Automated Timetable Scheduler for IIIT Dharwad

User Manual

Team Byte_Me

Indian Institute of Information Technology Dharwad

Version Information

Version: 1.0
Release Date: November 2025
Document Type: End User Manual
Classification: Public

November 18, 2025

Contents

Chapter 1

Overview

1.1 Introduction

The **Automated Timetable Scheduler** is a comprehensive software system designed to generate conflict-free academic timetables for educational institutions. Developed by Team Byte_Me for IIIT Dharwad, this system automates the complex process of scheduling classes, labs, and exams while respecting multiple constraints including faculty availability, room capacity, and student requirements.

1.2 Key Features

- **Automated Class Scheduling**: Generates timetables for multiple semesters (1, 3, 5, 7) and departments (CSE, DSAI, ECE)
- **Pre/Post Mid-Semester Support**: Handles half-semester courses with intelligent slot allocation
- **Combined Class Management**: Schedules large combined classes in 240-seater auditoriums
- **Elective & Basket Course Handling**: Groups electives cross-departmentally and baskets by department
- **Faculty Conflict Resolution**: Ensures 30-minute breaks between classes for faculty
- **Room Optimization**: Allocates appropriate classrooms and labs based on capacity requirements
- **Exam Scheduling**: Generates exam schedules with seating arrangements
- **Web Interface**: Role-based access for Students, Faculty, and Administrators
- **Excel Export**: Downloads department and faculty timetables as formatted Excel files
- **Validation Engine**: Comprehensive validation checks for conflicts and LTPSC fulfillment

1.3 System Architecture

The system follows a modular architecture with clear separation of concerns:

System Components:

1. **Data Layer** (`data_loader.py`): CSV parsing, course bundling, data normalization
2. **Core Models** (`models.py`): Course, Classroom, Section, Timetable objects
3. **Scheduling Engine** (`scheduler.py`): Multi-phase scheduling algorithm
4. **Validation Module** (`validators.py`): Conflict detection and LTPSC verification
5. **Export Module** (`excel_exporter.py`): Excel generation with formatting
6. **Web Interface** (`web_app.py`): Flask-based role-based portal
7. **Exam Scheduler** (`exam_scheduler_main.py`): Separate exam schedule generator

Figure 1.1: System Architecture Components

1.4 Target Users

1. **Students**: View their section-wise weekly class schedules
2. **Faculty**: Access their teaching schedules across all sections
3. **Administrators**: Generate timetables, manage data, download reports

1.5 Technical Specifications

Attribute	Details
Programming Language	Python 3.8+
Web Framework	Flask
Data Format	CSV (input), Excel (output)
Dependencies	pandas, openpyxl, Flask
Slot Duration	10 minutes
Daily Time Range	09:00 - 18:00 (54 slots)
Session Durations	Lecture: 90 min, Tutorial: 60 min, Practical: 120 min

Table 1.1: Technical Specifications

Chapter 2

Installation

2.1 System Requirements

2.1.1 Hardware Requirements

- **Processor**: Intel Core i3 or equivalent (i5/i7 recommended)
- **RAM**: Minimum 4 GB (8 GB recommended)
- **Storage**: 100 MB free disk space
- **Display**: 1366x768 or higher resolution

2.1.2 Software Requirements

- **Operating System**: Windows 10/11, macOS 10.14+, or Linux (Ubuntu 18.04+)
- **Python**: Version 3.8 or higher
- **Web Browser**: Chrome 90+, Firefox 88+, Safari 14+, or Edge 90+
- **Internet Connection**: Required for initial setup (pip package installation)

2.2 Installation Steps

2.2.1 Step 1: Install Python

Windows:

1. Download Python from https://www.python.org/downloads/
2. Run the installer and **check "Add Python to PATH"**
3. Verify installation: Open Command Prompt and type:

```
python --version
```

macOS/Linux:

```
# macOS (using Homebrew)
brew install python3

# Linux (Ubuntu/Debian)
sudo apt update
sudo apt install python3 python3-pip

# Verify installation
python3 --version
```

2.2.2 Step 2: Download the Project

1. Download the project ZIP file or clone from GitHub:

```
git clone https://github.com/shriyanssahoo/Byte_Me
```

2.2.3 Step 3: Install Dependencies

```
# Install required Python packages
pip install -r requirements.txt
```

The `requirements.txt` contains:

```
pandas
openpyxl
flask
```

2.2.4 Step 4: Verify Project Structure

Ensure your directory structure looks like this:

```
Byte_Me/
          main.py                    # Command-line timetable generator
          web_app.py                # Web interface
          requirements.txt
          data/
                course.csv
                classroom_data.csv
                students.csv
                exam_config.csv
                exam_rooms.csv
          src/
                __init__.py
                data_loader.py
                models.py
                scheduler.py
                validators.py
                excel_exporter.py
                exam_scheduler_main.py
                utils.py
          output/               # Generated files appear here
```

2.3 First-Time Setup

2.3.1 Preparing Data Files

Before running the system, ensure all CSV files in the `data/` folder are properly formatted:

1. **course.csv**: Contains course definitions (see Section 4.1)
2. **classroom_data.csv**: Room definitions with capacities
3. **students.csv**: Student enrollment data
4. **exam_config.csv**: Exam timing configuration
5. **exam_rooms.csv**: Exam room seating layouts

Important Note

Do not modify CSV headers. The system expects specific column names. Only modify the data rows.

2.4 Installation Verification

Run this test to verify everything is set up correctly:

```
# Test command-line generator
python main.py

# Test web interface
python web_app.py
```

If successful, you should see:

- Command-line: Messages showing scheduling progress
- Web interface: `Running on http://localhost:5000`

Chapter 3

Usage Scenarios

3.1 Command-Line Interface (CLI) Usage

3.1.1 Generating Timetables via CLI

The CLI mode is ideal for batch generation and automation:

```
# Run the scheduler
python main.py
```

Output:

- `output/Department_Timetables.xlsx`: All section timetables
- `output/Faculty_Timetables.xlsx`: All faculty schedules

Expected Console Output:

```
Starting Automated Timetable Scheduler...
Loading classrooms from data/classroom_data.csv...
Successfully loaded 29 classrooms.
Loading and processing courses from data/course.csv...
Bundling electives and baskets...
Found 9 cross-departmental ELECTIVE bundles (Type 1).
Found 16 department-specific BASKET bundles (Type 2).

--- INITIALIZING PRE MIDSEM SCHEDULING RUN ---
  Running Phase 4: Combined Classes
  Running Phase 3: Elective/Basket Slots
  Running Phase 5/6: Core Courses
  Running Phase 8: Assigning Electives to Rooms/Faculty
--- PRE RUN COMPLETE (Sem 1) ---

Validating PRE-Midsem master schedule...
Validation PASSED: All timetables are conflict-free

Successfully saved department timetables to output/
   Department_Timetables.xlsx
Successfully saved faculty timetables to output/Faculty_Timetables.xlsx

--- Timetable Generation Complete. ---
```

3.1.2 Generating Exam Schedules

```
# Generate exam schedule and seating plans
python src/exam_scheduler_main.py
```

Output:

- `output/exams/Exam_Schedule.csv`: Complete exam calendar
- `output/exams/Exam_[CourseCode][Date][Slot].xlsx`: Individual exam seating plans

3.2 Web Interface Usage

3.2.1 Starting the Web Server

```
python web_app.py
```

Open your browser and navigate to: `http://localhost:5000`

3.2.2 Login Page

Login Page Screenshot

URL: `http://localhost:5000/`
Features:

- Three role-based access options:
  - **Student** - View class schedules
  - **Faculty** - View teaching schedules
  - **Admin** - Manage and generate timetables
- Clean, modern interface with gradient backgrounds
- Click on any card to access the respective portal

Figure 3.1: Login Page - Role Selection

3.2.3 Student Portal

Step 1: Select Section

Student Section Selection

URL: `http://localhost:5000/student`
Features:

- Search box for filtering sections (e.g., "CSE-Sem1-PRE-A")
- Scrollable list of all available sections
- Sections organized by: Department-Semester-Period-Section

Example Sections:

- CSE-Sem1-PRE-A
- CSE-Sem1-PRE-B
- DSAI-Sem3-POST
- ECE-Sem5-PRE

Figure 3.2: Student Portal - Section Selection Screen

Step 2: View Timetable

Student Timetable View

URL: `http://localhost:5000/student/timetable?id=CSE-Sem1-PRE-A`
Timetable Layout:

- **Rows**: Days (Monday - Friday)
- **Columns**: Time slots (09:00 - 18:00 in 10-minute intervals)
- **Color Coding**: Each course has a unique color for easy identification

Cell Information:

- Course Name
- Session Type (Lecture/Tutorial/Practical)
- Instructor Name(s)
- Room Number(s)

Special Indicators:

- LUNCH: Gray background (12:30-13:00 for Sem 1 & 7)
- BREAK: Light gray (10-minute breaks between classes)
- Merged Cells: Multi-slot sessions span multiple columns

Figure 3.3: Student Timetable View

3.2.4 Faculty Portal

Step 1: Select Faculty Name

Faculty Name Selection

URL: `http://localhost:5000/faculty`
Features:

- Search box for filtering faculty names
- Alphabetically sorted list of all instructors
- Includes all faculty who have been assigned courses

Example Faculty:

- Abdul Wahid
- Animesh Roy
- Prakash Pawar
- Sunil P V

Figure 3.4: Faculty Portal - Name Selection

Step 2: View Teaching Schedule

Faculty Timetable View

URL: `http://localhost:5000/faculty/timetable?name=Sunil P V`
Timetable Layout:

- Same day/time grid as student view
- Shows all classes the faculty member teaches across all sections

Cell Information:

- Course Name
- Session Type
- Section ID (e.g., CSE-Sem1-PRE-A)
- Room Number

Use Cases:

- Identify teaching load distribution
- Check for adequate breaks (30 min between classes)
- Plan office hours in free slots

Figure 3.5: Faculty Teaching Schedule

3.2.5 Admin Dashboard

Dashboard Overview

Admin Dashboard

URL: `http://localhost:5000/admin`
Action Buttons:

1. **Re-Generate Timetable**: Runs the scheduling algorithm
2. **Download Class Timetables**: Exports all section timetables
3. **Download Faculty Timetables**: Exports all faculty schedules
4. ← **Back to Home**: Returns to login page

Dashboard Panels:

1. **System Statistics**: Shows counts of sections, faculty, rooms
2. **Courses Loaded**: Table of all courses with codes, names, departments
3. **Classrooms Loaded**: Table of rooms with IDs, types, capacities

Figure 3.6: Admin Dashboard Layout

Generating Timetables

1. Click ” **Re-Generate Timetable**”
2. Confirm the action (this may take 30-60 seconds)
3. Wait for the success message
4. Page will automatically reload with updated statistics

Warning

Re-generating will **overwrite** all existing timetables. Ensure you have downloaded any needed reports before re-generating.

Downloading Reports

Download Class Timetables:

- Click ” **Download Class Timetables**”
- File: `Department_Timetables.xlsx`
- Contains: One sheet per section (e.g., CSE-Sem1-PRE-A, DSAI-Sem3-POST)

Download Faculty Timetables:

- Click **" Download Faculty Timetables"**
- File: `Faculty_Timetables.xlsx`
- Contains: One sheet per faculty member

3.3 Typical Workflow

3.3.1 Semester Start Workflow

1. **Admin**: Update `data/course.csv` with new semester courses
2. **Admin**: Update `data/students.csv` with enrollment data
3. **Admin**: Access `http://localhost:5000/admin`
4. **Admin**: Click "Re-Generate Timetable"
5. **Admin**: Download and distribute Excel files to departments
6. **Students**: Access `http://localhost:5000/student` to view schedules
7. **Faculty**: Access `http://localhost:5000/faculty` to view teaching loads

3.3.2 Mid-Semester Adjustments

1. Modify affected course entries in `course.csv`
2. Re-generate timetable from Admin dashboard
3. Download updated timetables
4. Notify affected students and faculty

Chapter 4

Configuration Files

4.1 Course Configuration (course.csv)

4.1.1 File Location

`data/course.csv`

4.1.2 Column Descriptions

Column	Type	Description
Course Code	Text	Unique identifier (e.g., CS161, MA261)
Course Name	Text	Full course name
Semester	Number	1, 3, 5, or 7
Department	Text	CSE, DSAI, or ECE
LTPSC	Text	Format: L-T-P-S-C (e.g., 3-1-0-0-2)
Credits	Number	Course credits
Instructor	Text	Comma-separated names (e.g., "Sunil P V, Sunil C K")
Registered Students	Number	Total enrolled students
Elective (Yes/No)	Text	"Yes" or "No"
Half Semester (Yes/No)	Text	"Yes" or "No"
Combined class	Text	"yes" (lowercase) or blank
Pre /Post	Text	pre, post, full, elective, basket, pre/post
Basket Code	Text	A, B, C, D (for electives/baskets)

4.1.3 LTPSC Format Explained

The LTPSC string defines the teaching structure:

- **L (Lecture)**: Number of lecture hours per week
  - L=3 → 2 lecture sessions

  - L=2 → 2 lecture sessions
  - L=1 → 1 tutorial session (special rule)

- **T (Tutorial)**: Number of tutorial hours per week
- **P (Practical)**: Number of practical hours per week (must be even)
- **S (Self-study)**: Not used in scheduling
- **C (Credits)**: Not used in scheduling

Example: `3-1-2-0-4`

- 2 × 90-minute lectures
- 1 × 60-minute tutorial
- 1 × 120-minute practical (P=2 means 1 session)

4.1.4 Pre/Post Preferences

Value	Behavior
pre	Course scheduled only in PRE-midsem period
post	Course scheduled only in POST-midsem period
full	Course scheduled in both PRE and POST (full semester)
elective	Placed in PRE; if no room/faculty, overflow to POST
basket	Placed in both PRE and POST (basket course)
pre/post	Split course: Section A in PRE, Section B in POST

Table 4.2: Pre/Post Preference Values

4.1.5 Sample Course Entries

```
Course Code,Course Name,Semester,Department,LTPSC,Credits,Instructor,Registered Students,Elective (Yes/No),Half Semester (Yes/No),Combined class,Pre /Post,Basket Code
CS161,Problem Solving with Python,1,CSE,3-0-2-0-4,4,"Sunil P V, Sunil C K",200,No,No,no,full,
MA161,Statistics,1,CSE,3-1-0-0-2,2,Ramesh Athe,200,No,Yes,yes,Pre,
CS463,Parallel computing,5,CSE,3-1-0-0-4,4,Pramod,0,Yes,No,no,basket,A
```

4.2 Classroom Configuration (classroom_data.csv)

4.2.1 File Location

`data/classroom_data.csv`

4.2.2 Column Descriptions

Column	Description
Room Number	Unique ID (e.g., C101, L105)
Type	"Classroom" or "Lab"
Capacity	Number of students the room can hold
Facilities	Comma-separated (e.g., "Projector, Audio System")

Table 4.3: Classroom Data Columns

4.2.3 Room Naming Convention

- **C###**: Classroom (e.g., C101, C202)
- **L###**: Lab (e.g., L105, L206)
- **First Digit**: Floor number (C202 → 2nd floor)

4.2.4 Sample Classroom Entries

```
Room Number,Type,Capacity,Facilities
C004,Classroom,240,"Projector, Audio System"
C101,Classroom,96,Projector
L105,Lab,40,"Hardware Kits, Computers"
```

4.3 Student Data (students.csv)

4.3.1 File Location

`data/students.csv`

4.3.2 Column Descriptions

Column	Description
roll_number	Student ID (e.g., 24BCS001)
name	Full name
branch	CSE, DSAI, or ECE
section	A or B (for CSE); blank for others
semester	Current semester (1, 3, 5, 7)

Table 4.4: Student Data Columns

4.3.3 Usage

This file is primarily used by the exam scheduler to:

4.3.4 Sample Student Entries

```
roll_number,name,branch,section,semester
24BCS001,AAKASH BABASAHEB PATHRIKAR,CSE,A,3
24BCS086,NIKHIL KUMAR SINHA,CSE,B,3
24BDS001,AALEKH RAGHUVANSHI,DSAI,A,3
24BEC001,ABHINAV KUMAR,ECE,A,3
```

4.4 Exam Configuration (exam_config.csv)

4.4.1 File Location

`data/exam_config.csv`

4.4.2 Column Descriptions

Parameter	Value	Description
exam_start_date	2025-12-01	First day of exams (YYYY-MM-DD)
morning_slot_start	10:00	Morning exam start time
morning_slot_2hr_end	12:00	End time for 2-hour morning exams
morning_slot_3hr_end	13:00	End time for 3-hour morning exams
afternoon_slot_start	14:00	Afternoon exam start time
afternoon_slot_2hr_end	16:00	End time for 2-hour afternoon exams
afternoon_slot_3hr_end	17:00	End time for 3-hour afternoon exams

Table 4.5: Exam Configuration Parameters

4.4.3 Sample exam_config.csv

```
parameter,value
exam_start_date,2025-12-01
morning_slot_start,10:00
morning_slot_2hr_end,12:00
morning_slot_3hr_end,13:00
afternoon_slot_start,14:00
afternoon_slot_2hr_end,16:00
afternoon_slot_3hr_end,17:00
```

4.5 Exam Rooms (exam_rooms.csv)

4.5.1 File Location

`data/exam_rooms.csv`

4.5.2 Column Descriptions

Column	Description
room_id	Room identifier (e.g., C202)
capacity	Total seats in the room
rows	Number of rows in seating layout
columns	Number of columns (students per row)

Table 4.6: Exam Room Configuration

4.5.3 Seating Strategy

The exam scheduler uses a two-pass column filling strategy:

1. **Pass 1**: Fill odd columns (C1, C3, C5) with largest department
2. **Pass 2**: Fill even columns (C2, C4, C6) with other departments

This maximizes spacing between students from the same department.

4.5.4 Sample exam_rooms.csv

```
room_id,capacity,rows,columns
C202,48,8,3
C203,48,8,3
C402,48,8,3
```

Interpretation: C202 has 48 seats arranged in 8 rows × 3 columns (24 seats) but actual capacity is doubled due to the two-pass strategy utilizing partial rows.

Chapter 5

Advanced Topics

5.1 Scheduling Algorithm

5.1.1 Multi-Phase Scheduling Approach

The scheduler operates in 8 distinct phases to handle different course types and constraints:

1. **Phase 1-2**: Data Loading and Validation
2. **Phase 3**: Elective/Basket Slot Reservation
3. **Phase 4**: Combined Class Scheduling (240-seater)
4. **Phase 5-6**: Core Course Scheduling
5. **Phase 7**: Lab Scheduling
6. **Phase 8**: Elective Assignment to Reserved Slots

5.1.2 Constraint Satisfaction

The algorithm respects multiple hard constraints:

Hard Constraints

- **Faculty Break Rule**: 30-minute gap between consecutive classes
- **Daily Limit**: Maximum 1 lecture/tutorial and 1 lab per course per day
- **Room Capacity**: Students must fit in assigned room
- **No Overlaps**: Students, faculty, and rooms cannot be double-booked
- **Lunch Slots**: Fixed lunch times by semester (cannot be scheduled)
- **Session Duration**: Lecture=90min, Tutorial=60min, Practical=120min
- **LTPSC Fulfillment**: All required sessions must be scheduled

5.1.3 Elective Bundling

The system intelligently groups electives into two types:

Type 1 - Cross-Departmental Electives:

- Grouped by: (Semester, Basket Code)
- Example: All Semester 1, Basket A electives
- Scheduled together across all departments

Type 2 - Department-Specific Baskets:

- Grouped by: (Semester, Department, Basket Code)
- Example: CSE Semester 5, Basket A courses
- Scheduled only for that department's students

5.2 Validation System

5.2.1 Validation Checks

The system runs comprehensive post-scheduling validation:

1. **Student Conflict Check**: Detects double-booked slots in section timetables
2. **Faculty Conflict Check**: Verifies 30-min breaks and no overlaps
3. **Daily Limit Check**: Ensures max 1 class + 1 lab per course per day
4. **Break Verification**: Confirms 10-min breaks exist after each class
5. **LTPSC Fulfillment**: Matches scheduled sessions against course requirements

5.2.2 Validation Output

After scheduling, the console displays validation results:

```
--- RUNNING POST-SCHEDULING VALIDATION ---
Validation PASSED: All timetables are conflict-free and LTPSC is
   fulfilled.
```

Or, if issues are found:

```
Validation FAILED:
  Found 3 faculty conflicts.
    - Faculty Break Violation: Dr. Smith at Monday 10:50
  Found 1 daily limit violations.
    - Daily Limit Violation: CSE-Sem1-PRE-A has 2 'CS161_CLASS'
   sessions on Tuesday
```

5.3 Semester 7 Special Handling

Semester 7 follows a unique rule: **PRE and POST periods are identical**.

5.3.1 Implementation

1. Schedule Semester 7 courses in PRE-midsem
2. Deep copy all PRE timetables to POST
3. Clone all faculty and room bookings to POST master schedules

This ensures students have the same schedule all semester, accommodating final year projects and placements.

5.4 Room Allocation Strategy

5.4.1 Classroom Selection

The system uses a "best-fit" algorithm:

1. Filter rooms by type (Classroom vs Lab)
2. Filter by minimum capacity requirement
3. Sort by capacity (ascending)
4. Select the smallest room that fits

Rationale: Saves larger rooms for courses with more students.

5.4.2 Combined Classes

Courses marked with `Combined class = yes` receive special treatment:

- Automatically assigned to C004 (240-seat auditorium)
- Scheduled for all sections simultaneously
- Requires common free slot across all sections

5.4.3 Lab Allocation

Practicals are assigned to labs based on:

- Capacity of 40 students (standard lab size)
- Availability during the required slot
- Presence of required facilities (if specified)

5.5 Output File Formats

5.5.1 Excel Timetable Structure

Department_Timetables.xlsx:

- One sheet per section
- Sheet names: CSE-Sem1-PRE-A, DSAI-Sem3-POST, etc.
- Layout: Days (rows) × Time slots (columns)
- Color-coded by course
- Merged cells for multi-slot sessions

Faculty_Timetables.xlsx:

- One sheet per faculty member
- Sheet names: Faculty names (sanitized for Excel)
- Shows all teaching assignments across sections

5.5.2 Excel Formatting Features

- **Headers**: Blue background, white text, bold
- **Day Labels**: Light blue background, bold
- **Course Cells**: Unique pastel colors per course
- **Lunch Slots**: Gray background (F0F0F0)
- **Breaks**: White background, small gray text
- **Cell Content**: Course name, session type, instructor, room

5.5.3 Exam Output Files

Exam_Schedule.csv:

- Columns: Date, Slot, Duration, Course Code, Course Name, Semester, Department, Students
- One row per exam per department
- Sorted by date, then slot

Exam Seating Files (Excel):

- Filename format: `Exam_[CourseCode][YYYYMMDD][Slot].xlsx`
- Example: `Exam_CS16120251201Morning.xlsx`
- Contains:
  - Exam Details sheet: Course info, date, time, student counts
  - Room sheets: Seating grids (COL1, COL2, COL3) with roll numbers

Chapter 6

Troubleshooting

6.1 Common Errors and Solutions

6.1.1 Import Errors

Error:

```
ModuleNotFoundError: No module named 'pandas'
```

Solution:

```
pip install -r requirements.txt
```

6.1.2 File Not Found Errors

Error:

```
Fatal Error: Course file not found at data/course.csv
```

Solution:

- Verify you are running the script from the project root directory
- Check that `data/course.csv` exists
- Ensure file permissions allow reading

6.1.3 CSV Parsing Errors

Error:

```
Warning: Skipping invalid course row (Line 45): ...
```

Causes and Solutions:

- **Extra commas**: Remove trailing commas in CSV rows
- **Missing required columns**: Ensure all column headers match exactly
- **Invalid data types**: Check that Credits, Semester are numbers
- **Special characters**: Use UTF-8 encoding, avoid special quotes

6.1.4 Scheduling Failures

Error:

```
Failed: CS161 (lecture 1 for CSE-Sem1-PRE-A - No slot)
```

Common Causes:

1. **Over-constrained schedule**: Too many courses for available slots
2. **Faculty conflicts**: Instructor has back-to-back classes without breaks
3. **Room shortage**: Not enough rooms with required capacity
4. **Daily limit violations**: Course already scheduled on that day

Solutions:

- Review course load for the semester
- Add more classrooms to `classroom_data.csv`
- Assign co-instructors to distribute load
- Adjust LTPSC values if too many sessions required

6.1.5 Web Interface Issues

Error: Timetable not found
Solution:

1. Go to Admin dashboard
2. Click "Re-Generate Timetable"
3. Wait for success confirmation
4. Refresh student/faculty portal

Error: Port 5000 already in use
Solution:

```
# Windows
netstat -ano | findstr :5000
taskkill /PID <PID> /F

# macOS/Linux
lsof -ti:5000 | xargs kill -9

# Or change port in web_app.py
app.run(debug=True, port=5001)
```

6.1.6 Excel Generation Issues

Error: Permission denied when saving Excel
Solution:

- Close any open Excel files in the `output/` folder
- Check write permissions on the output directory
- Manually delete old files if stuck

Error: Excel file is corrupted
Solution:

- Ensure `openpyxl` is properly installed: `pip install --upgrade openpyxl`
- Check available disk space
- Try regenerating the timetable

6.2 Performance Optimization

6.2.1 Slow Scheduling

If timetable generation takes more than 2-3 minutes:

- **Reduce course count**: Remove unnecessary electives
- **Increase rooms**: Add more classrooms to reduce search time
- **Simplify LTPSC**: Reduce L, T, P values where possible
- **Fewer iterations**: Comment out validation checks during testing

6.2.2 Memory Issues

For very large institutions (500+ courses):

- Increase Python memory limit
- Process semesters separately
- Use 64-bit Python installation

6.3 Data Quality Checks

6.3.1 Pre-Generation Checklist

Before generating timetables, verify:

1. All instructor names are consistent (no typos)
2. Room capacities are accurate

3. Registered student counts are up-to-date
4. LTPSC values match official syllabus
5. Pre/Post preferences are correctly set
6. Basket codes are assigned correctly

6.3.2 Post-Generation Verification

After generation, check:

1. Console shows "Validation PASSED"
2. Excel files open without errors
3. Spot-check a few sections for correctness
4. Verify faculty schedules have adequate breaks
5. Confirm lunch slots are properly placed

Chapter 7

Frequently Asked Questions (FAQ)

7.1 General Questions

7.1.1 Q1: What is the maximum number of courses the system can handle?

A: The system has been tested with 200+ course offerings across 4 semesters and 3 departments. Performance remains good up to 300-400 courses. Beyond that, consider processing semesters separately.

7.1.2 Q2: Can I customize the time slot duration?

A: Yes, but it requires modifying `src/utils.py`. Change the `SLOT_DURATION_MINS` constant. Note that session durations (90, 60, 120 min) should be multiples of the slot duration.

7.1.3 Q3: Does the system support weekend classes?

A: Currently, only Monday-Friday is supported. To add Saturday, modify the `DAYS` list in `src/utils.py`.

7.1.4 Q4: Can I use this for schools or coaching institutes?

A: Yes, with minor adaptations. You may need to:

- Adjust semester values (1, 3, 5, 7) to grade levels
- Modify lunch times
- Change room types if needed

7.2 Scheduling Questions

7.2.1 Q5: Why did some courses fail to schedule?

A: Common reasons:

1. Faculty has no free slots (teaching too many courses)
2. No room available with sufficient capacity
3. Daily limit reached (course already scheduled that day)
4. Combined class slot conflicts with other section commitments

Solution: Check the console output for specific failure reasons. The system will print messages like:

```
Failed: CS161 (lecture 1 for CSE-Sem1-PRE-A - No room)
```

7.2.2 Q6: How are electives handled across departments?

A: Electives are bundled into two categories:

- **Cross-departmental (Type 1)**: Grouped by semester and basket code. All departments can choose from these.
- **Department-specific (Type 2)**: Grouped by semester, department, and basket code. Only that department's students can enroll.

The system first reserves common slots, then assigns individual elective courses to those slots based on room and faculty availability.

7.2.3 Q7: What happens if an elective doesn't get a room in PRE-midsem?

A: If marked as `Pre /Post = elective`, it automatically overflows to POST-midsem scheduling and gets another chance to find a slot.

7.2.4 Q8: Can I have a 3-hour lecture?

A: The system currently supports:

- Lectures: 90 minutes (9 slots)
- Tutorials: 60 minutes (6 slots)
- Practicals: 120 minutes (12 slots)

To add 3-hour lectures, modify `LECTURE_SLOTS` in `src/utils.py` and update the LTPSC parsing logic in `src/models.py`.

7.3 Data Management Questions

7.3.1 Q9: How do I add a new course mid-semester?

Steps:

1. Open `data/course.csv`

2. Add a new row with all required columns filled
3. Save the file
4. Go to Admin dashboard (`http://localhost:5000/admin`)
5. Click "Re-Generate Timetable"

Warning

Re-generating will affect ALL timetables, not just the new course. Coordinate with departments before making changes mid-semester.

7.3.2 Q10: Can I rename instructors without breaking schedules?

A: Yes, but follow these steps:

1. Do a global find-and-replace in `course.csv`
2. Ensure consistency (exact spelling)
3. Re-generate timetables

Example: Changing "Dr. John Smith" to "John Smith":

- Find: `Dr. John Smith`
- Replace: `John Smith`

7.3.3 Q11: How do I handle co-instructors?

A: Use comma-separated names in the Instructor column:

```
Instructor: "Sunil P V, Sunil C K, Animesh Roy"
```

Behavior:

- For CSE Section A lectures: Uses first instructor
- For CSE Section B lectures: Uses second instructor
- For practicals: Uses third instructor (if available)

7.3.4 Q12: Can I assign different rooms to different sections of the same course?

A: Yes, this happens automatically. The scheduler finds the best available room for each section independently.

7.4 Web Interface Questions

7.4.1 Q13: Can multiple users access the web interface simultaneously?

A: Yes, Flask supports multiple concurrent users. However:

- Only one admin should generate timetables at a time
- Student/faculty viewing is safe for unlimited users

7.4.2 Q14: Can I host this on a server for remote access?

A: Yes, but you need to:

1. Change `app.run()` in `web_app.py`:

```
app.run(host='0.0.0.0', port=5000, debug=False)
```

2. Set up a production server (e.g., Gunicorn, uWSGI)
3. Use a reverse proxy (e.g., Nginx)
4. Configure firewall and SSL/TLS

Simple production command:

```
pip install gunicorn
gunicorn -w 4 -b 0.0.0.0:5000 web_app:app
```

7.4.3 Q15: How do I reset the system?

To clear all generated timetables:

1. Delete files in `output/` folder
2. Restart the web server
3. Generate fresh timetables from Admin dashboard

To reset data:

1. Replace CSV files in `data/` folder with backups
2. Re-generate timetables

7.5 Exam Scheduling Questions

7.5.1 Q16: How are exam dates assigned?

A: The exam scheduler follows these rules:

1. Starts from the date in `exam_config.csv`
2. Schedules one course per time slot (morning/afternoon)
3. Different departments can have exams at the same time
4. Moves to next day after afternoon slot is filled

7.5.2 Q17: How is exam seating arranged?

A: The two-pass column strategy:

1. **Odd columns** (C1, C3, C5): Filled with largest department
2. **Even columns** (C2, C4, C6): Filled with other departments

This ensures maximum spacing between students from the same department (reduces cheating risk).

7.5.3 Q18: Can I manually adjust exam seating?

A: Yes, after generation:

1. Open the Excel file for the specific exam
2. Go to the room sheet (e.g., "Room_C202")
3. Manually edit roll numbers in cells
4. Save the file

7.6 Technical Questions

7.6.1 Q19: What Python version is required?

A: Python 3.8 or higher. Tested on:

- Python 3.8.10
- Python 3.9.7
- Python 3.10.5
- Python 3.11.0

7.6.2 Q20: Can I run this on Linux/macOS?

A: Yes, the system is cross-platform. All Python code is OS-independent. File paths use `os.path.join()` for compatibility.

7.6.3 Q21: How do I contribute to the project?

A: This is an academic project by Team Byte_Me. For modifications:

1. Fork the repository
2. Make changes in a feature branch
3. Test thoroughly
4. Submit a pull request with detailed description

7.6.4 Q22: Is there an API for integration with other systems?

A: The web interface provides JSON APIs:

- `/api/section-list`: Get all section IDs
- `/api/faculty-list`: Get all faculty names
- `/api/student-timetable?id=<section_id>`: Get HTML timetable
- `/api/faculty-timetable?name=<faculty_name>`: Get HTML timetable
- `/api/admin-data`: Get system statistics

You can extend these APIs or add new ones in `web_app.py`.

Chapter 8

Best Practices

8.1 Data Management

8.1.1 Version Control for CSV Files

1. **Backup before changes**: Always keep a copy of working CSVs
2. **Use descriptive filenames**: `course_Fall2025_v1.csv`
3. **Track changes**: Use Git or maintain a changelog
4. **Validate before generation**: Spot-check data for consistency

8.1.2 Naming Conventions

Instructor Names:

- Be consistent: "Dr. John Smith" or "John Smith", not both
- Avoid special characters except hyphens and spaces
- Use title case for readability

Course Codes:

- Follow institutional standards (e.g., CS161, MA261)
- Keep codes short (under 10 characters)
- Avoid spaces or special characters

Room IDs:

- Use clear prefixes: C for Classroom, L for Lab
- Include floor number: C202 = Classroom on Floor 2
- Keep consistent format

8.1.3 Data Validation Checklist

Before generating timetables, verify:

Pre-Generation Checklist

- [ ] All instructor names are spelled consistently
- [ ] No duplicate course codes within same semester/department
- [ ] Room capacities are realistic
- [ ] Student counts are updated
- [ ] LTPSC values match official curriculum
- [ ] Pre/Post preferences are logically set
- [ ] No missing required columns in CSVs
- [ ] No trailing commas or empty rows in CSVs

8.2 Scheduling Strategy

8.2.1 Handling High-Load Semesters

If Semester 3 or 5 has too many courses:

1. **Add more classrooms**: Increase capacity in `classroom_data.csv`
2. **Use co-instructors**: Distribute teaching load
3. **Split large sections**: Divide Section A into A1 and A2
4. **Extend time range**: Modify `END_TIME_STR` in `utils.py`

8.2.2 Optimizing Elective Allocation

- **Limit basket size**: Keep 3-5 courses per basket
- **Balance baskets**: Distribute popular electives across baskets
- **Monitor overflow**: Check console for electives that overflow to POST
- **Assign TAs**: Use graduate TAs for smaller elective sections

8.2.3 Faculty Workload Management

- **Limit courses per faculty**: Ideally 2-3 courses per semester
- **Block teaching days**: Group courses on specific days for research time
- **Co-teach large courses**: Split lectures between 2 instructors
- **Review schedules**: Check faculty timetables for back-to-back classes

8.3 System Maintenance

8.3.1 Regular Backups

1. **Weekly**: Backup `data/` folder
2. **After generation**: Backup `output/` folder
3. **Monthly**: Full system backup including code

8.3.2 Update Management

When updating the system:

1. Test on a copy first
2. Generate test timetables with sample data
3. Compare outputs with previous version
4. Document any changes to logic or constraints

8.3.3 Performance Monitoring

Track these metrics over time:

- Generation time (should be under 60 seconds)
- Number of failed courses (aim for 0)
- Validation pass rate (should be 100%)
- Room utilization percentage
- Faculty workload distribution

8.4 Communication Workflow

8.4.1 Semester Planning Timeline

Timeline	Action
T-30 days	Collect course offerings from departments
T-21 days	Update `course.csv` and validate data
T-14 days	Generate draft timetables
T-10 days	Share with department heads for review
T-7 days	Incorporate feedback and regenerate
T-3 days	Final approval and distribution
T-0 days	Publish on web interface

Table 8.1: Recommended Semester Planning Timeline

8.4.2 Stakeholder Communication

For Students:

- Send email with web portal link
- Include instructions for finding their section
- Announce any changes via notification system

For Faculty:

- Share both web view and Excel file
- Highlight any back-to-back classes
- Provide contact for reporting issues

For Administrators:

- Provide validation report
- Share utilization statistics
- Document any scheduling challenges

8.5 Quality Assurance

8.5.1 Manual Verification Steps

After generation, perform these checks:

1. **Spot Check (5-10 sections)**:
   - Open random section timetables
   - Verify course details match `course.csv`
   - Check room assignments are logical
2. **Faculty Review**:
   - Check 3-5 busy faculty schedules
   - Confirm 30-min breaks exist
   - Verify no double bookings
3. **Resource Utilization**:
   - Calculate room occupancy rates
   - Identify underutilized time slots
   - Check for peak congestion times
4. **Student Experience**:
   - Review daily load balance
   - Check for excessive gaps between classes
   - Verify lunch slots are free

8.5.2 Testing New Configurations

When changing system parameters:

```
# 1. Create test data (small subset)
cp data/course.csv data/course_test.csv
# Edit course_test.csv to include only 20-30 courses

# 2. Modify code to use test data
# In main.py, change:
# COURSE_FILE = os.path.join(DATA_DIR, "course_test.csv")

# 3. Generate and verify
python main.py

# 4. If successful, apply to full dataset
```

Chapter 9

Appendices

9.1 Appendix A: Quick Reference

9.1.1 Common Commands

Task	Command
Generate timetables (CLI)	`python main.py`
Start web interface	`python web_app.py`
Generate exam schedules	`python src/exam_scheduler_main.py`
Install dependencies	`pip install -r requirements.txt`
Update dependencies	`pip install --upgrade -r requirements.txt`
Check Python version	`python --version`

Table 9.1: Common Commands Quick Reference

9.1.2 File Locations

File Type	Location
Course data	`data/course.csv`
Classroom data	`data/classroom_data.csv`
Student data	`data/students.csv`
Exam config	`data/exam_config.csv`
Output timetables	`output/Department_Timetables.xlsx`
Faculty schedules	`output/Faculty_Timetables.xlsx`
Exam schedules	`output/exams/Exam_Schedule.csv`

Table 9.2: File Locations Reference

9.1.3 Important URLs

Portal	URL
Home/Login	`http://localhost:5000/`
Student Portal	`http://localhost:5000/student`
Faculty Portal	`http://localhost:5000/faculty`
Admin Dashboard	`http://localhost:5000/admin`

Table 9.3: Web Interface URLs

9.2 Appendix B: System Constants

9.2.1 Time Configuration

Constant	Value	Description
Slot Duration	10 minutes	Base time unit
Start Time	09:00	First slot of the day
End Time	18:00	Last slot of the day
Total Slots/Day	54	Number of 10-min slots
Lecture Duration	90 minutes	9 slots
Tutorial Duration	60 minutes	6 slots
Practical Duration	120 minutes	12 slots
Faculty Break	30 minutes	3 slots
Student Break	10 minutes	1 slot

Table 9.4: Time Configuration Constants

9.2.2 Lunch Times by Semester

Semester	Lunch Start	Duration
Semester 1	12:30	30 minutes
Semester 3	13:00	30 minutes
Semester 5	13:30	30 minutes
Semester 7	12:30	30 minutes

Table 9.5: Lunch Times by Semester

9.3 Appendix C: CSV Templates

9.3.1 Minimal course.csv Template

```
1 Course Code,Course Name,Semester,Department,LTPSC,Credits,Instructor,Registered Students,Elective (Yes/No),Half
      Semester (Yes/No),Combined class,Pre /Post,Basket Code
2 CS101,Introduction to CS,1,CSE,3-1-0-0-4,4,John Doe,200,No,No,no,full,
3 MA101,Mathematics I,1,CSE,3-1-0-0-4,4,Jane Smith,200,No,No,yes,full,
4 CS151,Python Programming,1,CSE,2-0-0-0-2,2,Bob Johnson,0,Yes,Yes,no,elective,A
```

9.3.2 Minimal classroom_data.csv Template

```
Room Number,Type,Capacity,Facilities
C101,Classroom,100,Projector
C102,Classroom,100,Projector
L101,Lab,40,"Computers, Software"
C004,Classroom,240,"Projector, Audio System"
```

9.3.3 Minimal students.csv Template

```
roll_number,name,branch,section,semester
24BCS001,STUDENT ONE,CSE,A,1
24BCS002,STUDENT TWO,CSE,A,1
24BDS001,STUDENT THREE,DSAI,A,1
```

9.4 Appendix D: Error Codes

9.4.1 Scheduling Error Messages

Error Message	Meaning & Solution
`No slot`	No common free time found for all required sections. **Solution:** Add more time slots or reduce course load.
`No room`	No room with sufficient capacity available. **Solution:** Add classrooms or reduce class size.
`Faculty conflict`	Instructor already teaching at this time. **Solution:** Assign co-instructor or reschedule.
`C004 missing`	240-seater auditorium not found in data. **Solution:** Add C004 to `classroom_data.csv`.
`Daily limit violation`	Course already scheduled once on this day. **Solution:** This is a soft warning; check if intentional.
`LTPSC Mismatch`	Scheduled sessions don't match required LTPSC. **Solution:** Verify LTPSC values and regenerate.
`Missing student break`	No 10-min break after a class. **Solution:** System should auto-insert; check for bugs.

9.5 Appendix E: Glossary

LTPSC
Lecture-Tutorial-Practical-Self Study-Credit format used to define course structure

PRE-Midsem
First half of the semester (before mid-semester exams)

POST-Midsem
Second half of the semester (after mid-semester exams)

Combined Class
A class where multiple sections attend together, typically in a large auditorium

Elective
Optional course that students can choose from a basket of courses

Basket
A group of elective courses from which students select one

Section
A subdivision of students within a department (e.g., CSE-A, CSE-B)

Slot A 10-minute time unit used for scheduling

Session
A single instance of a lecture, tutorial, or practical

Pseudo-Course
An internal placeholder used for bundled electives/baskets

Overflow
Elective courses that couldn't be scheduled in PRE and move to POST

Daily Limit
Constraint ensuring no course has more than one lecture+tutorial and one lab per day

Faculty Break
Mandatory 30-minute gap between consecutive classes for instructors

Student Break
10-minute break after each class for students

LTPSC Fulfillment
Verification that all required sessions have been scheduled

Validation
Post-scheduling checks for conflicts and constraint violations

9.6 Appendix F: Sample Outputs

9.6.1 Sample Console Output

```
Starting Automated Timetable Scheduler...
Loading classrooms from data/classroom_data.csv...
Successfully loaded 29 classrooms.

Loading and processing courses from data/course.csv...
Loaded and validated 150 course offerings for Semesters 1, 3, 5, 7.
Bundling electives and baskets...
Found 9 cross-departmental ELECTIVE bundles (Type 1).
Found 16 department-specific BASKET bundles (Type 2).
Bundling complete. Total processed courses: 145

--- INITIALIZING PRE MIDSEM SCHEDULING RUN ---
  Running Phase 4: Combined Classes (is_combined=yes)
  Running Phase 3: Elective/Basket Slots (is_pseudo_basket=true)
  Running Phase 5/6: Core Courses (is_combined=no)
  Running Phase 8: Assigning Electives to Rooms/Faculty
    Assigning slot: Elective (A) (lecture) at Monday 09:00
      -> SUCCESS: Booked PH151 in C101
      -> SUCCESS: Booked DS151 in C102
--- PRE RUN COMPLETE (Sem 1) ---

--- INITIALIZING POST MIDSEM SCHEDULING RUN ---
  Running Phase 4: Combined Classes (is_combined=yes)
  Running Phase 3: Elective/Basket Slots (is_pseudo_basket=true)
  Running Phase 5/6: Core Courses (is_combined=no)
  Running Phase 8: Assigning Electives to Rooms/Faculty
--- POST RUN COMPLETE (Sem 1) ---

--- All Scheduling Runs Complete ---
Total sections generated: 20

Validating PRE-Midsem master schedule...
--- RUNNING POST-SCHEDULING VALIDATION ---
Validation PASSED: All timetables are conflict-free and LTPSC is fulfilled.

Validating POST-Midsem master schedule...
--- RUNNING POST-SCHEDULING VALIDATION ---
Validation PASSED: All timetables are conflict-free and LTPSC is fulfilled.

Exporting department timetables to output/Department_Timetables.xlsx...
Successfully saved department timetables to output/Department_Timetables.xlsx

Exporting faculty timetables to output/Faculty_Timetables.xlsx...
Successfully saved faculty timetables to output/Faculty_Timetables.xlsx

--- Timetable Generation Complete. ---
Output files are in the 'output' directory.
```

9.6.2 Sample Timetable Cell

A typical cell in the generated Excel timetable contains:

Sample Timetable Cell

Problem Solving with Python
(Lecture)
Sunil P V, Sunil C K
C101

9.6.3 Sample Exam Seating Layout

COL1	COL2	COL3
24BCS001	24BDS001	24BCS002
24BCS003	24BDS002	24BCS004
24BCS005	24BEC001	24BCS006
...		

Table 9.7: Sample Exam Seating Pattern (COL1 & COL3 = CSE, COL2 = Mixed)

9.7 Appendix G: Changelog

9.7.1 Version History

Version	Date	Changes
1.0	Nov 2025	Initial release with all core features

Table 9.8: Version History

9.7.2 Known Issues

1. **Long Faculty Names**: Names over 31 characters are truncated in Excel sheet names
2. **Unicode Characters**: Some special characters in course names may not display correctly in Excel
3. **Browser Caching**: Web interface may cache old timetables; use Ctrl+F5 to force refresh

9.7.3 Future Enhancements

Planned features for future versions:

- Weekend class support
- Mobile-responsive web interface
- Direct database integration
- Automated conflict resolution suggestions
- Historical timetable comparison
- Room change notifications
- iCalendar export for student devices

Chapter 10

Support and Contact

10.1 Getting Help

10.1.1 Self-Help Resources

Before contacting support, try these resources:

1. **This Manual**: Check relevant sections and FAQ
2. **Console Output**: Read error messages carefully
3. **Validation Reports**: Review validation output for specific issues
4. **Log Files**: Check for detailed error traces (if logging is enabled)

10.1.2 Reporting Issues

When reporting a problem, include:

Issue Report Template

Issue Title: Brief description
Environment:

- Operating System: Windows 10 / macOS 12 / Ubuntu 20.04
- Python Version: `python --version`
- Browser (if web issue): Chrome 120 / Firefox 115

Steps to Reproduce:

1. Step 1
2. Step 2
3. Step 3

Expected Behavior: What should happen
Actual Behavior: What actually happened
Error Messages: Copy-paste exact error text
Attachments:

- Screenshot (if relevant)
- Sample CSV file (anonymized)
- Console output

10.1.3 Contact Information

Type	Contact
Development Team	Team Byte_Me, IIIT Dharwad
Technical Support	[Your institutional email/support system]
Documentation	This user manual (v1.0)
Bug Reports	[Your GitHub issues page / ticketing system]

Table 10.1: Contact Information

10.2 Contributing

10.2.1 Code Contributions

To contribute code improvements:

1. Fork the repository
2. Create a feature branch: `git checkout -b feature-name`
3. Make changes and test thoroughly

4. Document changes in code comments
5. Submit pull request with detailed description

10.2.2 Documentation Improvements

To improve this manual:

- Report unclear sections
- Suggest additional examples
- Provide screenshots for complex procedures
- Translate to other languages (if needed)

10.3 License and Terms

10.3.1 Software License

This software is developed for academic purposes by Team Byte_Me for IIIT Dharwad. Please refer to the LICENSE file in the repository for specific terms.

10.3.2 Disclaimer

Important Disclaimer

This software is provided "as is" without warranty of any kind. While every effort has been made to ensure accuracy and reliability:

- Always verify generated timetables manually
- Maintain backups of critical data
- Test changes in a non-production environment
- The development team is not liable for scheduling conflicts

Users assume all responsibility for deployment and usage decisions.

10.4 Acknowledgments

10.4.1 Development Team

Team Byte_Me:

- Shriyans S Sahoo (24BCS142)
- Syed Mahdee Hussain (24BCS151)
- Padala Gnandeep (24BCS095)
- Phanishree N (24BCS101)

Institution: Indian Institute of Information Technology Dharwad
Guidance: Dr. Vivekraj V K

10.4.2 Technologies Used

This project is built with:

- **Python 3**: Core programming language
- **Flask**: Web framework
- **pandas**: Data manipulation
- **openpyxl**: Excel file generation
- **HTML/CSS/JavaScript**: Web interface

Revision History

Version	Date	Author	Changes
1.0	Nov 2025	Team Byte_Me	Initial release

Table 10.2: Document Revision History

End of Document

For the latest version of this manual,
please contact the development team.