Technical Documentation

Introduction

The software is built separately as two parts. In the first application, a bitmap file of the barcode of a roll number can be uploaded to read the barcode. This is done by implementing the libbarpp library. EAN 128 barcode version is followed. The time and date is extracted from the system and a binary 1 is stored in the database against the roll number of the student and date extracted. The course is determined using the hour of the time. Alternatively, the roll number, date and time can also be added manually in a new window.

The software is built using MFC Applications on Visual C++. The main form(i.e. login.h) consists of the login portal which is connected to the MySQL database through the .c file. The login leads to 3 different session based on the input in the User Dropdown - Student/Professor/Admin. The student can view his attendance in any one of his courses by selecting it in the courses dropdown and is led to another form which shows his attendance on each of the days in a DataGridViewForm. The Professor can check the attendance of all his students in his course on various days on the same DataGridViewForm. The Admin login session has more functionality allowing him to add/delete users in any of his courses.

Modules

The following modules are present:

Barcode Attendance decryption – There is a module that takes the address of the image and decrypts the roll number stored in it.

- **Take Attendance:** An uploaded picture of bar code is processed and the system date and time is extracted. The database is updated with the attendance. This is implemented as a different application.
- **Student Login Session**: A dropdown is given to chose the course and the submit button opens a new form and shows the records of the student attendance by date by passing an SQL query. The percentage of attendance is also shown.
- **Professor Login Session:** A dropdown is given to chose the course and the submit button opens a new form and shows the records of all the students enrolled by passing an SQL query.
- **Admin Login Session:**
  - View, Add and Delete user records:
  - View course wise attendance: Functionality similar to that in the Professor Login Session is provided.

EXPLANATION OF MODULES

In this project, we used an open source library libbarpp to decode images in .bmp format. We use the function example_decoder_ean128() which takes the file path of the image in a .bmp format and returns the value that is encoded by the barcode

For the Take Attendance module we create a form that allows the user to search for the .bmp image in the computer. The value is passed to the function example_decoder_ean128() which then returns the value encoded in the image.This roll number along with system date and time is connected with the database to update the attendance system.

In this project we use MySQL WorkBench as the backend for database storage . For each request made by the user, the String query is generated which is then passed to the database system which is hosted on the local server . We have used try and catch system to handle any exception possible in the string query. The software is robust and it readily accepts and display proper output or message box depending upon the string query.

We have a file **dbconnect.cpp** This file is base for all database connections and methods.
Constructor establishes the connection to database server. This file contains different functions like search, add , viewall , total which sends query to the sql server and returns the appropriate data.

In some part we have used XML file reader to read the inner text and value of single cells in the database.

Login Page:- In the login page the user selects its type from the combobox and enter its credentials. Then   a proper string query is generated depending upon these three inputs and is passed to Mysql which checks its validation. The same parameters are sent to verifier.cpp which has a function named verifier which checks that whether the input query is valid or not and displays error if its not valid. The corresponding **login.cpp** controls all the processes of the login page. It store the three input strings then depending upon the user type it sends data to **verifier.cpp** and depending upon the value returned it displays the appropriate form for

Student Page(Form1 & Form_viewall):

In this page we try to show the student details in datagridview the whole code is explained properly with the in-line documentation. **Form1.cpp** is the c++ code for the student page.Onload it shows the username .In this file the user gets to select the subject name for which he wants to view the attendance . Each subject is mapped to special number as used in database For example :- “S/W Engg” is mapped to 2_4, etc. After this a function from **Form_viewall.cpp** is called which again decodes the subject names and displays it in the user form . These subject names and uname is used to grab the rollno from login database . Then together with the rollno and subject a new query is made which then shows that particular students row and pseudo user row(for total attendance), after that we calculate the % attendance of that particular student and display it.

Take Attendance (Form_attd):

In this page we at present grab the manual input of rollno , time and date.
Then these data are sent to db.course function in dbconnect.cpp. Where at first according to the date and time , the present subject is selected from “csett” database file Then using this subject name a new query is generated and it is used to update that particular rollno’s attendance in that subject’s table. We also update the pseudo users attendance so that we get to know the total no of classes held.

Professors Page(Form_prof2)

In this page we take give the professor an option to select from the courses.
The Program fetches the data from the corresponding database using **dbconnect.h**.
The **viewall** function is used which does the selection query and presents the result in proper format.
The Attendance of all enrolled students and the pseudo user(that represnts all the classes held are displayed).

Admin Page(Form_admin)

The Admin Portal provides feature of viewing attendance performance by selecting the subject.The functionality is same as the Professor Portal, which demonstrates an aspect of modularity of the Project.The Search button uses the **viewall** function.
The Admin can also view the Records and details of all the Users.
On clicking users the form **Form_users** is loaded which provides features to Add and Delete Users.
The Add User button makes the corresponding part of the part visible and enables it.
On clicking Add a query is run which manipulates the database accordingly.
Similarly the Delete User button makes the corresponding part of the part visible and enables it.On clicking Delete a query is run which deletes the selected users data from the database .
On clicking Records Button the **Form_records** form opens which uses viewall function defined in the database.

Possible Bugs

If the image cannot be decoded, there is a problem with exception handling. The barcode is getting scanned and decoded, but some problems in database connections.

We did a thorough check in the database side and don't expect any error or bug there. At present we have not encrypted the password that could give a chance for extremely professional hacker to hack into the account.

Known Bugs

The Auto Increment column used in each subject's table can cause errors when a student is deleted as it doesn't get decremented.So the give_all and give_none routines don't function in an entirely correct way.
Possible Solution : This bug can be resolved by maintaining another table storing the information about the number of rows in the table of each subject. This information must be updated everytime a new user is added or an existing user is deleted.

Scope for Improvement

1) We can later try to display graph results for students and professors.
2) We can encrypt the password used for Login.
3) We can try to have a roll no or name based search for a particular student.
4) Try to make the database more modular such that the SQL script can be easily generated by writing a simple code of for loop in C/C++.
5) We can work on improving the UI of this version. Proposed new UI
6) Improve the exception handling when dealing with images that can't be handled