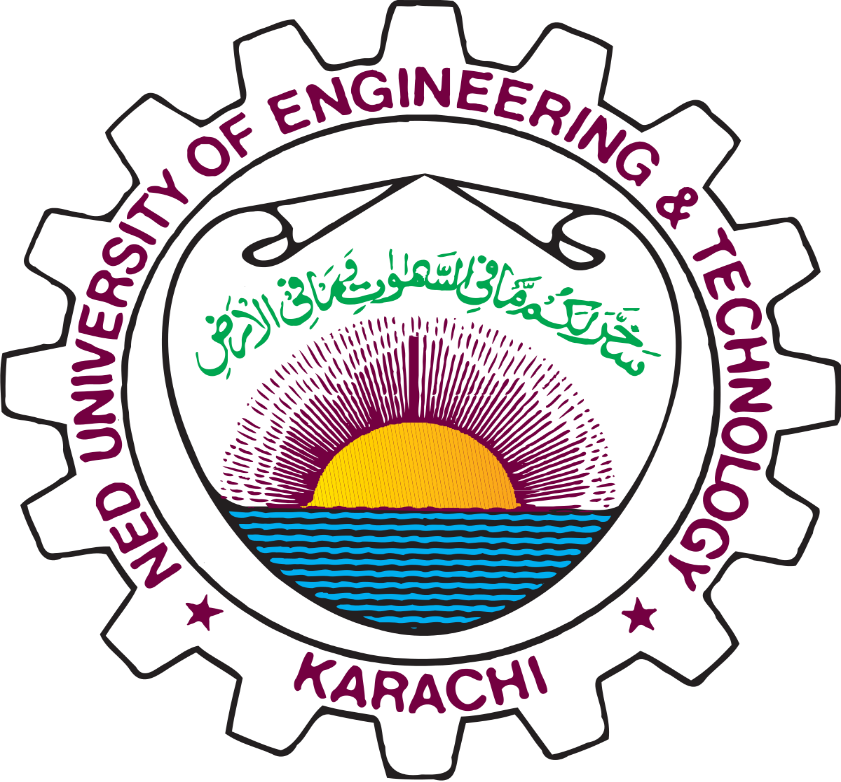
15/08/2018

**OOP PROJECT DOCUMENT**



**GROUP MEMBERS:**

* CT-047 [Shaheer Ghani Imam]
* CT-048 [Muhammad Asad Ali]
* CT-055 [Muhammad Shaheer Akram]
* CT-064 [Abdul Qadir]

**NED PORTAL SYSTEM**

**ABSTRACT:**

This project is a simpler more efficient version of the NEDUET portal giving a more user-friendly interface and functionality. The code provides a user menu for students, teachers and admins.

It is password protected for each user to ensure security of the data. There are exclusive functions for each type of user to limit access to important files.

We used C++ filing to create and store the database on a separate file system and can be accessed by all users.

**MOTIVATION:**

We noticed that the NED online portal had lack of information in some areas or had completely different access sites for certain things that needed to be in the same user interface like sessional marks, course outline or recommended books.

We decided to implement this idea and provide these functionalities to the user.

**SCOPE:**

Although on a smaller scale, this program can be used to understand the fundamentals that are put into use when working on high-end and sensitive data extraction projects.

**OOP CONCEPTS:**

**classes{};** Indeed OOP cannot persist itself entirely without classes; hence, we have used around 7 classes to accomplish our tasks as you can clearly see in the classes prototype sections.

**operator overloading** There are rather a few of the keyword operator in our program mainly because there was not much use of it to complete out task, but a clear example in presented in the date class to overload the insertion operator << to represent date and time as desired.

**abstraction** Data abstraction is one of the most essential and important features of object-oriented programming in C++. In this program we did not access the variables directly which allows the reusability of it and allows code duplication.

**member initializer list** has been used regularly and all throughout the program.

**this->** pointer has solved a great number of issues while the object itself has been passed a reference in the user and data classes. It has also been used for differentiating for the ambiguity when initializing a member variable in any constructor.

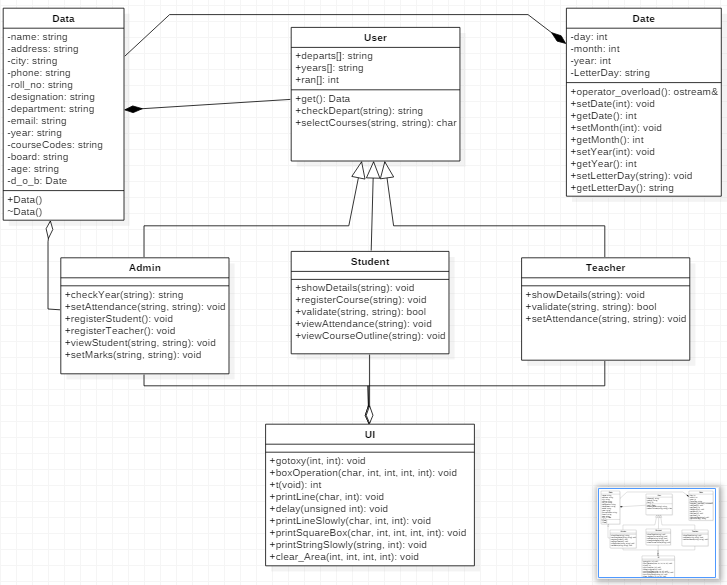
**composition** helped us overpower difficulties where inheritance could not play its part such as in data where date is used as composition

**aggregation** helped us overpower difficulties where inheritance could not play its part such as in Admin, Student and Teacher class where UI is used as aggregation

**inheritance** is the sole purpose of OOP. It has been used regularly throughout the code for example inheriting the user class to admin, student and teacher class. The overuse of inheritance was not appreciated throughout the code instead we used composition and aggregation

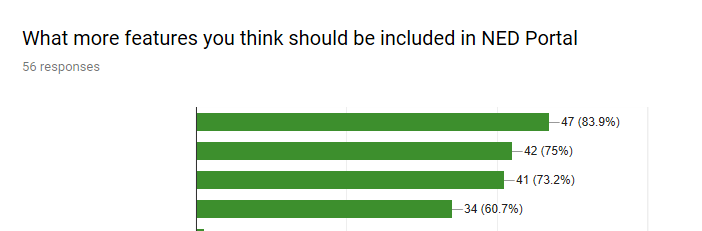
**encapsulation** or data hiding is something every OOP program must; henceforth, encapsulation in performed in every class private hiding and public access in maintained throughout, protected data type is visible in person class. Member and passenger user both have totally different level of access proving the sole purpose and need for data hiding.

**CLASS DIAGRAM:**

****

**SURVEY RESULTS:**

We took a survey from numerous students in the university if they were satisfied with the current NED portal and the pie chart below shows the results we obtained.

****

Seasonal marks display

Books, lectures and other course resources

Examination Results display

CGPA and SGPA display

**CONCLUSION:**

In our project we took a small part of the NED portal and implemented our OOP concepts in it to solve a real life problem around us. This project although not big enough to replace the current portal, still gives an idea on how to improve it.