**Analysis of Algorithms**

Spring 2020

**Members Details**

| Group ID | CS311S20PID30 |
| --- | --- |
| Registration Number of Group Members | 2018-CS-22  2018-CS-33 |
| Section | A |

**Project Details**

|  |  |
| --- | --- |
| ***Project*** |  |
| Project Title | Activity Scheduling Tool |
| Executive Summary | Our project is an Activity Scheduling Tool. It is a desktop application that generates the timetable automatically after providing it with certain information. The input given to make timetable will be the list of teachers, classroom in which class will be conducted, contact hour list for each subject in a week, list of courses of a session, and the user has also to tell if it is a lab or theory subject. We are using an “Activity Scheduling” algorithm to make an automatic timetable. Our algorithm will select the activity to be scheduled and will place it in the available slot keeping in the view the subject's contact hours and will manage timetable accordingly. We have an option to choose either a web or desktop application for our project. We made a Desktop application for our project. We worked in .Net Framework with C# (C Sharp) language for the frontend and used SQL server 2014 Management Studio for database and backend. In SQL Server 2014 Management Studio we used DDL (Data Definition Language) and DML (Data Manipulation Language). In the activity scheduling tool, the program generates a timetable depending upon the number of contact hours of the subject. It arranges the subjects such that it takes minimum days of the week and cover all subjects according to its contact hours. It has 5 working days; Monday, Tuesday, Wednesday, Thursday, and Friday. The algorithm checks where slots are available. It also checks the contact hour of the subject and keeping in the view all its dependencies it allocates the lecture slot to the subject. The room will be selected by the user itself. After generating the timetable the user can log in again and can see its generated timetable whenever he/she wants. For this, the user needs to remember the userID he entered while generating the timetable. |
| ***Business Case*** |  |
| Outline the business need for the project | *[This section to contain a clear articulation of the business need in the form of a statement that addresses the problem or opportunity. This statement should be no more than three or four sentences]* |
| End user of the product | *[Clearly identify the real life domain and user that you are going to target]* |
| Motivation for Project | The timetable generator was a different project than the other two that were file compression and plagiarism checker. These both were boring projects than the timetable one in our view. Moreover, in the timetable generator, we just need data from the user and we will present data to the user. It means that there is not much use of file handling.  *[This section to contain a clear statement of motivation which drives you to this project]* |
| Description of the project objective(s) | *[Identify the key objectives of the project]* |
| State the level of impact expected should the project proceed and implications of not proceeding | *[State whether the implementation would have an impact at an operational level and/or strategic level and state the impact(s) in 2-3 lines]* |
| Functional Requirements | *[State list of features/services that you have impemented* |
| ***Benefits*** |  |
| What benefits are expected/ anticipated? | Using this program the user can schedule all the activities in minimum days. It means that maximum activities will be arranged in early weekdays. |
| ***Implementation Details*** |  |
| Link to Github Repository | https://github.com/ayeshatahirme/CS311S20PID30.git |
| Total Number of commits in repository before 8th December 2019 | No commits before 8th December. We started committing after 23rd June. |
| Exact contribution of each member | **2018-CS-22**   * Drew interfaces fro our project on paper in Milestone 1. * Wrote the algorithm by myself. * Wrote correctness and complexity analysis of the algorithm. * Created all the interfaces from the signup page to the timetable generator of the frontend of the project. * Implemented algorithm in C# language. * Created all database tables for storing data from the user table to the generated timetable. * Integrated database with the project. (Made connections with database) * Added all the functionality to enter data, page connections, algorithm implementation except log in and signup in the project. * Stored entered data from user input text boxes to the respective database tables. * Implemented my own created algorithm in the project. * Wrote the whole project configuration document. * Did full report writing. * Pushed most of the files on GitHub repo.   **2018-CS-33**   * Wrote about algorithm selection in Milestone 1 * Wrote pseudoCode and calculated the cost of the algorithm. * Added log in and sign up functionality in project. |
| ***Commits in github repository by each member*** | |
| |  |  | | --- | --- | | **Member Registration No.** | **Total Commits** | | 2018-CS-22 | 80 | | 2018-CS-33 | 11 | |  |  | | |
| **Details of commits** | |
| |  |  |  |  | | --- | --- | --- | --- | | **Sr. No.** | **Details of commit** | **Date** | **Member Reg No.** | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| Have you used built in algorithms or you have implemented yourself? | We have not used any built-in algorithm. Instead, we implemented the algorithm created by ourselves to generate the timetable. The algorithm keeps in check the free slots and arranges lectures accordingly. |
| Formats of input | The user will input data in text form. He will enter the required details for the timetable in the application. There is no file handling in this project. The data will be entered in the respective text boxes and will be stored in database tables. |
| Validations | * The user should only enter positive integer values wherever numbers are to be entered like in userID, contact hour text boxes, etc. * The username once used can not be used again to register in the application because it is a unique value. * The userID once used can not be used again to create a timetable because it is a unique value. * The userID used to enter data must be remembered as the user may need it in the future to view his generated timetable by entering the userID. |
| Format of output | The timetable is generated and is presented in the grid view. The user can view the generated timetable whenever he wants to see using the userID. The output is shown on the design page, generator.cs. After adding the userID, the program matches userID in the database and fetch respective records and displays them in the grid view. So, the output is shown in the grid view. |
| Deployment | No, we have not deployed our project. |
| ***Details of algorithms*** | |
| *In this section, you are required to provided details of algorithms used in the project with the pseudo code, one paragraph description of each algorithm, complexity and correctness.* | |
| ***Interfaces for your project*** | |
| 1. The first page of our application will ask if the user wants to signup or he is already registered. If the user is already registered he will log in to the program and if he is not registered then the user will make his account by signing up and then he can access further functionalities.      1. If the user is not registered he will click the signup button and this page will appear. Here he has to enter some required data that is his first name, last name, username, his email address, and at last the password. The password entered in both text boxes; password and confirm password are matched. If they are not the same then the message is displayed that passwords did not match. And if they are same the user is registered. After signing up the user is good to use the program.      1. If the user is already registered. He will enter his username and correct password to log in to the program. In case of the wrong password, he will be notified that the password he entered is wrong.      1. After logging in/signing up, this page will appear asking the user if he wants to generate a timetable by adding data or if he wants to view the already generated timetable.      1. If the user clicks on the “Add Data” button, he will have to enter userId that is very important as it will be used to access the timetable. The user will enter userID to display a timetable whenever he wants to view the timetable even in the future he can access the generated timetable on this userID. Further, the user will input the code of the course, semester, and room number in which the classes will be held. Moreover, on clicking the “Subjects” button, a page will appear where the user will input subjects and its details.      1. On clicking on the Subject button, this page will appear. Here the user will input the userID so that respective data is stored with this userID. The user will input subject names, contact hours, and either it is a theory or lab subject. Moreover, for recognition, if the subject is of theory or lab, the user must enter L or T for lab and theory with the subject name.      1. At last, the timetable is generated, it shows all necessary data that is the course code, semester, room number, and the generated timetable. There is a logout button that will logout your account and will take you back on login page. Done button means you are all done and want to switch off. You can access this generated timetable whenever you want by clicking on the “Show Timetable” button and entering the userID of the respective timetable. If the userID is valid the timetable will be displayed.     *[Add actual interfaces of your project with description of each control in the UI]* | |
| ***Integration*** | |
| *What type of difficulties were faced by you while integration of UI and algorithms. What was your strategy in this regard.* | |
| ***Change Requests*** | |
| We made only one change that was of our algorithm. We first decided to work in the Activity scheduling algorithm and submitted it in the “decision of algorithm” on its respective deadline. We could not implement our algorithm according to what we chose. Then we decided to change our algorithm to the Genetic Algorithm and tried to understand it. We submitted the pseudo-code of the algorithm but its cost was too much that we had to make many changes in the algorithm and finally we implemented our algorithm. It is the greedy choice algorithm. It has a linear cost. It means it is an efficient one. Its cost is θ(n) and it seems to be a good and efficient one, | |
| ***Testing*** | |
| *In this section, you are required to mention the issues report and solution proposed.* | |
| ***Technology*** |  |
| Programming Language | C Sharp (C#) |
| Platform | Desktop Application |