**Industrial Internship Report on**

**” College Timetable Generator”**

Prepared by

[S. Senorin Shafe]

|  |
| --- |
| Executive Summary |
| This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT).  This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 4 weeks’ time.  My project was (“Student Record Management System (Console Based)”)  This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship. |

Contents

[1 Preface 3](#_Toc205399547)

[2 Introduction 4](#_Toc205399548)

[2.1 About UniConverge Technologies Pvt Ltd 4](#_Toc205399549)

[2.2 About upskill Campus (USC) 4](#_Toc205399550)

[2.3 Objectives of this Internship program 4](#_Toc205399551)

[2.4 Reference 5](#_Toc205399552)

[2.5 Glossary 5](#_Toc205399553)

[3 Problem Statement 6](#_Toc205399554)

[4 Existing and Proposed solution 7](#_Toc205399555)

[4.1 Code Submission (github link): 8](#_Toc205399556)

[4.2 Report Submission (Github link): 8](#_Toc205399557)

[5 Proposed Design / Model 9](#_Toc205399558)

[5.1 Interfaces (if applicable) 9](#_Toc205399559)

[6 Performance Test 10](#_Toc205399560)

[6.1 Test Plan / Test Cases 10](#_Toc205399561)

[6.2 Test Procedure 11](#_Toc205399562)

[6.3 Performance Outcome 12](#_Toc205399563)

[7 My Learnings 14](#_Toc205399564)

[8 Future Work Scope 15](#_Toc205399565)

# Preface

Over the course of four weeks, I developed a College Timetable Generator using Python, focusing on interactive user input, randomized subject allocation, and structured data presentation. This project was completed as part of my internship under UniConverge Technologies Pvt. Ltd. in association with Upskill Campus.

The internship provided valuable hands-on exposure beyond the conventional classroom learning environment. The goal of this project was to simplify and automate the timetable creation process in educational institutions — a task that is often time-consuming when done manually. The program allows users to input working days, time slots, and subjects, and then generates a randomized yet evenly distributed timetable.

The development process was well-structured, moving through phases of planning, coding, debugging, testing, and final documentation. This journey significantly enhanced my Python skills, deepened my understanding of logic design and automation, and gave me confidence in managing a complete project independently.

I am grateful to my mentors, peers, and everyone who supported me during this internship. I encourage my juniors to actively participate in such real-time projects and internships, as they are instrumental in bridging the gap between theory and practice.

# Introduction

## About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end etc.

## About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

USC is a career development platform that delivers personalized executive coaching in a more affordable, scalable and measurable way.

The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

## Objectives of this Internship program

The primary objective of this internship project was to design and implement a simple console-based application that automates the generation of a college timetable. The application should accept user inputs for days, time slots, and subjects, and produce a randomized yet structured timetable that avoids repetition and enhances efficiency.

✅ Key Objectives:

✅ To develop a console-based Python application for generating college timetables dynamically.

✅ To implement logic that distributes subjects fairly across time slots without overlaps.

✅ To practice and strengthen Python programming skills, especially in user input handling, randomization, and modular code design.

✅ To reduce the manual workload of staff by automating the timetable creation process.

✅ To understand the importance of user interaction, program flow, and clean code architecture in real-world projects.

✅ To gain hands-on experience in project development, debugging, and documenting software solutions.

✅ To apply the skills learned during the internship and understand how automation tools can solve everyday academic problems.

## Reference

[1] Python Official Documentation

https://docs.python.org/3/

[2] <https://www.w3schools.com/python/>

[3] [https://www.geeksforgeeks.org](https://www.geeksforgeeks.org/)

## Glossary

|  |  |
| --- | --- |
| Terms | Acronym |
| CRUD | Create, Read, Update, Delete |
| Console App | An application that runs via command line |
| File I/O | Reading and writing data to/from a file |
| Python | The programming language used to build this project |
| User input | Data entered by the user through the keyboard |

# Problem Statement

In the assigned problem statement, the objective was to design and implement a console-based College Timetable Generator using Python that could efficiently handle the creation of academic timetables based on user-defined inputs such as working days, time slots, and subjects. The system needed to be simple, interactive, and capable of running on basic systems without the need for complex installations or external databases.

In many educational institutions, especially smaller colleges and departments, the process of timetable generation is still carried out manually using pen and paper or basic spreadsheet tools. This manual approach is time-consuming, prone to errors, and often leads to uneven subject distribution or conflicts in scheduling. As institutions grow and course structures become more complex, generating timetables manually becomes increasingly inefficient.

There is a clear need for a lightweight, offline solution that can assist academic coordinators in creating balanced timetables quickly, without requiring advanced technical knowledge or heavy system resources. Most existing timetable software solutions are either too sophisticated, costly, or dependent on internet access and database backends — making them unsuitable for smaller institutions or individual departments.

This project addresses the problem by providing a Python-based console application that takes user input for key parameters and generates a structured timetable using built-in logic and randomized subject allocation. The system allows users to:

Define the number of working days

Enter custom time slots for each day

List the subjects to be scheduled

Generate a timetable with fairly distributed subject sessions

The application is designed to be intuitive, fast, and educational — serving both as a functional utility and a learning experience for the developer. It showcases how fundamental programming concepts like loops, conditionals, lists, and randomization can be effectively combined to solve a real-world academic problem in a resource-friendly manner.

# Existing and Proposed solution

Existing Solutions:

Traditionally, college and school timetables are prepared manually using paper registers or Excel spreadsheets. Some institutions rely on third-party scheduling software, which is often complex and designed for large-scale use.

These methods, while functional, are not optimized for small academic departments, student developers, or institutions with limited budgets and technical support.

• Limitations:

Manual processes are time-consuming and prone to human error, especially when schedules need frequent changes.

Excel sheets offer limited automation and lack validation or logic, making it difficult to ensure balanced subject distribution.

Professional timetable software can be expensive, require internet access, and often demand training or technical knowledge to use effectively.

• Proposed Solution:

The proposed solution is a console-based Python application that allows users to input working days, lecture time slots, and subject names — then automatically generates a well-distributed timetable using randomization and logic.

This tool:

Runs offline

Requires no external database

Is lightweight, easy to understand, and ideal for beginners or academic projects

• Value Addition:

Simple and user-friendly interface

Completely offline — no dependency on internet or heavy software

Great for learning Python basics like loops, conditionals, functions, and random module

Can be extended in the future to include features like:

Subject-teacher mapping

Avoiding repeated subjects in consecutive slots

GUI with Tkinter or PyQt

Integration with databases for persistence

## Code Submission (github link):

https://github.com/senorinshafe/upskillCampus/blob/main/TimeTable.py

## Report Submission (Github link):

https://github.com/senorinshafe/upskillCampus/blob/main/TimeTableAutomation\_Senorinshafe\_USC\_UCT.pdf

# Proposed Design / Model

The College Timetable Generator is a console-based Python application designed to automate the creation of class schedules. It follows a clean modular structure and is aimed at being beginner-friendly and easy to extend.

The program collects inputs such as number of working days, time slots per day, and subject names.

Each logical component — input collection, timetable generation, and timetable display — is handled by separate functions for clarity and reusability.

The timetable is generated using Python's random module to ensure fair and non-repetitive subject distribution across the week.

The application runs completely offline and requires no database or GUI, making it ideal for small academic institutions or student learning projects.

Output is optionally displayed in a tabular format using the tabulate library (with a fallback plain format if the library isn't available).

## Interfaces (if applicable)

Console Interface:  
The entire interaction happens through a text-based console. Users are prompted to enter:

Working days (e.g., Monday to Friday)

Time slots (e.g., 9:00–10:00, 10:00–11:00)

Subject names (e.g., Maths, Physics, Chemistry)

Function Interface:  
The project is broken down into the following main functions:

get\_user\_input() – Collects all required inputs from the user.

generate\_timetable() – Randomly assigns subjects to time slots.

display\_timetable() – Displays the final timetable in a clean tabular format.

# Performance Test

The College Timetable Generator was tested for performance, correctness, and user experience across different input sizes and usage conditions.

✅ Speed:

The application responds instantly, even with multiple working days and several time slots, ensuring a smooth experience for small to medium scheduling needs.

✅ Accuracy:

The timetable ensures that subjects are evenly and randomly distributed without duplicates in a single day unless needed due to input limitations.

✅ Stability:

The program handles incorrect or empty inputs gracefully. It reprompts users for valid data without crashing or exiting unexpectedly.

✅ Scalability:

Tested with up to 7 working days and 8 time slots per day, the system remained responsive and generated balanced schedules efficiently.

## Test Plan / Test Cases

The College Timetable Generator project was tested thoroughly to validate all key functionalities, including user input handling, subject distribution, timetable generation, and output display. Test cases were designed to include both valid and invalid inputs to ensure robust error handling and correctness of results.

✅ Tested Functionalities:

✅ Input Collection – Accurately accepts number of working days, time slots, and subjects.

✅ Randomized Subject Allocation – Ensures subjects are fairly distributed without excessive repetition.

✅ Timetable Generation – Creates a structured weekly schedule based on inputs.

✅ Output Display – Correctly displays the timetable in tabular or plain format.

✅ Fallback Handling – Handles missing modules (like tabulate) gracefully.

✅ Sample Test Cases:

| Test Case ID | Description | Input | Expected Outcome | Status |
| --- | --- | --- | --- | --- |
| TC01 | Valid input test | 5 days, 5 slots, 6 subjects | Timetable generated with no repeated subjects/day | ✅ |
| TC02 | Fewer subjects than slots | 5 days, 6 slots, 3 subjects | Subjects reused fairly without duplication per day | ✅ |
| TC03 | Invalid numeric input | Enter negative number for slots | Prompts for valid input again | ✅ |
| TC04 | Empty subject or time slot name | Press Enter without typing | Fallback name like “Subject1” or “Slot1” used | ✅ |
| TC05 | tabulate module not installed | Run program without tabulate | Timetable printed in plain format without crashing | ✅ |

## Test Procedure

The testing process for the College Timetable Generator was carried out manually using the command-line terminal. The objective was to verify each feature independently and ensure consistent behavior across different input scenarios.

🧪 Steps Followed:

Launch Program

Run the script TimetableProject.py using:  
python TimetableProject.py

Provide Inputs

Enter number of working days (e.g., 5)

Enter time slots (e.g., 9:00–10:00, 10:00–11:00)

Enter subject names (e.g., Physics, Math, English)

Verify Output

Confirm the timetable displays with correct subject allocation

Check for repeated or missing subjects in each day

Test Edge Cases

Provide invalid input (e.g., empty string, zero, negative number)

Check how the system handles fallback defaults or prompts

Test Tabulate Module (Optional)

Uninstall tabulate and rerun program to test plain text display

Reinstall tabulate to verify grid display format

Repeat Tests

Run the program multiple times with different configurations to ensure consistent, random generation without crash or logic failure

## Performance Outcome

The College Timetable Generator (Console-Based) performed all expected operations efficiently during testing. The system successfully handled user inputs, processed timetable logic, and displayed structured outputs without errors or crashes.

🔑 Key Outcomes:

🧠 Efficient Execution:  
All core operations—collecting inputs, generating the timetable, and displaying results—were completed quickly and without delays.

📄 Data Handling:  
The application managed internal data (lists of days, time slots, and subjects) accurately using in-memory data structures, eliminating the need for external storage.

📋 User Interaction:  
Console prompts were well-structured and intuitive, guiding users step-by-step through the input process.

🔄 Consistency:  
Repeated test runs produced logically correct and evenly distributed timetables, maintaining fairness in subject allocation.

💻 Low Resource Usage:  
As a lightweight console application, it consumed minimal system resources and worked seamlessly on basic hardware.

# My Learnings

Working on the College Timetable Generator project significantly strengthened my Python programming skills and provided valuable hands-on experience with real-world problem-solving.

✅ Key Skills Gained:

✅ Python Fundamentals:  
Deepened understanding of functions, loops, conditionals, and user input handling.

📂 Logic Design & Randomization:  
Applied logic to handle fair and randomized subject distribution across multiple days and slots.

🧠 Modular Programming:  
Structured the code into independent, reusable functions for clarity and maintainability.

🧪 Debugging and Testing:  
Practiced testing the application with a variety of inputs and refined the program based on observed behavior.

🖥️ Real-World Application Development:  
Built a practical, offline utility that addresses a common academic need in institutions.

📋 Project Structuring:  
Learned how to write professional project reports, organize code, and document functionality clearly.

# Future Work Scope

The current version of the College Timetable Generator was developed as a simple, console-based Python application due to time constraints and the scope of the internship. However, there are multiple directions in which the project can be expanded and enhanced in the future:

🚀 Potential Enhancements:

🔐 User Authentication:  
Add a login system for different user roles such as administrators, teachers, and students to control and personalize access.

🗃️ Database Integration:  
Replace in-memory data structures with SQLite or MySQL to store subjects, timetables, and user profiles persistently and securely.

🖥️ GUI Interface:  
Build a graphical user interface using Tkinter, PyQt, or Kivy to improve user interaction and make the tool more accessible to non-technical users.

🌐 Web Version:  
Upgrade the system into a full-fledged web application using frameworks like Flask or Django, allowing institutions to generate timetables online and access them remotely.

📅 Teacher/Room Allocation:  
Introduce features for assigning teachers and rooms to each time slot, along with conflict resolution (e.g., avoiding overlapping classes for the same teacher).

📊 Export & Print Options:  
Enable timetable export to PDF, Excel, or CSV formats, along with options to print or email schedules.

🧠 Smart Scheduling:  
Incorporate advanced logic or AI techniques to optimize the timetable for factors like teacher availability, preferred time slots, and subject difficulty levels.