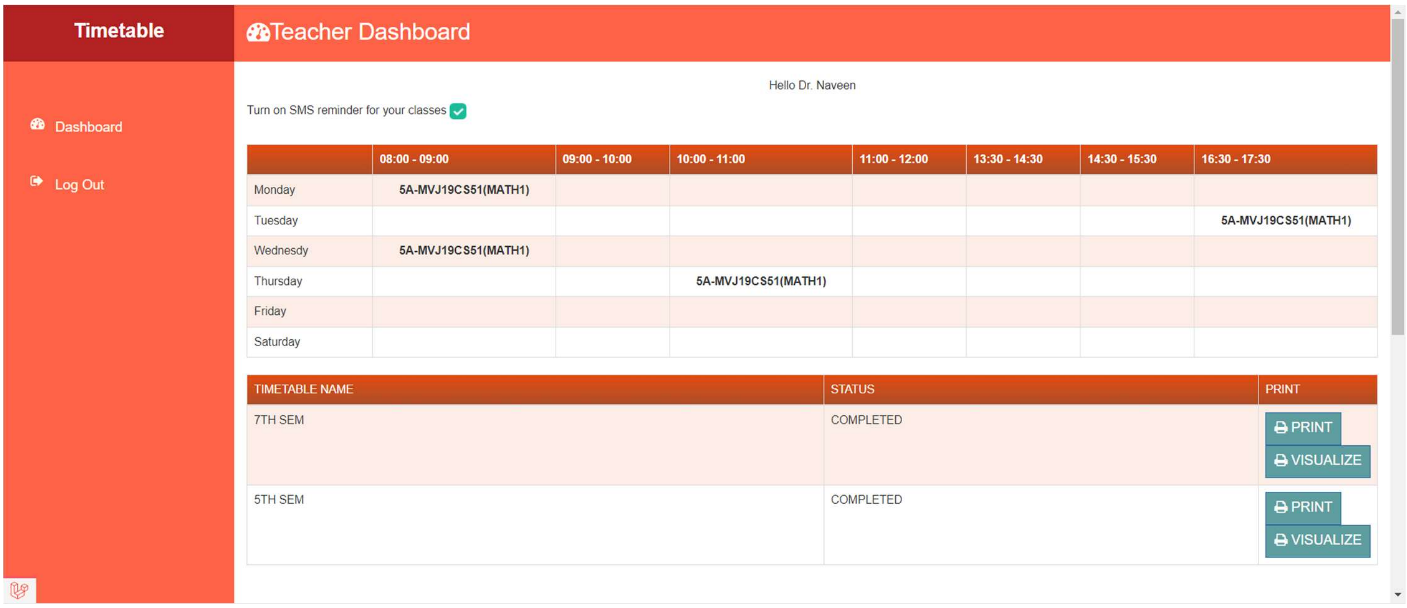


Add New Timeslot

Time: 00:00 – 00:00

PERIOD	ACTIONS
08:00 - 09:00	
09:00 - 10:00	
10:00 - 11:00	
11:00 - 12:00	
13:30 - 14:30	
14:30 - 15:30	
16:30 - 17:30	

Figure 8.16 Add New Timeslot



Hello Dr. Naveen

Turn on SMS reminder for your classes ☑

	08:00 - 09:00	09:00 - 10:00	10:00 - 11:00	11:00 - 12:00	13:30 - 14:30	14:30 - 15:30	16:30 - 17:30
Monday	5A-MVJ19CS51(MATH1)						
Tuesday							5A-MVJ19CS51(MATH1)
Wednesdy	5A-MVJ19CS51(MATH1)						
Thursday			5A-MVJ19CS51(MATH1)				
Friday							
Saturday							

TIMETABLE NAME	STATUS	PRINT
7TH SEM	COMPLETED	PRINT VISUALIZE
5TH SEM	COMPLETED	PRINT VISUALIZE

Figure 8.17 Teacher DashBoard

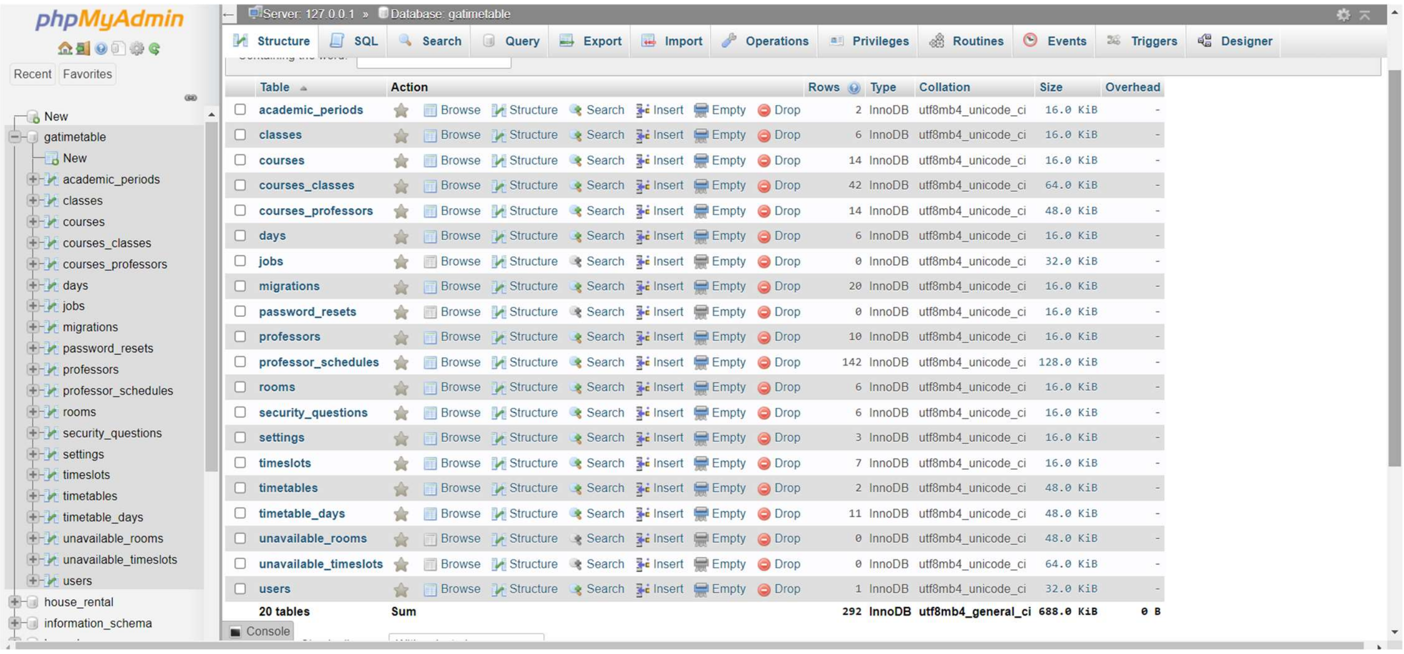


	Table	Action	Rows	Type	Collation	Size	Overhead
□	academic_periods	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	classes	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	courses	Browse Structure Search Insert Empty Drop	14	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	courses_classes	Browse Structure Search Insert Empty Drop	42	InnoDB	utf8mb4_unicode_ci	64.0 KiB	-
□	courses_professors	Browse Structure Search Insert Empty Drop	14	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
□	days	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	jobs	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
□	migrations	Browse Structure Search Insert Empty Drop	20	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	password_resets	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	professors	Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	professor_schedules	Browse Structure Search Insert Empty Drop	142	InnoDB	utf8mb4_unicode_ci	128.0 KiB	-
□	rooms	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	security_questions	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	settings	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	timeslots	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
□	timetables	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
□	timetable_days	Browse Structure Search Insert Empty Drop	11	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
□	unavailable_rooms	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	48.0 KiB	-
□	unavailable_timeslots	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	64.0 KiB	-
□	users	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci	32.0 KiB	-
	20 tables	Sum	292	InnoDB	utf8mb4_general_ci	688.0 KiB	0 B

Figure 8.18 Table Names

FUTURE SCOPE

The Automatic Timetable Generator is driven portal for educational organization and is a web-based application which will be helpful for creating timetables. This project will be a great helpful for the institutions because, it is a great difficult task that to manage many Faculty's and allocating subjects for them at a time manually and this project will help to manage it properly. This manage timetable for faculty with considering maximum and minimum workload and can be managed easily.

CONCLUSION

Automatic Timetable Generator is a web-based application for generating timetable automatically. It is a great difficult task that to manage many Faculty's and allocating subjects for them at a time manually. So proposed system will help to overcome this disadvantage. Thus, we can generate timetable for any number of courses and multiple semesters. This system will help to create dynamic pages so that for implementing such a system we can make use of the different tools are widely applicable and free to use also.

REFERENCES

[1] Utkarsh Kumar, Yash Kamboj, Vaibhav Kumar, Rajesh Singh "Timetable Generator Using Heuristic Technique", International Journal for Research in Applied Science & Engineering Technology (IJRASET),2022.

[2] Mr. Vadipina Amarnadh, B. Hemanthi, G. Sravanthi, S. SaiSanketh "Automatic Timetable Generator", International Journal of Research (IJR), 2020.

[3] Shraddha Ambhore1, Pooja Walke2, Rohit Ghundgrudkar3, Akshay Alone4, Anushree Khedkar "Timetable Generator Using N-Queen Algorithm", International Journal of Research in Engineering, Science and Management (IJRESM), 2020.

[4] V. Abhinaya, K. Sahithi, K. Akaanksha, "Online Application of Automatic Time-Table Generator", International Research Journal of Engineering and Technology (IRJET), 2019.

[5] Akshay Puttaswamy, H M Arshad Ali Khan, Chandan S.V, Parkavi.A "A Study On Automatic Timetable Generator", International Journal Of Science And Innovative Engineering And Technology (IJSIET), 2018.

[6] Sundresan Al Perumal, Mujahid Tabassum, Norita Md Norwawi, Ganthan Al Narayana Samy, "Development of an Efficient Timetable System", 8th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), 2018.

[7] Tanzila Islam, Zunayed Shahriar, Mohammad Anower Perves, Monirul Hasan "Timetable Generator Using Tabu Search", Journal of Computer and Communications, 2016.

[8] Mayuri R. Bagul, Sunil C. Chaudhari, Sunita N. Nagare, Pushkar R. Patil, K.S. Kumavat "A Novel Approach for Automatic Timetable Generation", International Journal of Computer Applications, 2015.

[9] Andeep Kumar, Kawaljeet Singh, Neeraj Sharma, "Timetable generator using Particle Swarm Optimization", International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC), 2013.