STUDENT PORTAL

PF Project

Member 1: Maryam Siddiqui (20K-0434)
Member 2: Mehma Waseem (20K-0261)
Member 3: Eisha Fatima (20K-0383)
BSCS-D

Contents

❖ Acknowledgements

We would like to express our sincere gratitude to our teachers Sir Musawar, Miss Rahemeen and Sir Kariz for their invaluable guidance, support and influence regarding this project. They helped us in various ways, answering any query we have and helping us in improving our projects by giving suggestions. We are also grateful to them for their approval to our selected topic for the project, and for all their assistance.

❖ Introduction

In our proposed system, we aimed to design an enhanced student portal. There are three users of the system: student, teacher and admin. This application will facilitate student with an easily accessible portal, providing them additional features. The purpose is to modify the pre-existing portal, so that the student can access their information at one place. It has provided more authority to admin to handle the students/teacher accounts; also, had a feature to view and update academic calendar. It has provided two-way communication between students and teacher to provide interactive environment in the portal.

❖ Background

➢ <u>Research</u>

While researching for topics for our PF project, we pooled in ideas focusing on education and realized that an education portal is the key to providing students with access to all the information they need regarding their profile i.e. their educational institute, involving current marks, attendance statistics, academic documents such as the very necessary admit card. These portals enhance student-centered learning, and are therefore very important. Hence we thoroughly looked out at the layout of the current student portal of our university and evaluated its current problems through careful analysis.

➢ <u>Project selection</u>

We chose this topic in order to design a more efficient and user-friendly student portal as compared to previous ones. We especially looked at the complexity of making a student portal for our project, in order to achieve maximum learning outcomes. The greater efficiency of the portal ensures greater and quicker response from both sides i.e. the teacher and student, which in turn improves the interaction between them. Also, greater accessibility of the admit card provides ease to student to view it in case of the printed version not being available during the paper, so the student can access the admit card anytime by simply opening his account on the student portal.

❖ Problem Analysis

In order to provide an alternative student portal tackling the problems of the previous ones, we introduced new features in our alternative student portal.

This included the:

- addition of a chat box
- introducing more precise feedback
- an academic calendar
- updating contacts information of registered students/teachers
- providing an admit card to students
- ability to delete a student's and teacher's registered account.

➢ **Admin**

The academic calendar feature is added to keep students and teachers updated regarding weeks of examinations, starting/ending dates of the semester, and holidays. Moreover, the update contact information function used in the portal provides ease to admin particularly when it comes to editing the contact details (i.e. email and contact number) instead of inputting all details again in response to requested updated contact details in their registered details. Lastly, deletion of the registered account of the student/teacher enables admin to not only delete the login account of the student but also registration detail file, the file of marks of a student, in the case of a student/teacher leaving the university.

- ## Teacher

  The account for teacher was previously created by admin in the admin block. We have designed our teachers block ensuring account security. For that purpose, we have secured every teachers account with a passcode. Once logged in, teachers are provided with variety of options. As teachers have the authority to grade students too, for this purpose we have added a marks function in teachers block which will take marks from teachers then upload those marks in a file. Then teachers also have an option to view marks of a particular student using **marks_display()** function. The relationship and interaction between student and teacher plays a large role in the trajectory of a student's academic success. For a friendly environment a function of chat box is also added. They can easily interact with their students via this chat box.

- ## Student

  For the pre-existing features, we let a student login to their account which was previously created. The user input was compared to whether it matched the data in the record and if correct, the student redirected to the student main page where they could view 7 different features in their account. This included a homepage where they could view their current student details, the marks section where the student could view the marks that had been previously set by the teachers, a study plan where they could view their courses for whichever semester, academic calendar for viewing once updated by the admin. The two new features in the homepage were the academic document which included the admit card for viewing in the very same portal for easy access, and the chat box so that students could interact with their teachers simply by entering a string of comments. These two features addressed our initial problems in the proposal and made the portal more accessible and interactive. All these features had been made available for students in the student portal using techniques from C language, for an easier user interface and better version of the portal.

❖ Workload Distribution

1. **K200434** (admin block):

The admin block is designed to ensure account security. For that purpose, when the user inputs his login details it is verified through string function **(strcmp())**to compare the inputted detail with the username and password of admin.

If it is verified, the admin views his main page options. This includes options of registering, creating an account, updating contact details, and deleting accounts of teachers and students with the option of an academic calendar that can not only be viewed but also be updated. Once the option is selected, with the help of **case statements** admin is redirected to a specific function. There is a **wrongA()** function , if wrong input is entered anywhere in the admin block or the task is completed of a selected function it will redirect to this function, which will give an option to admin to either continue or exit.To provide admin a convenient work environment, every function is further categorized into two: press **1.for teachers and 2.for students**.

The first option is an academic calendar that redirects to a function named **calender()** which provides an option either to display or to update the calendar, and it has two sub-functions named **displaycalender()** and **updatecalender().** The **displaycalender()** function is to display the file using created by the **cal()** function that uses an array of pointers to store a string of headings and subheading details in the file named **calender.txt**. While **updatecalender()** function updates the calendar according to the number entered for the update (i.e year,fall/spring column).

For students' and teachers' accounts for the portal, the admin has the authority to register them and create their accounts. Thus, the registration and account creation can be performed by selecting the option 2 and 3, redirecting admin to **details()** and **createaccount()** functions respectively. The details for both the function is inputted using structures and stored in a file named after the user(student/teacher) roll number using **file processing and handling** . Also, to have separate files **strcat()** is used to join it with a string to have a discriminating file name (i.e**fac001.tp** for teacher's registered details file and **fac001.Tdetails** for teacher's account details file).

The fourth option, "UPDATE CONTACT DETAILS", will enable admin to update student\teacher registered details that include their **registered email and mobile number.** This aids admin to specifically change only them if requested, instead of adding all the details again to update the requested person profile. This will create a temporary file that will store details (i.e name, section etc of the previous file) and the updated details. Then will remove the old file replacing it with a new file with the same name. Once edited, the admin can view the file immediately through the **updatedfile()**function by passing the ID number/roll number as a parameter that has been entered to update. Also, if that student/teacher is not registered (i.e. file will not exist of their roll no) it will display an error message.

Lastly, the admin is provided an option to delete the account of the student/teacher using function **del_account().** This enables him to delete all the files linking to that entered roll number either for student or teacher ,by redirecting further to sub functions of **studentdel()** for student and **teacherdel()** for teacher depending on admin selected option**.** But if that file is not registered (i.e. fill will not exist) it will display an error message.

2. **K200261** (teacher block):

   A function named **userlogin()** is designed to provide a lock to teachers account. This function will help the program to verify whether the username or password entered to login to the teachers account is correct or not. Once a teacher's account is logged in, the program will pass the user ID of teacher to another function named as **userloginchoice()** which will display the main menu of the teachers block. The program will give the user five options. If the user selects the first option, then the program will call the function **homepageT()** which will display all personal details of the user i.e. the teacher. Moving on to the second option, it will give access to the teacher to upload marks of any student by entering their ID number using the function **marks_display()**. Thirdly, the user is also allowed to view the updated academic calendar and to achieve this aim a function had been previously designed in the admin block, named **displaycalender()** which will display the calendar initially uploaded by the admin. The teacher can also interact with their students using the chat box option which will call the **chatbox()** function, which will first display the chat box and then ask the user to add comments, creating an interactive environment between teachers and students.

3. **K200383** (student block):

The student block is tackled using multiple **functions**. The previously created student accounts are used in case a student wants to login, and then for login the inputted data is compared to the text present in the files using **file handling**. The function **studentlogin()** has the feature of student login. Then in the function **student()** 7 options are given to the student in case of login successful and for these options, different functions are created. **Case statements** are used for the student to get redirected to the respective functions after selecting an option.

The first option is **homepage()** where the student could view their details as the program opened and printed the respective file on the console. The second option is to display the student marks in function **display_marks()**, as the program simply opened and displayed the file created from the teacher's block where the teacher had entered the student's marks. The third option being course feedback in function **feedback()**, the function has the user input the feedback using structures and by simply entering a number (making the feedback concise), and the feedback is stored in a file. For the fourth option i.e. the study plan in function **studyplan()**, we had already created files having the outline of each semester and those files are compared using string functions and then displayed. We created two separate functions of **Bscs()** and **Bsse()** for this purpose. The fifth option was the academic document in function **academicdocument()** in which we created one file and stored name and section on it, and the second file was the semester file. We merged both files into a third file using **file processing** and displayed the third file on the console. In the sixth option which is the academic calendar in function **displaycalendar()**, the program prints the file of the updated academic calendar from the admin block. For the seventh option i.e. the chat box in function **chatbox()**, the student enters any **string** as text and that string was stored in the file, so that the student can interact with the teacher. Lastly, if the student wants to go back to the student menu or exit the program or there is a wrong input, the **wrongS()** function is used with **if-else conditional statements.**

❖ Methodology

➢ Verification

In our code, we enabled the user to login into their account, whether it be the admin, teachers or a student. We verified the user details by comparing whether the new data matched the previously stored data.

➢ System design

For our portal, we used the following techniques in C language:

- Structures of different data types
- File processing and handling
- Arrays for file names and for input of roll numbers
- Array of pointers to store strings
- Loops and nested loops to reduce repetition
- Conditional statements involving if/else and switch statements
- Functions for every task needed in the program
- Libraries used are:
  - stdio.h(for standard input/output functions i.e scanf,printf)
  - string.h(for importing String functions i.e strcpy,strcat)
  - conio.h(for getch() function)
  - stdlib.h(for exit() function)

❖ Implementation

We used Windows 10 for the whole project, along with writing all the code in C. Dev-C++ was the compiler and notepad was for the storage and usage of files involved.

❖ Results

In the end, our program achieved the following results for every type of user:

- **Admin**: Verification for account to login, upload academic calendar, update academic calendar, student/teacher registration, student/teacher login account creation, update contact information of registered student/teacher and authority to deletion of registered accounts.
- **Teacher**: Verification of account to login, view teacher details on homepage, upload marks, view updated calendar and use chat box to communicate with students.
- **Student**: Verification of account to login, view student details on homepage, view updated calendar, fill course feedback, use chat box, view tentative study plan, view academic document i.e. admit card and view marks.

❖ Conclusion

We would like to conclude that it was an immense learning experience while preparing the project. To state on the concluding of the student portal, our idea is that it can be enhanced and modified using simple C language techniques such as functions, structures, filing, strings, conditional statements, arrays and arrays of pointers. Our goal was mainly to create an easier user interface for students, teachers and admin, along with informing students about their academic document and to keep a track of their academic record, assist admins in account creation and updating of details, and for the teachers to update the student record. In the end, our goals were achieved along with having a deeper understanding of C language.