Software Requirements Document

CS308: Large Applications Practicum
Indian Institute of Technology Mandi

October-November 2019

Group 9:

B17096 Namrata Malkhani
B17031 Aashima
B17033 Akhil Rajput
B17064 Suraj Kumar

Time-Table Assist Tool
Requirements Document

Revision History

Version	Date	Author(s)	Description
V1.0	15/10/19	Aashima	Designed and implemented main DB, created a basic GUI for constraint checking
V2.0	4/11/19	Akhil Rajput	Added add, delete, new time-table, generate and reload functionality in the main GUI, created GUI for DB modification and handled electives and core courses constraints.

Table of Contents

1 Introduction

Time-Table Assist Tool is a user interface designed to help users avoid making errors while making time-table. It does not make the time table, it just assists the user by flashing errors and warnings. The user interface is connected to a database. The database stores all the information required to evaluate all sort of clashes. Each course is to be allotted a slot in the time table, a classroom and instructor(s). The entire decision of allotment lies with the user but the interface warns him in case of clashes. Therefore, the user has full freedom while making the time-table but at the same time he is well aware of where things might go wrong.

1.1 Need and Purpose

A correct time-table is very crucial for better functioning of the institute, faculty and students. Resources like time slots, classrooms and faculty are limited, hence, they should be allocated efficiently. There are additional constraints like transportation etc which can be better understood by humans than the program. With so many requirements and a few ambiguous ones, it is very difficult to design a program which constructs the time-table from scratch. Therefore, time-table is designed manually.

Identifying clashes can be a very daunting task for a human when there are more than 150 courses in a semester. Manual preparation solves the problem of designing the hard algorithm but it increases the chances or errors. Moreover, the person who is given the responsibility to create time-table for institute, has to work under a very strict deadline. Therefore, errors are bound to incorporate. The Time-Table Assist Tool solves this problem. Identifying clashes is easier for a program than a human. It can check the constraints on every step and can warn the user if a clash occurs. So, the user can know if he is making a mistake and also he can know why is he making a mistake.

1.2 Intended Audience

This document is intended for anyone who knows how to use a computer. The user only has to interact to the GUI which is very simple to understand and requires no additional technical or software knowledge. He can just follow the user manual for installation and to know what all can be done using the software. It gives freedom to the user and makes time-table designing process easier and error-free.

2 Description

The Time-Table assist tool has a user interface for allocating each course a particular slot (and a classroom) without any clashing. By clashing, we mean that at the same time no resource (classroom or professors) is being used more than once. Even two courses can clash if they are assigned the same slot and they belong to the same elective basket or are core to same batch of students.

There is a main database that contains all the information of courses, respective instructor(s), classrooms available and elective baskets. The slot-wise course distribution tables are also

stored in the main database which are modified in runtime. At each step, the interface queries the main database to evaluate constraints.
The tool makes sure that a course is added in the database only once and no aforementioned clashing happens. If the allocation does not cause any clash, it is inserted in the runtime database else a warning message is displayed along with the reason for the warning. The user has the option to ignore the warning if he wants and continue knowing what might go wrong.

2.1 Features and Functions

2.1.1 Features

User can use the time-table assist tool on his/her PC. It works offline so it does not need internet connection. The user needs to run a python script which opens up a user interface and all the work is done on the interface.

2.1.2 Functions

In the main GUI, the user can do the following:

1. add new courses to different slots
2. delete previously added courses
3. save current progress
4. restore last saved work
5. generate the time-table
6. modify existing database
7. reload the database

So, if a user wants to change the existing database, he can do it using our GUI.
The database modification GUI has the following functions:

1. select table to modify
2. search for a record in that table
3. add a record to that table
4. modify a record in that table
5. delete a record from that table

The user can perform each of these operations without any software/technical knowledge. All he needs to do is to follow the user manual.

2.2 Users

Time- Table Assist tool can be used by anyone, whether he is from CS background is not a requirement. We have made this software keeping in mind that it can be used by anyone from any background. Anyone who knows how to operate a computer and has followed the user manual, will find it easy to use.

2.3 Operating Environment

2.3.1 Hardware

The Time-Table Assist Tool works on your personal computer. It is a small program, so small RAM will not be an issue. The processing power required by the program is very minimal. It can work on most of the modern day computers and laptops.

2.3.2 Software

The Tool can run on Linux and Windows operating systems. The only requirements are:

1. Python
2. Sqlite3
3. Tkinter

If you have any of the recent versions of aforementioned modules/languages, the software will work fine for you.

3 Specific Requirements

3.1 Design Constraints

We created this software with a rough estimate of how many maximum courses can be added to a single slot. Also, we assumed that there cannot be more than 8 slots in a time-table as it has been eight since the very beginning. So, in case if the number of courses increases drastically or number of slots increase in the coming years, the software would require some minor adjustments.

4 Mini-Project Plan

Version 1

- Designed and implemented the main database. Extracted data from files, arranged it and imported it to the main database.
- Designed a basic GUI which takes input from the user, flashes error if a clash else insert it into the runtime database. Checked all the constraints by querying on the runtime and the main database. The runtime database is visible on the terminal each time the user inputs some data.
- Tested on few entries.
- The First version fulfilled all the needs for adding new allocations.

Version 2 (final version)

- Designed and implemented the main GUI for slot-wise course division. There are separate input fields for add and delete feature. The time table is displayed as a 8*20 matrix. This task was primarily done by *Akhil Rajput.*
- Implemented Add and Check Constraint functionality. The user fills input fields according to the allocation he/she wishes to make. By Check Constraint, the user is only shown warnings but the allocation can still be made using add, irrespective of the various clashes - Classroom, instructor, core and basket. Invalid inputs also need to be validated.
  The Clashes need to be checked by creating runtime tables and querying them. Anomalies need to be taken care of upon each addition and deletion. They are not directly seen. This task was done by *Namrata Malkani.*

Input Validation and displaying warnings was done by *Aashima.*

- Added delete functionality in the main GUI i.e. now we can remove a course assigned to any slot from the GUI. Anomalies need to be taken care of. This task was primarily done by *Akhil Rajput.*
- Designed and implemented a GUI for uploading .csv files required to feed in the database. This task was primarily done by *Suraj Kumar.*
- Added New time-table feature, i.e. we can create a new time-table while scrapping the previous time-table (before scrapping we can generate the time-table and store it in .csv format). Done by *Aashima.*
- Added reload database feature, i.e. we can reload a database and incorporate the changes taken place in the database without closing the application and re-running it. Done by *Aashima.*
- Added generate functionality in the main GUI, i.e. we can generate the time-table we had created, save it in form of a .csv file and then we can use it for creating the real time-table. Done by *Aashima.*
- Handled basket and core courses constraints. Core course constraints are not given directly, they need to fetched and evaluated dynamically from given files. Now, core course clashing and elective courses clashing can also be handled. Done entirely by *Namrata Malkani.*
- Implemented and incorporated a separate GUI for modifying existing records in the database. Now, new records can be added, older ones can be deleted or modified without using SQL directly. This task's responsibility was primarily on *Akhil Rajput.*
- *Namrata* has designed the database. Database creation and manipulation has been largely done by her. For deletion, the manipulation has been by *Akhil.*
- Integrated the aforementioned tasks with the database. This was done by *Aashima.*
- Tested the software with various test cases and on different OS machines.
  OS compatibility has been incorporated by *Namrata Malkani.*
  Testing has been done collectively by *Namrata* and *Aashima.*
