GOVERNMENT POLYTECHNIC COLLEGE ANANTNAG

MAJOR PROJECT REPORT

ON

ClassMesh – Timetable Generation System

(Standalone Windows Application)

SUBMITTED BY

Musaib Nazir

Under Registration No. **31-01-R21-4268**

From Batch 2022-2025

Under the Mentorship of

Er. Shakeel Ahmad

GOVERNMENT POLYTECHNIC COLLEGE ANANTNAG

MAJOR PROJECT REPORT

ON

ClassMesh – Timetable Generation System

(Standalone Windows Application)

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE THREE-YEAR DIPLOMA IN

COMPUTER ENGINEERING

SUBMITTED BY

Musaib Nazir

Under Registration No. **31-01-R21-4268**

From Batch 2022-2025

Mentor

Er. Shakeel Ahmad

Principal **Department Head**

CERTIFICATE

This is to certify that ***Musaib Nazir*** under registration number **31-01-R21-4268** , from batch 2022-2025, has successfully completed the project work entitled "**ClassMesh - Timetable Generation System**" as a partial fulfilment for the award of the Three-Year Diploma in Computer Engineering from Government Polytechnic College, Anantnag, Kashmir, during the academic year 2024-2025. This work has been carried out under my supervision and guidance.

Principal	**Department Incharge**	**Project Mentor**
Er. Ashaq Nabi Butt	Er. Shakeel Ahmad	Er. Shakeel Ahmad

DECLARATION

I, **Musaib Nazir,** hereby declare that the Major Project Report entitled "ClassMesh - Timetable Generation System" submitted in partial fulfilment of the requirements for the award of the Three Year Diploma in Computer Engineering to Government Polytechnic College, Anantnag, is a record of bonafide work carried out by me .

The design of the application, including the user interface, system architecture, and the core timetable generation logic (referred to as generateTimetableAlgorithmV2 within this report), are my original contributions and were independently conceived, designed, developed, and implemented by me. This work has not been copied from any other source.

I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

(Owners Signature) **Musaib Nazir**

ACKNOWLEDGEMENT

I wish to express my sincere gratitude to my mentor, **Er. Shakeel Ahmad**, *Department of Computer Engineering* for constant encouragement, and insightful suggestions throughout the course of this major project. His inspiration and support were instrumental in the successful completion of **"ClassMesh – Timetable Generation System".**

My heartfelt thanks go to my parents and friends for their unwavering support, patience, and motivation during this endeavour. Their belief in me has always been a great source of strength.

Finally, I am grateful for the opportunity to work on this project, which has been a significant learning experience, enhancing my skills in software development, problem-solving, and project management.

Musaib Nazir

ABSTRACT

The creation of academic timetables in educational institutions is a complex and often tedious task when performed manually. This project, "ClassMesh," presents a JavaFX-based desktop application designed to automate and streamline the timetable generation process. The system aims to minimize scheduling conflicts and reduce the manual effort involved by providing a user-friendly interface for inputting various institutional constraints and preferences.

ClassMesh allows users to configure essential parameters such as school/department details, working hours, break durations, class slot lengths, maximum daily teaching loads for teachers, and maximum consecutive classes per subject. It supports dynamic input for multiple semesters/classes, along with their respective subjects and weekly hour requirements. Teachers can be defined and mapped to the specific subjects they are qualified to teach across different semesters, with built-in checks to prevent duplicate assignments of a unique subject-semester context.

The core of the application is a heuristic-based algorithm (generateTimetableAlgorithmV2) that processes these inputs to construct a viable timetable. This algorithm considers teacher availability, subject hour fulfilment, and specified constraints to schedule classes. The generated timetable is displayed in a clear, grid-based format, showing days, time slots, semesters, subjects, and assigned teachers.

Key functionalities include the ability to save the entire configuration to a JSON file for later retrieval and reuse, and to export the final timetable into standard PDF and Excel formats for easy distribution and printing. A basic MAC address-based security feature is also implemented to restrict unauthorized application use. The project successfully demonstrates a practical software solution to a common administrative challenge in academic settings.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS / NOTATIONS / NOMENCLATURE

Below is a list of abbreviations used in this report:

- **GUI:** Graphical User Interface
- **JDK:** Java Development Kit
- **JSON:** JavaScript Object Notation
- **MAC:** Media Access Control
- **PDF:** Portable Document Format
- **XLSX:** XML Spreadsheet (Microsoft Excel Open XML)
- **HOD:** Head of Department
- **OS:** Operating System
- **RAM:** Random Access Memory
- **IDE:** Integrated Development Environment
- **UI:** User Interface
- **FR:** Functional Requirement
- **NFR:** Non-Functional Requirement

CHAPTER 1: INTRODUCTION

1.1 Project Overview

ClassMesh is a desktop application developed using JavaFX, designed to automate the creation of academic timetables for educational institutions like schools and colleges. It aims to replace manual, time-consuming, and error-prone methods with an efficient, user-friendly, and configurable software solution. The system takes various inputs such as institutional details, time constraints, semester structures, subject details, teacher information, and their subject assignments to generate an optimized and conflict-free timetable.

1.2 Problem Statement

Manual timetable generation is a significant administrative challenge in educational institutions. It involves balancing numerous constraints, including: ensuring subjects are allocated sufficient hours as per curriculum; avoiding clashes for teachers teaching multiple subjects or across different semesters; distributing workload evenly among teachers; adhering to specified working hours and break times; and managing maximum consecutive classes for subjects and maximum daily load for teachers. This complex combinatorial problem often leads to suboptimal schedules, requires considerable time and effort, and is difficult to adapt when changes occur (e.g., a teacher's unavailability). An automated system is needed to streamline this process, reduce manual effort, minimize errors, and produce efficient timetables.

1.3 Purpose and Scope

The primary purpose of ClassMesh is to provide a robust and intuitive tool for generating academic timetables. The scope of the project includes: **User Interface** (a graphical user interface (GUI) for easy input of all necessary data); **Configuration** (allowing users to define School/College and Department names, working hours, break durations, class slot durations, maximum daily teaching slots per teacher and maximum consecutive slots per subject, number of semesters/classes, subjects per semester including names and weekly hours, number of teachers and their names, and mapping of teachers to the subjects they can teach across different semesters); **Timetable Generation** (implementing an algorithm to automatically generate a conflict-free timetable based on the provided configurations and constraints); **Display** (presenting the generated timetable in a clear, grid-based format); **Export** (allowing users to export the generated timetable to PDF and Excel formats); **Persistence** (enabling users to save the current input configuration to a file and load it later); and **Security** (basic MAC address-based restriction for application usage). The system is designed for small to medium-sized departments or

institutions where a centralized desktop solution is feasible. It does not currently include features like multi-user access, real-time updates, or integration with other institutional management systems.

1.4 Objectives

The main objectives of the ClassMesh project are:

1. To design and develop a user-friendly desktop application for timetable generation.
2. To allow comprehensive configuration of institutional parameters, time settings, semesters, subjects, and teachers.
3. To implement an effective algorithm for automated, constraint-based timetable generation.
4. To provide a clear visual representation of the generated timetable.
5. To enable exporting of the timetable into standard formats (PDF, Excel) for distribution and printing.
6. To facilitate saving and loading of user configurations for reusability.
7. To ensure the generated timetable is conflict-free and attempts to meet specified constraints.

1.5 Target Audience

The primary target audience for ClassMesh includes: administrators in schools and colleges responsible for creating academic timetables; Heads of Departments or academic coordinators; and small to medium-sized educational institutions seeking an affordable and easy-to-use timetable solution.

1.6 Report Structure

This report is organized into the following chapters: **Chapter 1: Introduction** (provides an overview of the project, problem statement, purpose, scope, objectives, and target audience); **Chapter 2: System Requirements** (details the software, hardware, functional, and non-functional requirements for the application); **Chapter 3: System Design and Architecture** (describes the overall architecture, data models, and key components of the system); **Chapter 4: Implementation Details** (explains the technologies used, UI implementation, core logic including the timetable algorithm, data handling, and error management); **Chapter 5: System Features and Functionality** (elaborates on the various features offered by ClassMesh); **Chapter 6: Testing** (discusses the testing strategies employed and key test cases); **Chapter 7: Results and Discussion** (presents the outcomes, strengths, and limitations of the developed system); **Chapter 8: Conclusion and Future Scope** (summarizes the project and suggests potential areas for future development).

CHAPTER 2: SYSTEM REQUIREMENTS

2.1 Software Requirements

- **Operating System:** Windows, macOS, or Linux (any OS that supports Java).
- **Java Development Kit (JDK):** Version 11 or higher.
- **JavaFX SDK:** Version 11 or higher (if not included with JDK).
- **Libraries:** iText 7 Core (for PDF export), Apache POI (for Excel export), Jackson Databind (for JSON serialization/deserialization).
- **Integrated Development Environment (IDE):** (For development) e.g., IntelliJ IDEA, Eclipse.

2.2 Hardware Requirements

- **Processor:**Minimum Intel Core i3 or equivalent AMD processor.
- **RAM:** Minimum 4 GB (8 GB recommended).
- **Hard Disk Space:** Minimum 200 MB.
- **Display:** Standard display monitor (1024x768 or higher).
- **Input Devices:** Keyboard and Mouse.

2.3 Functional Requirements

The ClassMesh application shall:

1. **Functional Requirement 1 (FR1):** Restrict application usage based on a predefined MAC address.
2. **FR2:** Allow users to input School/College Name and Department Name.
3. **FR3:** Allow users to set Work Start/End Time, Break Start/End Time, Class Slot Duration, Max Classes per Teacher/Day, and Max Consecutive Classes/Subject. Display Net Working Hours.
4. **FR4:** Allow users to specify the number of Semesters/Classes. For each semester: input Semester Name, number of Subjects. For each subject: input Subject Name, Weekly Hours. Validate/adjust total subject hours.
5. **FR5:** Allow users to specify the number of Teachers. For each teacher: input Name, number of Subjects taught, assign specific subjects (from defined semesters), preventing duplicate

assignments of the same subject context.

6. **FR6:** Generate a timetable based on all configurations, attempting to schedule all hours, respect teacher availability/load, max consecutive slots, and work/break times.
7. **FR7:** Display the timetable in a grid (Days, Time Slots, Semesters, Subjects, Teachers), indicating breaks and institution names.
8. **FR8:** Allow export to PDF and Excel, including institution names and proper formatting.
9. **FR9:** Allow saving current configuration to JSON and loading a saved configuration.
10. **FR10:** Provide an intuitive tabbed interface, guiding users sequentially, with clear labels and feedback.
11. **FR11:** Provide an "About" dialog with application/developer details.
12. **FR12:** Allow users to reset inputs when loading a new configuration.

2.4 Non-Functional Requirements

1. **NFR1: Usability:** Easy to learn and use with an intuitive GUI.
2. **NFR2: Performance:** Timetable generation for moderate data within a reasonable time; responsive UI.
3. **NFR3: Reliability:** Consistent function without crashes; accurate generation based on inputs.
4. **NFR4: Maintainability:** Well-structured and commented code.
5. **NFR5: Scalability (Limited):** Handle up to 12 semesters, 50 subjects/sem, 25 teachers.
6. **NFR6: Security (Basic):** MAC address validation; configuration files human-readable (JSON).
7. **NFR7: Error Handling:** Informative error messages for invalid inputs or failures.
8. **NFR8: Platform Independence:** Run on any platform with Java/JavaFX.

CHAPTER 3: SYSTEM DESIGN AND ARCHITECTURE

3.1 Application Architecture

ClassMesh employs a desktop-based, event-driven architecture typical of JavaFX applications, resembling a Model-View-Controller (MVC) pattern. The **View** comprises JavaFX UI components handled within the `ClassMesh` class. Event handling methods in `ClassMesh` act as **Controllers**, responding to user actions, processing inputs, interacting with the Model, and updating the View. The **Model** consists of data structures (`Subject`, `Semester`, `Teacher`, `SubjectContext`, `TimetableEntry`, configuration records) and business logic, including the `generateTimetableAlgorithmV2`. Key characteristics include a single main `ClassMesh` class, event-driven flow, modular UI construction via dedicated methods, and dynamic UI generation for elements like semester/teacher rows.

3.2 Data Model

The application uses several custom Java classes and records to model domain-specific data.

3.2.1 Subject Class

Attributes: `name` (String), `weeklyHours` (int). Purpose: Represents an academic subject. Key Methods: `equals()` and `hashCode()` overridden to consider subjects equal if names match.

3.2.2 Semester Class

Attributes: `name` (String), `subjects` (List<Subject>). Purpose: Represents an academic period or class group. Key Methods: `equals()` and `hashCode()` based on semester name.

3.2.3 Teacher Class

Attributes: `name` (String), `subjectsTaught` (List<Subject>). Purpose: Represents a teacher. Key Methods: `equals()` and `hashCode()` based on teacher's name.

3.2.4 SubjectContext Class

Attributes: `subject` (Subject), `semester` (Semester). Purpose: Uniquely identifies a subject within a specific semester, crucial for teacher assignments. Key Methods: `toString()` for user-friendly representation, `equals()` and `hashCode()` based on both subject and semester.

3.2.5 TimetableEntry Class

Attributes: `day` (String), `startTime` (LocalTime), `endTime` (LocalTime), `semester` (Semester), `subject` (Subject), `teacher` (Teacher). Purpose: Represents a single scheduled class slot.

3.2.6 Configuration Data Structures (Records)

These records (`TimeConfiguration`, `SubjectDetailConfig`, `SemesterConfigData`, `TeacherAssignmentConfig`, `TeacherConfigData`, `AppConfiguration`, `TeacherRestoreState`) are primarily used for saving/loading application state to/from JSON and for temporarily holding UI data.

3.3 Key Modules/Components

The application's functionality is delivered through several interconnected logical modules:

3.3.1 User Interface (UI) Module

Responsibility: Handles all user interactions, data input, and display. Implementation: Built with JavaFX (`TabPane`, `GridPane`, various controls), with dynamic UI generation for semesters, subjects, and teachers. Key Methods: `start()`, `create...Section()` methods, `displayTimetableGrid()`.

3.3.2 Configuration Management Module

Responsibility: Manages user-defined settings and constraints. Implementation: Involves storing UI inputs into model objects, validating inputs, and populating UI from loaded configurations. Key Methods: `validateAndStoreTimeConfig()`, `handleSemesterNextClick()`, `populate...FromConfig()` methods.

3.3.3 Timetable Generation Engine

Responsibility: Implements the core algorithm (`generateTimetableAlgorithmV2()`) to create a schedule. Implementation: Calculates time slots, maps subjects to teachers, tracks required vs. scheduled slots, iterates through days/slots, manages teacher availability/load, and enforces constraints like max consecutive subject slots.

3.3.4 Data Persistence Module

Responsibility: Enables saving/loading of application configuration. Implementation: Uses Jackson Databind for JSON serialization/deserialization of `AppConfiguration`. JavaFX `FileChooser` for file selection. Key Methods: `handleSaveConfiguration()`, `handleLoadConfiguration()`.

3.3.5 Export Functionality Module

Responsibility: Allows exporting the timetable. Implementation: Uses iText 7 for PDF export (`handleExportPdf()`, `createPdfTimetableTable()`) and Apache POI for Excel export (`handleExportExcel()`, `createExcel...Style()` helpers). `FileChooser` for file selection.

3.3.6 Security Module (Basic)

Responsibility: Restricts application usage. Implementation: Compares system MAC address(es) against a hardcoded `ALLOWED_MAC_ADDRESS`. Key Methods: `isOperationAllowed()`, `getCurrentMacAddresses()`.

CHAPTER 4: IMPLEMENTATION DETAILS

4.1 Technologies Used

ClassMesh is developed using **Java SE** (Version 11+) and **JavaFX** for the GUI. Key third-party libraries include **iText 7 (Core)** for PDF creation, **Apache POI** for Excel (XLSX) file generation, and **Jackson Databind** for JSON processing (saving/loading configurations). Standard Java libraries such as Java Time API, Java Collections Framework, Java I/O, and Java Networking (for MAC address retrieval) are also utilized extensively.

4.2 User Interface Implementation

The UI is built with JavaFX, emphasizing a step-by-step configuration flow.

4.2.1 Main Application Window and Navigation

The main `Stage` contains a `Scene` with a `TabPane` (`mainTabPane`) for "Input Details" and "Generated Timetable". A `MenuBar` provides File (Load/Save) and Help (About) options. Styling is via `modern-light.css` and inline styles.

4.2.2 Institution Information Section (`createSchoolInfoSection`)

Uses `TextField` elements (`schoolNameField`, `departmentNameField`) for school/department names.

4.2.3 Time Configuration Section (`createTimeConfigurationSection`)

Employs `ComboBox` elements for Work Start/End, Break Start/End, and Slot Duration. `Spinner` elements for Max Classes/Teacher/Day and Max Consecutive Classes/Subject. A `Label` (`workingHoursLabel`) dynamically shows net working hours. Validation is triggered before proceeding via "Next -> Define Semesters" button.

4.2.4 Semester and Subject Configuration Section (`createSemesterInputSection`, `buildSemesterSubjectGrid`, `createAndAddSemesterRowUI`)

A `ComboBox` (`semCountComboBox`) selects semester count. A `GridPane` (`semesterSubjectGrid`) dynamically shows fields per semester: a `TextField` for semester name, a `ComboBox` for subject count, and a `FlowPane` (`subDetailPane`) dynamically adding `TextField` (subject name) and `Spinner` (weekly hours) for each subject. Logic

(`adjustHoursWithinSemester`) helps manage total subject hours against available weekly time. UI updates non-destructively.

4.2.5 Teacher Configuration and Mapping Section (`createTeacherMappingSection`, `populateTeacherMappingGrid`, `createAndAddTeacherRowUI`)

A `ComboBox` (`teacherCountComboBox`) selects teacher count. A `GridPane` (`teacherMappingGrid`) dynamically shows fields per teacher: `TextField` for name, `ComboBox` for subject count, and a `FlowPane` (`tSubMapFlowPane`) dynamically adding `ComboBox` elements for selecting `SubjectContext` objects. Custom `ListCell` factories manage display and prevent duplicate `SubjectContext` assignments using `globallySelectedSubjectContexts`. "Generate Timetable" button visibility depends on valid configuration.

4.2.6 Timetable Display (`createTimetableTab`, `displayTimetableGrid`)

The "Generated Timetable" tab uses a `GridPane` (`timetableDisplayGrid`) within a `ScrollPane`. `displayTimetableGrid()` populates it with headers (Days, Time Slots), break period indicators, and cells showing Subject Name and Teacher Name. School/Department names are displayed above. Export buttons are enabled if a timetable exists.

4.3 Core Logic

4.3.1 Input Validation

Time configuration is validated by `validateAndStoreTimeConfig()` for logical sequences and completeness. `adjustHoursWithinSemester()` provides feedback on subject hour allocation. Teacher configuration checks for names and consistent subject selections. `checkIfReadyToGenerate()` controls generation readiness. Errors are shown via `showErrorAlert()` and field highlighting.

4.3.2 Timetable Generation Algorithm (V2)

This heuristic algorithm initializes by calculating time slots, mapping subjects to teachers, and determining required slots per subject. It then iterates through days, slots, and semesters, attempting to schedule subjects. Key checks include: maximum consecutive subject slots (using `recentSubjectsMap`), teacher availability (not busy in `teacherBusyMap`, not exceeding daily load in `teacherDailyLoadMap`). If a valid teacher is found, a `TimetableEntry` is created, and tracking structures are updated. Console outputs (`printUnmetNeeds`, `printTeacherLoad`) provide diagnostic information.

4.3.3 Constraint Handling

Constraints are managed at UI level (spinner min/max, hour adjustments), pre-generation validation (time config), and within the algorithm (max teacher slots/day, max consecutive subject slots, teacher availability, subject hour fulfillment, work/break times).

4.4 Data Handling

4.4.1 Saving and Loading Configuration

Saving involves gathering UI state into an `AppConfiguration` object, serializing it to JSON using Jackson `ObjectMapper`, and writing to a user-selected file. Loading involves reading a JSON file, deserializing into `AppConfiguration`, resetting the UI (`resetUIAndDataToDefaults`), and repopulating UI fields and internal data structures, using `Platform.runLater()` for UI updates.

4.4.2 Exporting Timetable Data

Checks for `lastGeneratedTimetable`. Excel export uses Apache POI to create an `XSSFWorkbook`, adding title rows, headers, and data cells with styling. PDF export uses iText 7 to create a document with institution names and a formatted `Table` of the timetable. `FileChooser` is used for save locations.

4.5 Error Handling and User Feedback

`showErrorAlert()` and `showInfoAlert()` display messages. `highlightError()` visually marks fields with errors. Button states (enabled/disabled) guide user actions. UI sections are progressively disclosed. Console output provides generation diagnostics.

CHAPTER 5: SYSTEM FEATURES AND FUNCTIONALITY

5.1 Institution Information Input

Users can input School/College Name and Department Name via `TextFields`. This information is used for display on the timetable and in exports.

5.2 Time Configuration

Allows setting Work Start/End Times, Break Start/End Times, Class Slot Duration, Max Classes per Teacher/Day, and Max Consecutive Classes per Subject. Net Working Hours are dynamically displayed. Validation ensures logical settings.

5.3 Semester and Subject Management

Users select Number of Semesters. The UI dynamically provides fields for Semester Name and Number of Subjects. For each subject, Name and Weekly Hours are input. `adjustHoursWithinSemester` helps manage total hours against available time.

5.4 Teacher and Subject Assignment

Users select Number of Teachers. For each, Name and Number of Subjects taught are input. `ComboBoxes` list available `SubjectContext` objects (Subject + Semester) for assignment, preventing duplicate assignments of a unique context using `globallySelectedSubjectContexts`.

5.5 Automated Timetable Generation

Triggered by "Generate Timetable" button, `generateTimetableAlgorithmV2` processes inputs considering working days, time slots, subject hours, teacher availability, daily load limits, and consecutive slot limits to create a schedule.

5.6 Interactive Timetable Display

The "Generated Timetable" tab shows the schedule in a `GridPane` within a `ScrollPane`. Rows represent semesters under days, columns represent time slots. Cells show Subject Name and Teacher Name. Breaks are marked. Institution names are displayed.

5.7 Data Export (PDF and Excel)

Exports the timetable to `.xlsx` (Excel) using Apache POI and `.pdf` using iText 7. Files include institution names, formatted headers, and timetable data. `FileChooser` is used for saving.

5.8 Configuration Persistence (Save/Load)

Via "File" menu: "Save Configuration" serializes all inputs to a JSON file. "Load Configuration" parses a JSON file and repopulates the UI and internal data, restoring the saved state.

5.9 Security (MAC Address Restriction)

A hardcoded `ALLOWED_MAC_ADDRESS` is checked against the host machine's MAC address(es) on startup (`isOperationAllowed()`). If no match, an error is shown, and the application exits.

5.10 About Section

Via "Help" -> "About ClassMesh", an information dialog (`showAboutDialog`) displays application name, version, developer details, a brief description, and a logo if available.

CHAPTER 6: TESTING

6.1 Testing Approach

A multi-faceted approach including **Unit Testing** (conceptual, for individual components), **Integration Testing** (interaction between modules), **System/Functional Testing** (whole application against requirements), **User Interface Testing** (GUI intuitiveness, responsiveness), and **Usability Testing** was considered.

6.2 Unit Testing (Conceptual)

Targets: Data Model Classes (constructors, getters, `equals()`, `hashCode()`, `toString()`); Utility Methods (`calculateTimeSlots()`, validation logic); Configuration Records (serialization/deserialization in isolation).

6.3 Integration Testing (Conceptual)

Focuses on: UI to Configuration Data flow; Configuration Data to Algorithm input; Algorithm to UI Display rendering; Save/Load integration ensuring state restoration; Export integration ensuring correct data passage to iText/POI.

6.4 User Interface Testing

Manual interaction to verify: Navigation (tabs, buttons, menus); Input Controls (`ComboBoxes`, `TextFields`, `Spinners`); Dynamic UI Updates (semester/subject/teacher rows, subject availability in ComboBoxes); Feedback (error/info messages, field highlighting); Responsiveness; Layout and Appearance.

6.5 Functional Testing - Key Test Cases

6.5.1 Valid Timetable Generation

Tested simple scenarios (1 semester, few subjects, 1 teacher) and more complex ones (multiple semesters/teachers). Also considered highly constrained scenarios to observe behavior (partial generation, unmet needs feedback).

6.5.2 Constraint Adherence

Verified Max Teacher Slots/Day, Max Consecutive Subject Slots, Work/Break Times, and Subject Weekly Hours allocation through specific input configurations designed to test these limits.

6.5.3 Export Functionality

Generated timetables and exported to Excel and PDF, verifying file creation, formatting, and data accuracy against GUI. Tested behavior for exporting empty/failed timetables (buttons disabled/message shown).

6.5.4 Save/Load Configuration

Entered full configurations, saved, cleared UI, loaded the config, and verified accurate restoration of all inputs and subsequent timetable generation consistency. Tested loading invalid files (error message, state unchanged).

6.5.5 Input Validation and Error Handling

Entered conflicting time configurations, incomplete teacher/subject details, and observed correct error alerts, field highlighting, and prevention of further actions until resolved.

6.6 Test Results Summary

The ClassMesh application underwent testing focusing on its core functionalities and performance under various conditions. Critical path functionalities, including comprehensive data input for institution details, time configurations, semester and subject structures, and teacher assignments, were validated. The timetable generation algorithm (generateTimetableAlgorithmV2) was tested with a range of scenarios, from simple configurations to more complex setups designed to simulate challenging, near worst-case conditions based on the input parameters available within the application's scope (e.g., maximum number of semesters, subjects per semester, and teachers with varied constraints). In these test scenarios, the application demonstrated robust performance, successfully generating timetables that adhered to the specified constraints. The algorithm effectively handled typical constraints, and the overall system remained stable. The "Save Configuration" and "Load Configuration" features were tested to ensure data integrity and accurate restoration of user inputs, which performed as expected. Export functionalities to both PDF and Excel formats were also verified, with the generated files accurately reflecting the displayed timetable and maintaining proper formatting. No significant UI discrepancies or critical issues that would prevent core functionality were observed during testing with the available input ranges and designed scenarios. The application performed reliably even when pushed with complex data sets permissible by its input controls.

CHAPTER 7: RESULTS AND DISCUSSION

7.1 Achievement of Objectives

The ClassMesh project successfully met its primary objectives by delivering a user-friendly JavaFX application for automated timetable generation. It allows comprehensive configuration of institutional parameters, time settings, academic structures, and teacher assignments. An effective algorithm (`generateTimetableAlgorithmV2`) was implemented for constraint-based timetable creation. The system provides clear visual representation of schedules, enables export to PDF/Excel, facilitates configuration persistence via save/load, and attempts to ensure generated timetables are conflict-free while meeting specified constraints. Version 1.3.0 provides a functional solution for academic timetable management.

7.2 System Performance

The UI is generally responsive during data input and navigation. Timetable generation speed for small to moderate datasets is expected to be quick (within seconds). Performance for very large or highly constrained datasets might be slower due to the heuristic nature of the algorithm, but it is designed to provide a feasible solution in a reasonable timeframe for its target scale. Export and Save/Load operations are efficient. Formal benchmarks were not part of this project phase but are a consideration for future work.

7.3 Strengths of the Application

Key strengths include: a **user-friendly and well-structured GUI**; **comprehensive configuration options** allowing detailed customization; **automated handling of complex constraints**, reducing manual error; **dynamic UI updates** for flexibility in defining academic structures; a significant time-saving **Save/Load configuration feature**; **multiple export options (PDF, Excel)** for broad usability; **clear visual timetable display**; intelligent **preservation of teacher subject selections** during semester modifications; a reasonably modular code structure within the main application class; and a **basic MAC address-based security** layer.

7.4 Limitations of the Application

Current limitations include: the heuristic algorithm (`generateTimetableAlgorithmV2`) may not always produce the globally optimal timetable or a complete solution for extremely constrained scenarios; potential performance degradation with very large institutional datasets; real-time conflict indication during input is limited (though present for teacher subject context

uniqueness); it's a single-user desktop application lacking multi-user or web capabilities; GUI feedback on generation failures could be more detailed for end-users; the MAC address security is basic and not enterprise-grade, with unencrypted configuration files; the system does not include room allocation; support for more advanced or ad-hoc constraints (like teacher time preferences) is not built-in; and working days are currently hardcoded (Monday-Saturday).

CHAPTER 8: CONCLUSION AND FUTURE SCOPE

8.1 Conclusion

ClassMesh (Version 1.3.0) successfully demonstrates the development of an automated timetable generation system using Java and JavaFX. It offers a comprehensive, user-friendly solution for managing academic scheduling. Key features include detailed configuration of time settings, academic structures, and teaching staff; automated, constraint-based timetable generation via the `generateTimetableAlgorithmV2`; clear visual display of the generated schedule; and essential functionalities like saving/loading configurations and exporting timetables to PDF and Excel formats. The project effectively meets its defined objectives by providing a practical tool that can significantly reduce the administrative burden associated with manual timetable creation in small to medium-sized academic institutions. It serves as a strong foundation built upon solid software development principles, GUI design, algorithmic problem-solving, and the integration of third-party libraries for enhanced capabilities.

8.2 Future Enhancements

Potential future enhancements for ClassMesh to further improve its functionality, usability, and applicability include:

1. **Advanced Algorithm & Optimization:** Implementing more sophisticated heuristics (e.g., Genetic Algorithms, Simulated Annealing) or allowing definition of soft constraints for optimization.
2. **Enhanced GUI Feedback for Generation:** Providing more detailed GUI feedback on generation failures or unscheduled classes.
3. **Room Allocation Module:** Adding functionality to define rooms and integrate their allocation into the timetable.
4. **Configurable Working Days:** Allowing UI configuration of working days.
5. **Teacher Preferences and Unavailability:** Incorporating teacher time preferences or unavailability periods into the generation algorithm.
6. **Student Group Management:** Adding features to manage student groups and ensure their timetables are conflict-free, particularly for elective subjects.
7. **Manual Adjustment Interface:** Providing an interface to manually adjust the generated timetable (e.g., drag-and-drop classes) with real-time conflict checking for fine-tuning.

8. **Multi-User and Web-Based Version:** Developing a client-server architecture or a web-based version for collaborative timetable management and wider accessibility.
9. **Enhanced Security:** Implementing more robust user authentication and authorization, and considering encryption for saved configuration files.
10. **Reporting and Analytics:** Generating reports on teacher workloads, room utilization, subject distribution, etc.
11. **Integration with Other Systems:** Allowing import/export of data from/to other Student Information Systems (SIS) or calendar applications.
12. **Localization:** Supporting multiple languages in the user interface.
13. **Improved Help System:** Integrating a more comprehensive help system or tooltips for various configuration options.

REFERENCES

- Oracle Java SE Documentation. (For core Java features)
- Oracle JavaFX Documentation. (For GUI development)
- iText 7 Documentation: *https://itextpdf.com/docs/itext7/maindoc/* (For PDF export)
- Apache POI Project Documentation: *https://poi.apache.org/* (For Excel export)
- Jackson Project (FasterXML) Documentation: *https://github.com/FasterXML/jackson-docs* (For JSON processing)

APPENDICES

Appendix A: Pseudo-code for `generateTimetableAlgorithmV2` Core Loop

```
FUNCTION generateTimetableAlgorithmV2(semesters, teachers, assignments, timeConfig,
days, constraints)
  Initialize timetable_entries_list
  Calculate available_time_slots based on timeConfig
  Prepare subject_to_teacher_map from assignments
  Calculate required_slots_per_subject_context

  Initialize teacher_busy_map[day][slot]
  Initialize scheduled_slots_count[subject_context]
  Initialize recent_subjects_scheduled[day][semester]
  Initialize teacher_daily_load[day][teacher]

  FOR EACH day IN working_days
    Reset daily states (e.g., recent_subjects_map if post-break)
    FOR EACH slot IN available_time_slots FOR day
      FOR EACH semester IN sorted_semesters
        IF recent_subjects_map indicates max_consecutive_slots_for_subject_reached
THEN
          CONTINUE to next subject or semester
        ENDIF

        FOR EACH subject IN semester.getSubjects() (shuffled)
          current_subject_context = (subject, semester)
          IF scheduled_slots_count[current_subject_context] <
required_slots_per_subject_context[current_subject_context] THEN
            available_teacher = FIND_AVAILABLE_TEACHER(current_subject_context, day,
slot,
                                                       teacher_busy_map,
teacher_daily_load,

constraints.max_slots_per_teacher_day)
            IF available_teacher IS NOT NULL THEN
              ADD new TimetableEntry(day, slot, semester, subject,
available_teacher) TO timetable_entries_list
              UPDATE teacher_busy_map for available_teacher, day, slot
              INCREMENT scheduled_slots_count[current_subject_context]
              INCREMENT teacher_daily_load[day][available_teacher]
              UPDATE recent_subjects_scheduled[day][semester] with subject
              BREAK from inner subject loop (slot filled for this semester)
            ENDIF
          ENDIF
        ENDFOR (subject loop)
        IF no subject scheduled for semester in this slot THEN
            UPDATE recent_subjects_scheduled[day][semester] with NULL (to break
continuity)
```

```
        ENDIF
      ENDFOR (semester loop)
    ENDFOR (slot loop)
  ENDFOR (day loop)

  PRINT unmet needs and teacher loads (for diagnostics)
  RETURN timetable_entries_list
ENDFUNCTION
```

Appendix B: ClassMesh Application Workflow

Application Start

`ClassMesh.start(Stage)`

1. Initial UI Setup

Main layout, MenuBar, Tabs created. Initial sections hidden.

`createMenuBar()` `createSchoolInfoSection()` `createTimeConfigurationSection()`
`createSemesterInputSection()` `createTeacherMappingSection()` `createTimetableTab()`

2. User Inputs School/Department Info (Optional)

Text fields for names.

3. User Configures Time Settings

Work/Break times, Slot duration, Max slots/teacher, Max consecutive/subject.

Listeners update `workingHoursLabel`

Clicks "Next -> Define Semesters"

System Action: Validate Time Config

`handleTimeConfigNextClick()` calls `validateAndStoreTimeConfig()`

Time Config Valid?

Yes

4. User Configures Semesters & Subjects

Selects semester count. UI dynamically builds via `buildSemesterSubjectGrid()` & `createAndAddSemesterRowUI()`.

Inputs names, subject counts, subject names & hours.

`adjustHoursWithinSemester()` may trigger.

Clicks "Next -> Define Teachers"

No

Show Error Alert

User corrects Time Settings (Back to Step 3)

System Action: Process Semester/Subject Data

`handleSemesterNextClick()`

Updates `semesterList`, `allSubjectsObservableList`. Shows Teacher Section.

5. User Configures Teachers & Assigns Subjects

Selects teacher count. UI dynamically builds via `populateTeacherMappingGrid()` & `createAndAddTeacherRowUI()`.

Inputs teacher names, assigns subjects using `setupDynamicSubjectCombosForTeacher()`.

`globallySelectedSubjectContexts` ensures unique assignments.

`checkIfReadyToGenerate()` enables "Generate" button.

6. User Clicks "Generate Timetable"

System Action: Handle Generate Click

`handleGenerateClick()`

Re-validates time config: `validateAndStoreTimeConfig()`

Processes teacher data from UI.

Calls Core Algorithm: `generateTimetableAlgorithmV2()`

If successful, displays timetable: `displayTimetableGrid()`

Enables Export buttons.

Timetable Generated Successfully?

Yes

7. View Timetable & Optionally Export

Timetable shown in "Generated Timetable" tab.

User can click "Export to Excel" (`handleExportExcel()`) or "Export to PDF" (`handleExportPdf()`).

No / Partial

Show Info/Warning Alert

User may need to adjust constraints (Back to Steps 3-5).

Other User Actions (Anytime)

File Menu:

`handleSaveConfiguration()`

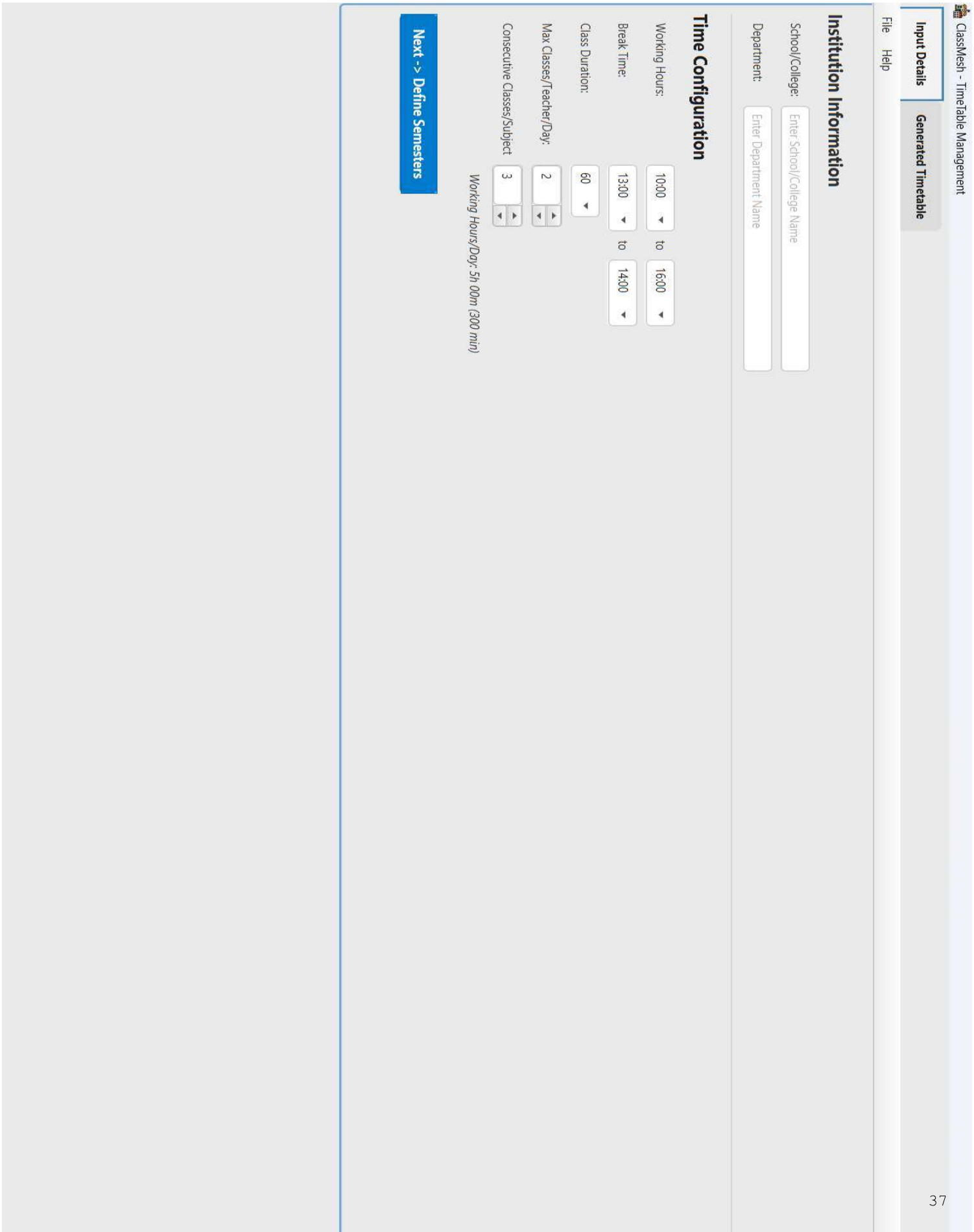
`handleLoadConfiguration()` (resets UI and populates)

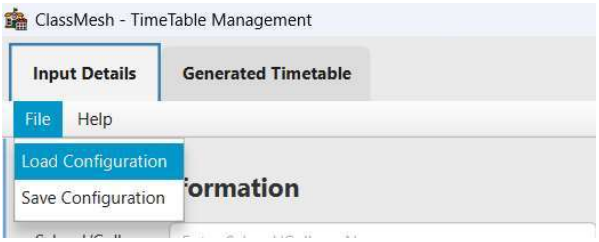
Help Menu:

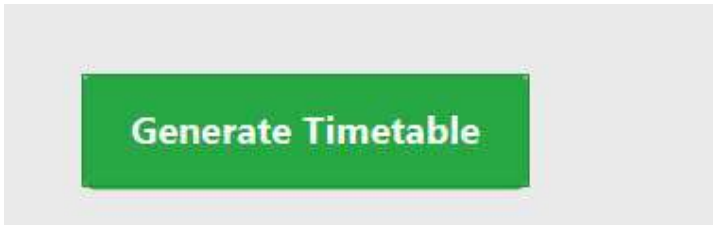
`showAboutDialog()`

Application Interaction Ends / Continues

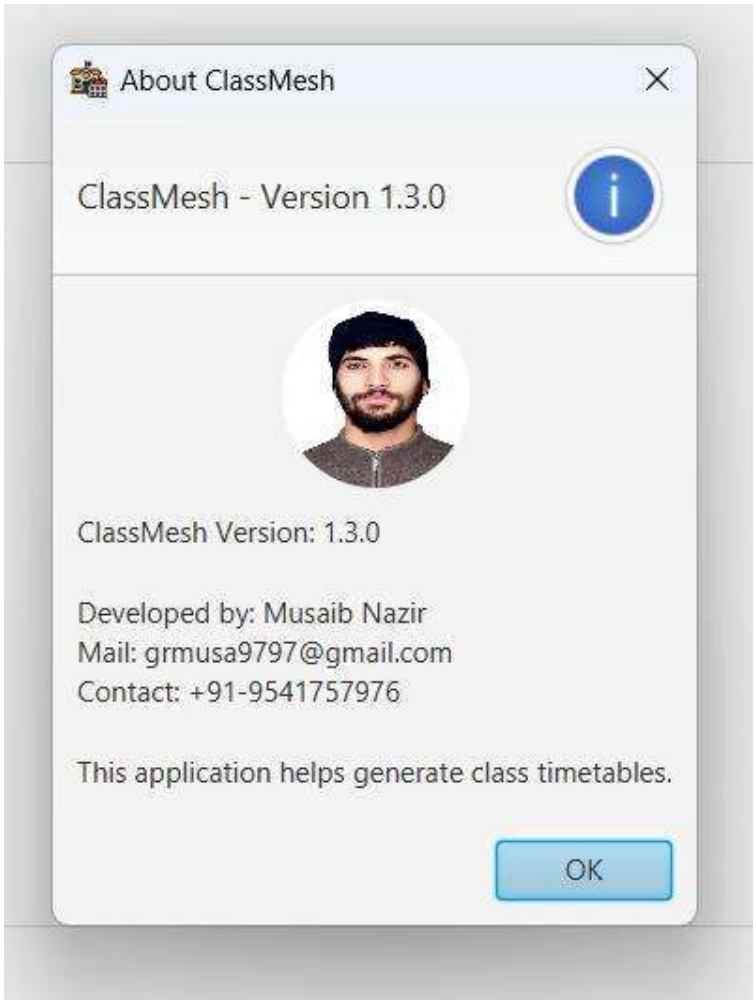
Appendix C: Screenshots of the application




ClassMesh - TimeTable Management
Input Details
Generated Timetable
File
Help
Load Configuration
Save Configuration


Generate Timetable


Input Details
Generated Timetable
File
Help
About ClassMesh
Institution Information


About ClassMesh
ClassMesh - Version 1.3.0
ClassMesh Version: 1.3.0
Developed by: Musaib Nazir
Mail: grmusa9797@gmail.com
Contact: +91-9541757976
This application helps generate class timetables.
OK

ClassMesh - TimeTable Management

Input Details | Generated Timetable

File Help

Working Hours/Day: 5h 00m (300 min)

Next -> Define Semesters

Semester Configuration

3

Semester/Class Names & Subjects:

Sem2	8							
	Applied Mathematics-II	Hrs/Week: 4	Introduction to Computers and Inf	Hrs/Week: 4	Programming in 'C'	Hrs/Week: 5		
	Programming in 'C' Lab	Hrs/Week: 2	Computer Workshop	Hrs/Week: 4	Basics of Electrical and Electronics	Hrs/Week: 5		
	Basics of Electrical and Electronics	Hrs/Week: 2	Communication and Interpersonal	Hrs/Week: 4				
Sem4	10							
	Computer Organization and Archit	Hrs/Week: 3	Object Oriented Programming usir	Hrs/Week: 3	Object Oriented Programming usir	Hrs/Week: 3		
	Internet of Things	Hrs/Week: 3	Internet of Things Lab	Hrs/Week: 2	Full Stack Web Development	Hrs/Week: 2		
	Full Stack Web Development Lab	Hrs/Week: 4	Database Management Systems	Hrs/Week: 3	Database Management Systems La	Hrs/Week: 2		
	Basics of Entrepreneurship Develop	Hrs/Week: 2						
Sem6	5							
	Software Project Management	Hrs/Week: 8	Artificial Intelligence and Neural N	Hrs/Week: 3	Artificial Intelligence and Neural N	Hrs/Week: 2		
	Discrete Mathematics	Hrs/Week: 3	Major Project	Hrs/Week: 14				

Next -> Define Teachers

Teaching Staff Configuration

6

ClassMesh - TimeTable Management

Input Details | Generated Timetable

File Help

Teaching Staff Configuration

6

Teacher - Subject Mapping

	4		
Shakeel Ahmad	Applied Mathematics-II (Sem2)	Artificial Intelligence and Neural Networks (Sem6)	Artificial Intelligence and Neural Networks Lab (Sem6)
	Basics of Electrical and Electronics Engineering (Sem2)		
	2		
Hameem Shah	Basics of Electrical and Electronics Engineering Lab (Sem2)	Basics of Entrepreneurship Development (Sem4)	
	5		
Zubair Manzoor	Communication and Interpersonal Skills (Sem2)	Computer Organization and Architecture (Sem4)	Computer Workshop (Sem2)
	Database Management Systems (Sem4)	Database Management Systems Lab (Sem4)	
	4		
Irfan Rasool	Discrete Mathematics (Sem6)	Full Stack Web Development (Sem4)	Full Stack Web Development Lab (Sem4)
	Internet of Things (Sem4)		
	5		
Jan Mohammad Najar	Internet of Things Lab (Sem4)	Introduction to Computers and Information Technology (Sem2)	Major Project (Sem6)
	Object Oriented Programming using Java (Sem4)	Object Oriented Programming using Java Lab (Sem4)	
	3		
Musaib Nazir	Programming in 'C' (Sem2)	Software Project Management (Sem6)	Programming in 'C' Lab (Sem2)

Generate Timetable

ClassMesh - TimeTable Management

Input Details | Generated Timetable

41

Government Polytechnic College Larkipora Anantnag

Department Of Computer Engineering

Day / Sem	10:00 - 11:00	11:00 - 12:00	12:00 - 13:00	BREAK	14:00 - 15:00	15:00 - 16:00
Monday						
Sem2	**Introduction to Computers and Information Technology** (Jan Mohammad Najar)	**Basics of Electrical and Electronics Engineering** (Shakeel Ahmad)	**Basics of Electrical and Electronics Engineering Lab** (Hameem Shah)	BREAK	**Programming in 'C'** (Musaib Nazir)	**Communication and Interpersonal Skills** (Zubair Manzoor)
Sem4	**Computer Organization and Architecture** (Zubair Manzoor)	**Internet of Things Lab** (Jan Mohammad Najar)	**Full Stack Web Development Lab** (Irfan Rasool)	BREAK	**Object Oriented Programming using Java Lab** (Jan Mohammad Najar)	**Internet of Things Lab** (Jan Mohammad Najar)
Sem6	**Artificial Intelligence and Neural Networks Lab** (Shakeel Ahmad)	**Discrete Mathematics** (Irfan Rasool)	**Software Project Management** (Musaib Nazir)	BREAK	**Discrete Mathematics** (Irfan Rasool)	**Artificial Intelligence and Neural Networks** (Shakeel Ahmad)
Tuesday						
Sem2	**Introduction to Computers and Information Technology** (Jan Mohammad Najar)	**Communication and Interpersonal Skills** (Zubair Manzoor)	**Communication and Interpersonal Skills** (Zubair Manzoor)	BREAK	**Basics of Electrical and Electronics Engineering Lab** (Hameem Shah)	**Basics of Electrical and Electronics Engineering** (Shakeel Ahmad)
Sem4	**Computer Organization and Architecture** (Zubair Manzoor)	**Object Oriented Programming using Java Lab** (Jan Mohammad Najar)	**Object Oriented Programming using Java Lab** (Jan Mohammad Najar)	BREAK	**Computer Organization and Architecture** (Zubair Manzoor)	**Database Management Systems Lab** (Zubair Manzoor)
Sem6	**Software Project Management** (Musaib Nazir)	**Discrete Mathematics** (Irfan Rasool)	**Artificial Intelligence and Neural Networks** (Shakeel Ahmad)	BREAK	**Artificial Intelligence and Neural Networks Lab** (Shakeel Ahmad)	**Major Project** (Jan Mohammad Najar)
Wednesday						
Sem2	**Programming in 'C' Lab** (Musaib Nazir)	**Programming in 'C' Lab** (Musaib Nazir)	**Programming in 'C'** (Musaib Nazir)	BREAK	**Communication and Interpersonal Skills** (Zubair Manzoor)	**Applied Mathematics-II** (Shakeel Ahmad)
Sem4	**Database Management Systems** (Zubair Manzoor)	**Database Management Systems Lab** (Zubair Manzoor)	**Internet of Things** (Irfan Rasool)	BREAK	**Full Stack Web Development Lab** (Irfan Rasool)	**Database Management Systems** (Zubair Manzoor)
Sem6	**Artificial Intelligence and Neural Networks** (Shakeel Ahmad)	**Major Project** (Jan Mohammad Najar)	**Major Project** (Jan Mohammad Najar)	BREAK	**Software Project Management** (Musaib Nazir)	**Software Project Management** (Musaib Nazir)
Thursday						
Sem2	**Applied Mathematics-II** (Shakeel Ahmad)	**Applied Mathematics-II** (Shakeel Ahmad)	**Basics of Electrical and Electronics Engineering** (Shakeel Ahmad)	BREAK	**Computer Workshop** (Zubair Manzoor)	**Applied Mathematics-II** (Shakeel Ahmad)
Sem4	**Full Stack Web Development Lab** (Irfan Rasool)	**Computer Organization and Architecture** (Zubair Manzoor)	**Object Oriented Programming using Java** (Jan Mohammad Najar)	BREAK	**Full Stack Web Development Lab** (Irfan Rasool)	**Computer Organization and Architecture** (Zubair Manzoor)
Sem6	**Software Project Management** (Musaib Nazir)	**Software Project Management** (Musaib Nazir)	**Software Project Management** (Musaib Nazir)	BREAK	**Major Project** (Jan Mohammad Najar)	**Major Project** (Jan Mohammad Najar)
Friday						

Export to Excel | Export to PDF

Application: Exported Outputs

Government Polytechnic College Larkipora Anantnag

Department Of Computer Engineering

Day / Sem	10:00-11:00	11:00-12:00	12:00-13:00	BREAK	14:00-15:00	15:00-16:00
Monday						
Sem2	Introduction to Computers and Information Technology (Jan Mohammad Najar)	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)	Basics of Electrical and Electronics Engineering Lab (Hameem Shah)		Programming in 'C' (Musaib Nazir)	Communication and Interpersonal Skills (Zubair Manzoor)
Sem4	Computer Organization and Architecture (Zubair Manzoor)	Internet of Things Lab (Jan Mohammad Najar)	Full Stack Web Development Lab (Irfan Rasool)		Object Oriented Programming using Java Lab (Jan Mohammad Najar)	Internet of Things Lab (Jan Mohammad Najar)
Sem6	Artificial Intelligence and Neural Networks Lab (Shakeel Ahmad)	Discrete Mathematics (Irfan Rasool)	Software Project Management (Musaib Nazir)		Discrete Mathematics (Irfan Rasool)	Artificial Intelligence and Neural Networks (Shakeel Ahmad)
Tuesday						
Sem2	Introduction to Computers and Information Technology (Jan Mohammad Najar)	Communication and Interpersonal Skills (Zubair Manzoor)	Communication and Interpersonal Skills (Zubair Manzoor)		Basics of Electrical and Electronics Engineering Lab (Hameem Shah)	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)
Sem4	Computer Organization and Architecture (Zubair Manzoor)	Object Oriented Programming using Java Lab (Jan Mohammad Najar)	Object Oriented Programming using Java Lab (Jan Mohammad Najar)		Computer Organization and Architecture (Zubair Manzoor)	Database Management Systems Lab (Zubair Manzoor)
Sem6	Software Project Management (Musaib Nazir)	Discrete Mathematics (Irfan Rasool)	Artificial Intelligence and Neural Networks (Shakeel Ahmad)		Artificial Intelligence and Neural Networks Lab (Shakeel Ahmad)	Major Project (Jan Mohammad Najar)
Wednesday						
Sem2	Programming in 'C' Lab (Musaib Nazir)	Programming in 'C' Lab (Musaib Nazir)	Programming in 'C' (Musaib Nazir)		Communication and Interpersonal Skills (Zubair Manzoor)	Applied Mathematics-II (Shakeel Ahmad)
Sem4	Database Management Systems (Zubair Manzoor)	Database Management Systems Lab (Zubair Manzoor)	Internet of Things (Irfan Rasool)		Full Stack Web Development Lab (Irfan Rasool)	Database Management Systems (Zubair Manzoor)
Sem6	Artificial Intelligence and Neural Networks (Shakeel Ahmad)	Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)		Software Project Management (Musaib Nazir)	Software Project Management (Musaib Nazir)
Thursday						
Sem2	Applied Mathematics-II (Shakeel Ahmad)	Applied Mathematics-II (Shakeel Ahmad)	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)		Computer Workshop (Zubair Manzoor)	Applied Mathematics-II (Shakeel Ahmad)
Sem4	Full Stack Web Development	Computer Organization and	Object Oriented Programming		Full Stack Web Development	Computer Organization and

Day / Sem	10:00- 11:00	11:00- 12:00	12:00- 13:00	BREAK	14:00- 15:00	15:00- 16:00
	Lab (Irfan Rasool)	Architecture (Zubair Manzoor)	using Java (Jan Mohammad Najar)		Lab (Irfan Rasool)	Architecture (Zubair Manzoor)
Sem6	Software Project Management (Musaib Nazir)	Software Project Management (Musaib Nazir)	Software Project Management (Musaib Nazir)		Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)
Friday						
Sem2	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)	Programming in 'C' (Musaib Nazir)	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)		Computer Workshop (Zubair Manzoor)	Programming in 'C' (Musaib Nazir)
Sem4	Basics of Entrepreneurship Development (Hameem Shah)	Internet of Things (Irfan Rasool)	Full Stack Web Development (Irfan Rasool)		Internet of Things (Irfan Rasool)	Full Stack Web Development (Irfan Rasool)
Sem6	Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)		Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)
Saturday						
Sem2	Programming in 'C' (Musaib Nazir)	Introduction to Computers and Information Technology (Jan Mohammad Najar)	Introduction to Computers and Information Technology (Jan Mohammad Najar)		Computer Workshop (Zubair Manzoor)	Computer Workshop (Zubair Manzoor)
Sem4	Basics of Entrepreneurship Development (Hameem Shah)	Computer Organization and Architecture (Zubair Manzoor)	Database Management Systems (Zubair Manzoor)		Object Oriented Programming using Java (Jan Mohammad Najar)	Object Oriented Programming using Java (Jan Mohammad Najar)
Sem6	Major Project (Jan Mohammad Najar)	Software Project Management (Musaib Nazir)				

Government Polytechnic College Larkipora Anantnag

Department Of Computer Engineering

Day / Sem	10:00-11:00	11:00-12:00	12:00-13:00	BREAK	14:00-15:00	15:00-16:00
Monday						
Sem2	Introduction to Computers and Information Technology (Jan Mohammad Najar)	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)	Basics of Electrical and Electronics Engineering Lab (Hameem Shah)	BREAK	Programming in 'C' (Musaib Nazir)	Communication and Interpersonal Skills (Zubair Manzoor)
Sem4	Computer Organization and Architecture (Zubair Manzoor)	Internet of Things Lab (Jan Mohammad Najar)	Full Stack Web Development Lab (Irfan Rasool)	BREAK	Object Oriented Programming using Java Lab (Jan Mohammad Najar)	Internet of Things Lab (Jan Mohammad Najar)
Sem6	Artificial Intelligence and Neural Networks Lab (Shakeel Ahmad)	Discrete Mathematics (Irfan Rasool)	Software Project Management (Musaib Nazir)	BREAK	Discrete Mathematics (Irfan Rasool)	Artificial Intelligence and Neural Networks (Shakeel Ahmad)
Tuesday						
Sem2	Introduction to Computers and Information Technology (Jan Mohammad Najar)	Communication and Interpersonal Skills (Zubair Manzoor)	Communication and Interpersonal Skills (Zubair Manzoor)	BREAK	Basics of Electrical and Electronics Engineering Lab (Hameem Shah)	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)
Sem4	Computer Organization and Architecture (Zubair Manzoor)	Object Oriented Programming using Java Lab (Jan Mohammad Najar)	Object Oriented Programming using Java Lab (Jan Mohammad Najar)	BREAK	Computer Organization and Architecture (Zubair Manzoor)	Database Management Systems Lab (Zubair Manzoor)
Sem6	Software Project Management (Musaib Nazir)	Discrete Mathematics (Irfan Rasool)	Artificial Intelligence and Neural Networks (Shakeel Ahmad)	BREAK	Artificial Intelligence and Neural Networks Lab (Shakeel Ahmad)	Major Project (Jan Mohammad Najar)
Wednesday						
Sem2	Programming in 'C' Lab (Musaib Nazir)	Programming in 'C' Lab (Musaib Nazir)	Programming in 'C' (Musaib Nazir)	BREAK	Communication and Interpersonal Skills (Zubair Manzoor)	Applied Mathematics-II (Shakeel Ahmad)
Sem4	Database Management Systems (Zubair Manzoor)	Database Management Systems Lab (Zubair Manzoor)	Internet of Things (Irfan Rasool)	BREAK	Full Stack Web Development Lab (Irfan Rasool)	Database Management Systems (Zubair Manzoor)
Sem6	Artificial Intelligence and Neural Networks (Shakeel Ahmad)	Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)	BREAK	Software Project Management (Musaib Nazir)	Software Project Management (Musaib Nazir)
Thursday						
Sem2	Applied Mathematics-II (Shakeel Ahmad)	Applied Mathematics-II (Shakeel Ahmad)	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)	BREAK	Computer Workshop (Zubair Manzoor)	Applied Mathematics-II (Shakeel Ahmad)
Sem4	Full Stack Web Development Lab (Irfan Rasool)	Computer Organization and Architecture (Zubair Manzoor)	Object Oriented Programming using Java (Jan Mohammad Najar)	BREAK	Full Stack Web Development Lab (Irfan Rasool)	Computer Organization and Architecture (Zubair Manzoor)
Sem6	Software Project Management (Musaib Nazir)	Software Project Management (Musaib Nazir)	Software Project Management (Musaib Nazir)	BREAK	Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)
Friday						
Sem2	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)	Programming in 'C' (Musaib Nazir)	Basics of Electrical and Electronics Engineering (Shakeel Ahmad)	BREAK	Computer Workshop (Zubair Manzoor)	Programming in 'C' (Musaib Nazir)
Sem4	Basics of Entrepreneurship Development (Hameem Shah)	Internet of Things (Irfan Rasool)	Full Stack Web Development (Irfan Rasool)	BREAK	Internet of Things (Irfan Rasool)	Full Stack Web Development (Irfan Rasool)
Sem6	Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)	BREAK	Major Project (Jan Mohammad Najar)	Major Project (Jan Mohammad Najar)
Saturday						
Sem2	Programming in 'C' (Musaib Nazir)	Introduction to Computers and Information Technology (Jan Mohammad Najar)	Introduction to Computers and Information Technology (Jan Mohammad Najar)	BREAK	Computer Workshop (Zubair Manzoor)	Computer Workshop (Zubair Manzoor)
Sem4	Basics of Entrepreneurship Development (Hameem Shah)	Computer Organization and Architecture (Zubair Manzoor)	Database Management Systems (Zubair Manzoor)	BREAK	Object Oriented Programming using Java (Jan Mohammad Najar)	Object Oriented Programming using Java (Jan Mohammad Najar)
Sem6	Major Project (Jan Mohammad Najar)	Software Project Management (Musaib Nazir)		BREAK		

Algorithm Flow: generateTimetableAlgorithmV2

START: generateTimetableAlgorithmV2

Input: semesters, teachers, assignments, timeConfig, days, constraints

1. Initialization Phase

- Create empty `timetable` list.
- Calculate `slotStartTimes` using `calculateTimeSlots()`.
- Build `subjectToTeacherMap`.
- Calculate `requiredSlotsMap` for each SubjectContext.
- Initialize tracking maps: `teacherBusyMap`, `scheduledSlotsMap`, `recentSubjectsMap`, `teacherDailyLoadMap`.

2. FOR EACH `day` IN `WORKING_DAYS`

Initialize daily tracking for teacher load, busy status, recent subjects.

Set `postBreakResetDoneForDay = false`.

3. FOR EACH `slotStart` IN `slotStartTimes`

Initialize `teacherBusyMap[day][slotStart]` if not exists.

Set `clearRecentHistoryForThisSlot = false`.

4. Is this slot post-break and reset not done?

`( !postBreakResetDoneForDay && breakEnd != null && !slotStart.isBefore(breakEnd) )`

IF YES:

4.1. Perform Post-Break Reset

- Set `clearRecentHistoryForThisSlot = true`.
- Set `postBreakResetDoneForDay = true`.

Log: "Resetting consecutive history."

5. FOR EACH `semester` IN `sortedSemesters`

Get `recentSubjects` list for this day/semester.

IF `clearRecentHistoryForThisSlot`: Clear `recentSubjects`.

Set `slotFilledForThisSem = false`.

6. FOR EACH `subject` IN `semester.getSubjects()` (shuffled)

7. Subject Fully Scheduled?

`( scheduledSlotsMap.getOrDefault(SC, 0) >= requiredSlotsMap.getOrDefault(SC, 0) )`

IF YES: Continue to next Subject

8. Max Consecutive Slots Reached for this Subject?

`( maxConsecutiveSlotsPerSubject > 0 && recentSubjects.size() == maxConsecutive && allMatch )`

IF YES: Continue to next Subject

9. Find Available Teacher for Subject

- Filter teachers from `subjectToTeacherMap` who:
- Are NOT in `busyTeachersThisSlot`.
- AND `dailyLoad.getOrDefault(teacher, 0) < maxSlotsPerTeacherPerDay`.

10. Available Teacher Found?

IF YES:

10.1. Schedule Class

- Create `TimetableEntry`.
- Add teacher to `busyTeachersThisSlot`.
- Increment `scheduledSlotsMap` for SubjectContext.
- Increment `dailyLoad` for teacher.
- Update `recentSubjects` history.
- Set `slotFilledForThisSem = true`.
- **BREAK from Subject Loop (go to next Semester for this slot)**

11. After Subject Loop for Semester

IF `!slotFilledForThisSem`: Update `recentSubjects` with null/break.

12. Post-Processing (Diagnostics)

- Call `printUnmetNeeds()`.
- Call `printTeacherLoad()`.

END: Return `timetable` list