

A result management system for Government Schools

Design Project : Desktop Application (CS5201)

Project Report

Submitted by-

Jayant Choudhary

AU18B1003

**Introduction:**

In this course we have to create a desktop application. So creating a desktop application has a three stages first is the planning and initial design, second is the working of application i.e. connection with database, algorithms implementation, event calls, etc and the third stage is about integration, testing and deployment of the application. In stage one I identified the problem which I am going to solve using a desktop application. Then I identified the technology I will be needing and which are best for the problem after that I created a rough sketch for the application. Then I implemented it in pyqt5. This stage was about Database and Algorithm. So first I identified the fields which I will be fetching from the database. Then I identified the relationship between the fields and will the data be shared to others. I found out that I need a relational database which can be shared with other authorities. Then I researched about the different databases. And reached at a conclusion of using MariaDB engine on SQL language.

**Problem Identification:**

**Domain:** Education

**Stakeholders**: Teachers, Government officials related to teaching(DEO, etc.).

**Problem:**

This is digital age, where all the schools should be utilizing the digital tools. But for some reason they are not and the problem has to be faced by the teachers. There are a lot of problems which can be solved using software’s. So one such problem is result making problem. The teachers have to fill the marksheets and other data on multiple registers then they have upload the data on the government’s site then the results is generated. So they have to fill the data multiple times. Where the time is wasted. There is another problem is that the government has to print the registers and sending them to the schools.

**Other problem:**

* There are some government officials who let that happen.
* There are not enough trainings given to the teachers. Nor the teachers nor the instructors are present at the trainings.
* The updates are shared from DEO office through WhatsApp. There is no official medium of communication.

**Reason Behind the problem:**

* There are some government officials who let that happen.
* There are not enough trainings given to the teachers. Nor the teachers not the instructors are present at the trainings.
* The updates are shared from DEO office through WhatsApp. There is no official medium of communication.
* The teachers have to fill the marksheets and other data on multiple registers then they have upload the data on the government’s site and then the results is generated. So they have to do the work triple and the time is wasted there.

**Objective:**

Is to create a desktop application which will have students details and marks associates with them. Where teachers can add the students marks. The result will be calculated based on the marks entered.

**Proposed Solution:**

The records of marks and performance of students will be digitized. There will be a software which will contain the data of students. The data includes scholar number, name, parents name, class, Aadhar number, Samagra Id, etc. And foreach student there will be his performance throughout the year. The performance includes academics, extra-curricular and personal evaluation.

The solution to this problem is a LMS and a CDN for teachers and authorities. The Gov. can directly send circulars to the software for all the teachers to see. The use of WhatsApp will be eliminated. The attendance of the teachers and students should be on the software. The student records, their documents, etc will be on that software. There will be a LMS for teachers where the learning content will be available and they can go through it anytime. The content can be regularly updated by the Gov.

**Scope of the project:**

The scope I kept for the project is the result generation. I will solve the problem using an desktop application. The desktop application will have students details for class 6th, 7th and 8th. The user will be able to add/update the marks of every student. The final grade will be calculated based on the input given by the user. The students data will be stored in a database.

**Flow of user:**

When the user opens the application he will see a table listing students from class 6th. The table will contains details like Scholar number, student name, his father and mother name, title, class, Aadhar number, Samagra Id, etc. Now the user can change the class if he wants to fill the details of some student of other class or he can select the student he wants to fill the result for. Once he clicks on the student a new window will open which will consist of the student details like scholar number, name, fathers name, class, section and save button. Below that there will be a similar format the user is used to see. Then the user can enter the marks of the students per month.  After entering the marks the user will hit the save button. The save button will update the database with the marks and grades. Then the user will close the window. When he hits the close button a dialogue will appear which will confirm if the user really wants to leave the window. If yes is pressed then the dialogue and the performance sheet window will close.  If the user clicks the cancel then the dialogue will close and the user can continue working on the performance sheet window.

**Flow of the user application**:

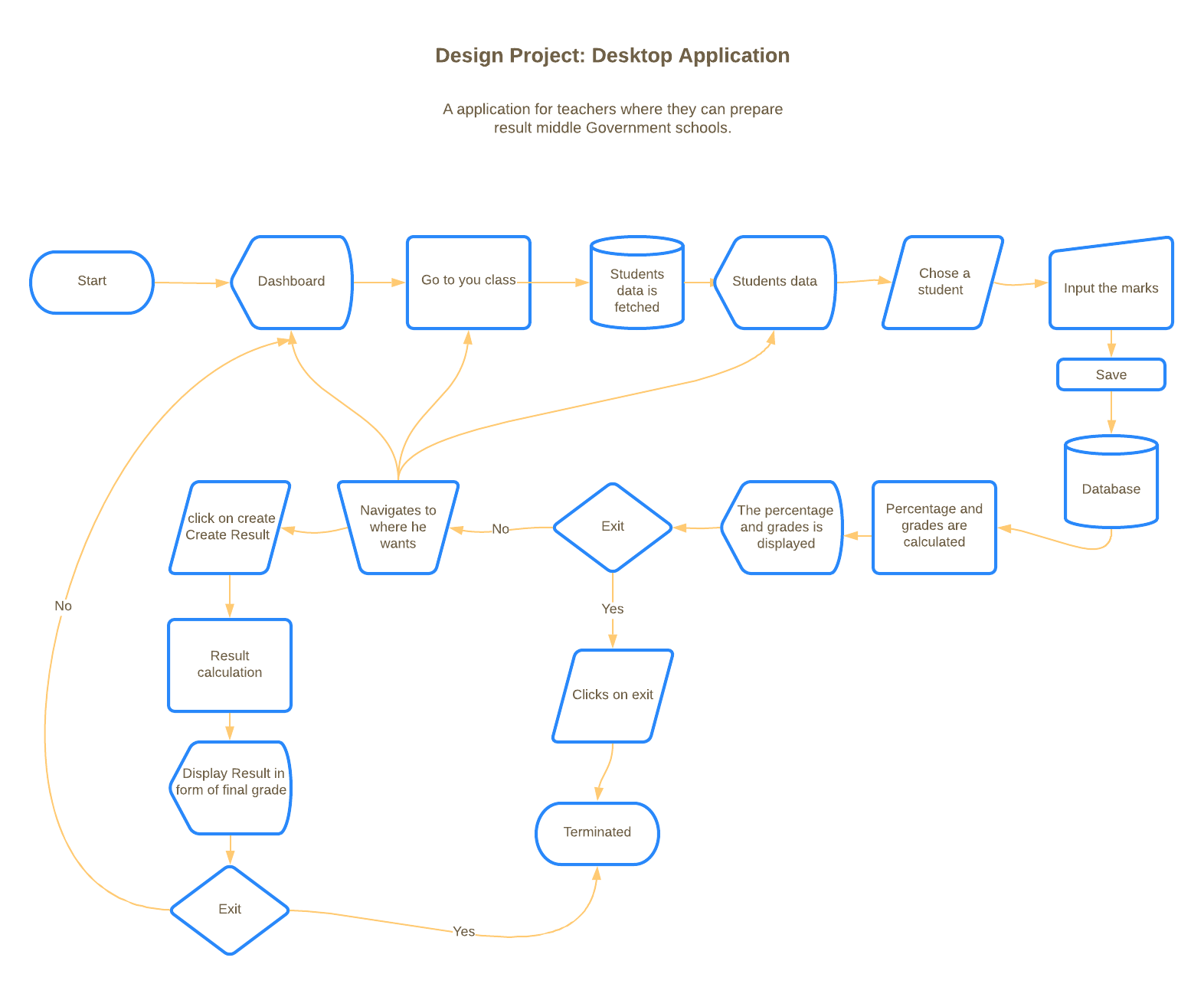


Figure Flow of the user in application with standard symbols

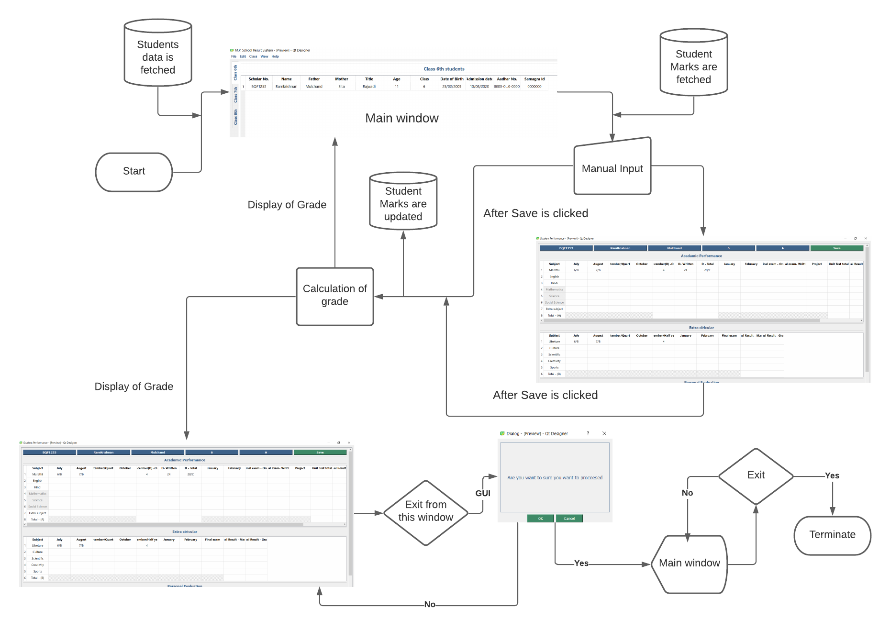


Figure Flow of user in application with GUI elements

**UI design:**

**Framework selection:** I have used Pyqt5 framework/binding for the UI part of the project. I have chosen this framework because this framework is more stable than other frameworks. It also has higher loading speed than other framework. Other than that this framework is widely used by large industries for their application’s UI.

**Tool used:** The tool I used to create The UI is Qt Designer. It generates .ui files which can be converted into python using pyuic module. Pyuic module is created by Qt for conversion of .ui files. It comes with the package PyQt5. PyQt5 is a python library which is used to create GUI using the Qt framework. The Qt framework is written in C++. Qt is a cross-platform application development framework for desktop, embedded and mobile.

**Steps incorporated in UI design:**

Step1: Gathering of information about the fields.

Step2: Ideating and wireframing

Step3: Creating UI.

Step4: Exporting UI files to python files

**Information Gathering:**

I asked the users about the fields, subjects format of that information.

The user provided me with the format they used to fill. The format contains the fields and subjects.

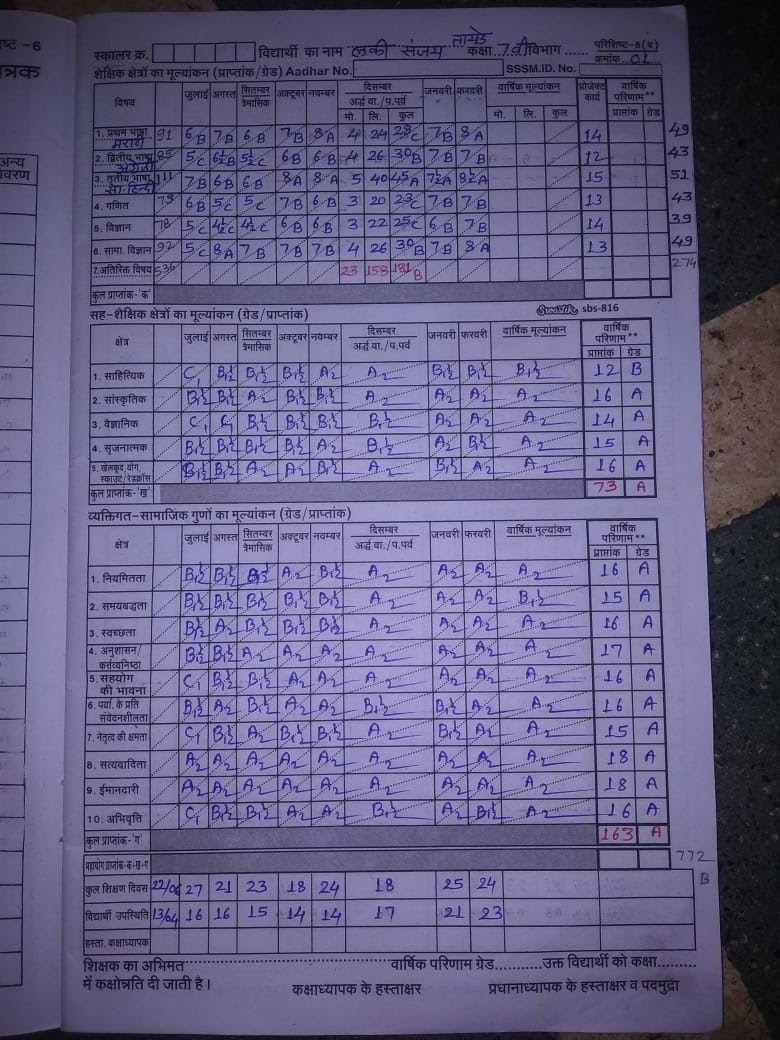
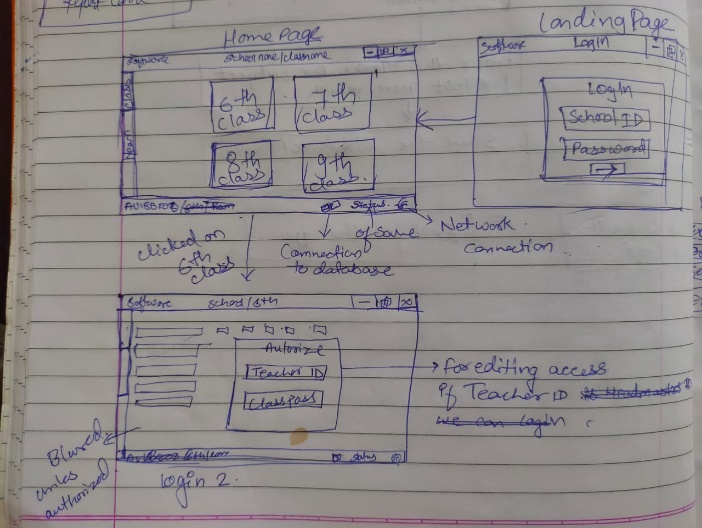
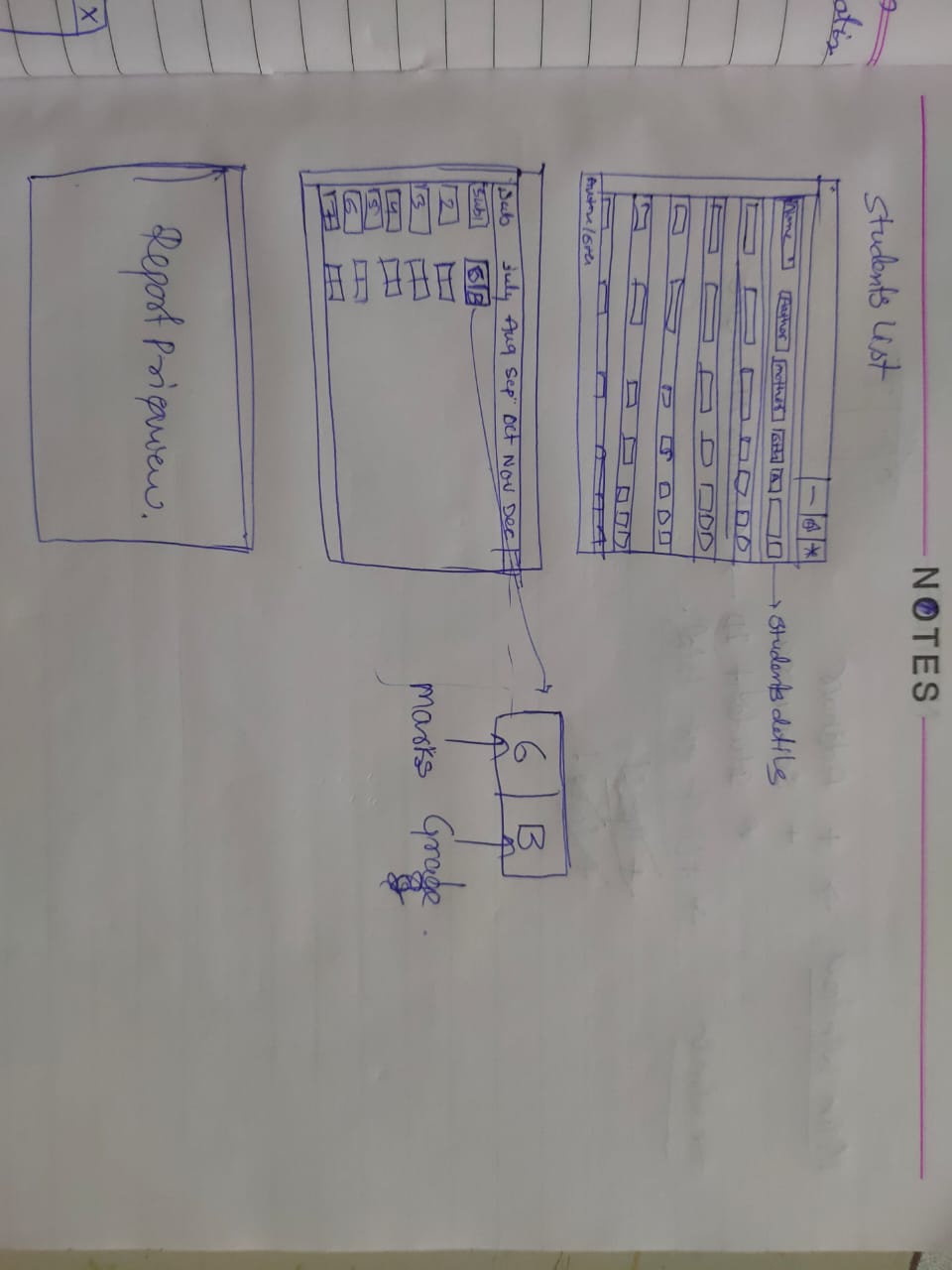


Figure Snapshot of the format used by the user

**Ideating and wireframing:**

For wireframing I started with pen and paper. I designed the screens which can be used in the software.

**Ideating and wireframing**:-

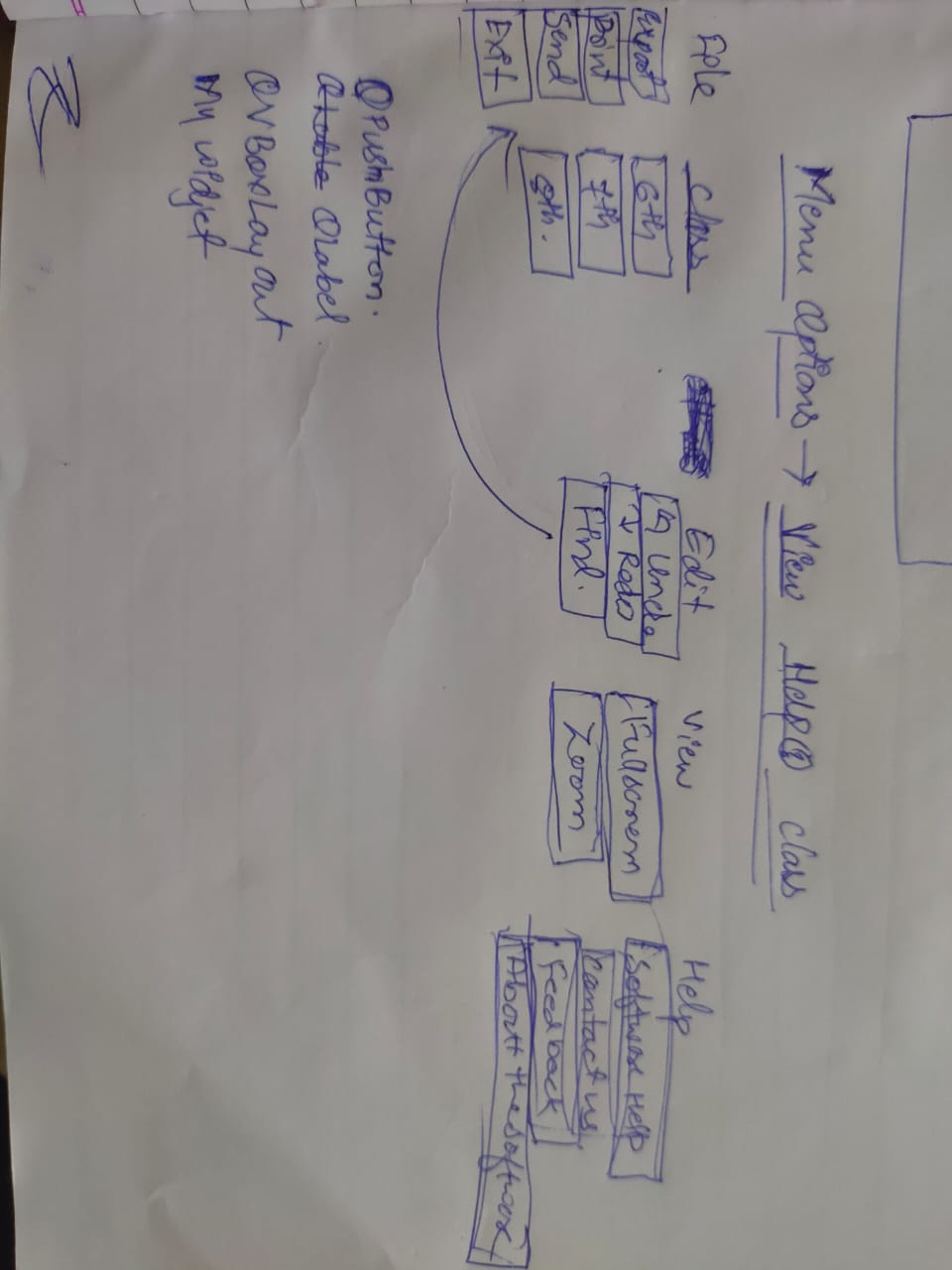
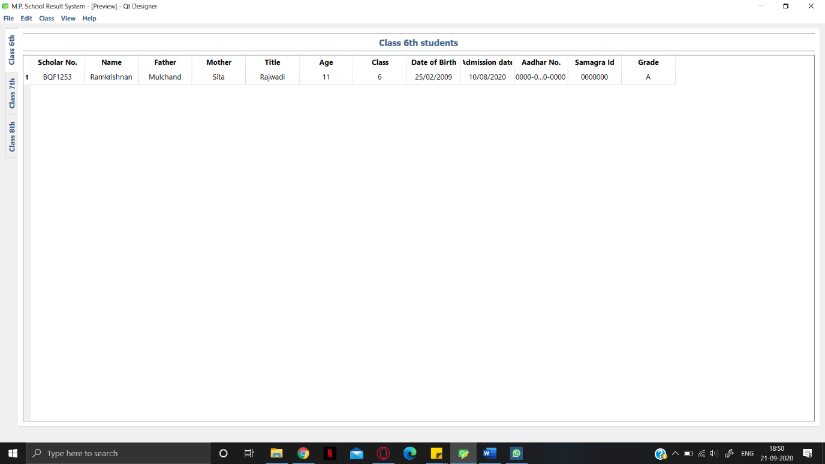
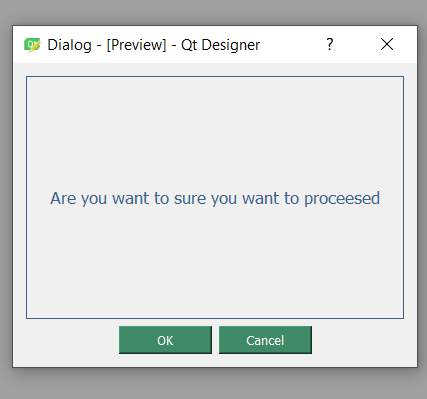
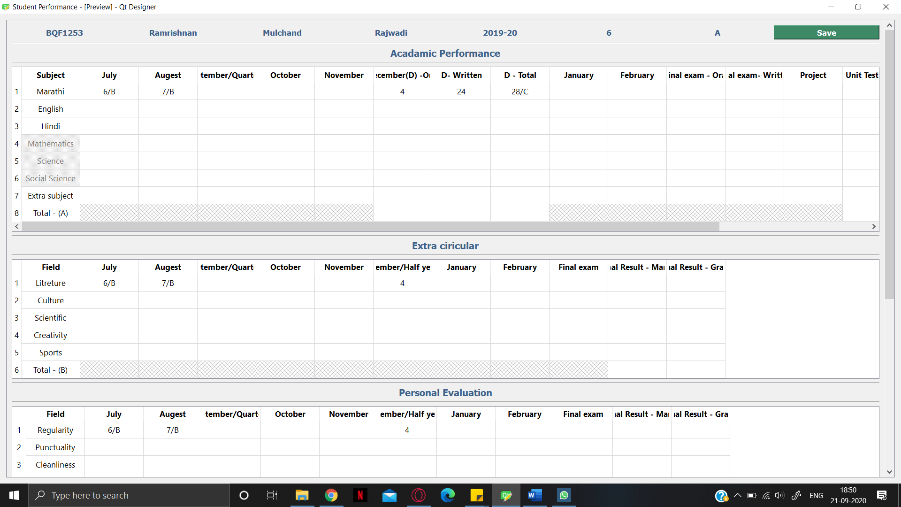


Figure Initial UI concept and ideas

**Implementation in Qt:**



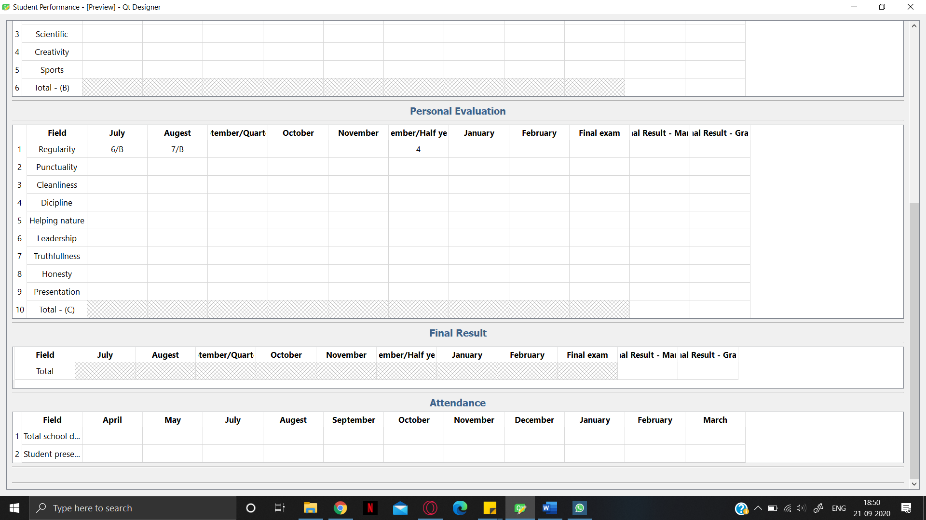


Figure GUI designed using Qt

**Classes and widgets used:**

1. QMainWindow
2. QWidget
3. QTabWidget
4. QLabel
5. QTableWidget
6. QMenubar
7. QMenu
8. QAction
9. QStatusBar
10. QScrollArea
11. QHBoxLayout
12. QPushButton
13. QDialog
14. QDialogButtonBox
15. QMessageBox

Identifiers and lookup table:

|  |  |  |
| --- | --- | --- |
| **Item** |  | **Use** |
|  | Main Window |  |
| Class 6th | This a 1st tab button. | When clicked it the window will show the details of class 6th students. |
| Class 7th | This a 2nd tab button. | When clicked it the window will show the details of class 7th students. |
| Class 8th | This a 3rd tab button. | When clicked it the window will show the details of class 8th students. |
|  | Student Performance |  |
| Save | This is at the top-right. | Uploads the marks to the database |
| Table field | All the fields which are not grey | This fields can be edited with marks. |

**Database Identification:**

For the database creation I needed the fields which the user is going to fill. Which I got at the time of information gathering. So there are 24 fields and each field has 16 - 10 fields which needs to be filled up. Each student is graded based on these 24 fields. All the fields are required and they should be in a structured. The data needs to be shared with the government. There is a need of database sharing i.e. multiple users can access the data from a remote location.

Recruitments for the database:-

* I need relation between the data. - Relational
* I need to share the data to the government authorities. – Multiuser
* Need to store data at multiple locations for the redundancy.
* Low cost
* Secure

So I chose MariaDB because:-

* opensource
* Secure
* Multiple users can login at same time.
* Highly scalable

**Tools Used:**

* I used Xampp for server MySQL server.
* PyCharm as Editor

**Algorithm used:**

I used sorting in select query for displaying students data for different classes and print them Alphabetically.

sql\_select\_Query = "select \* from students where class = 6 ORDER BY name"

I used searching while uploading the data to the database.

"""select \* from marathi where scholar\_no = %s"""

**ER Diagram in UML Notations:**

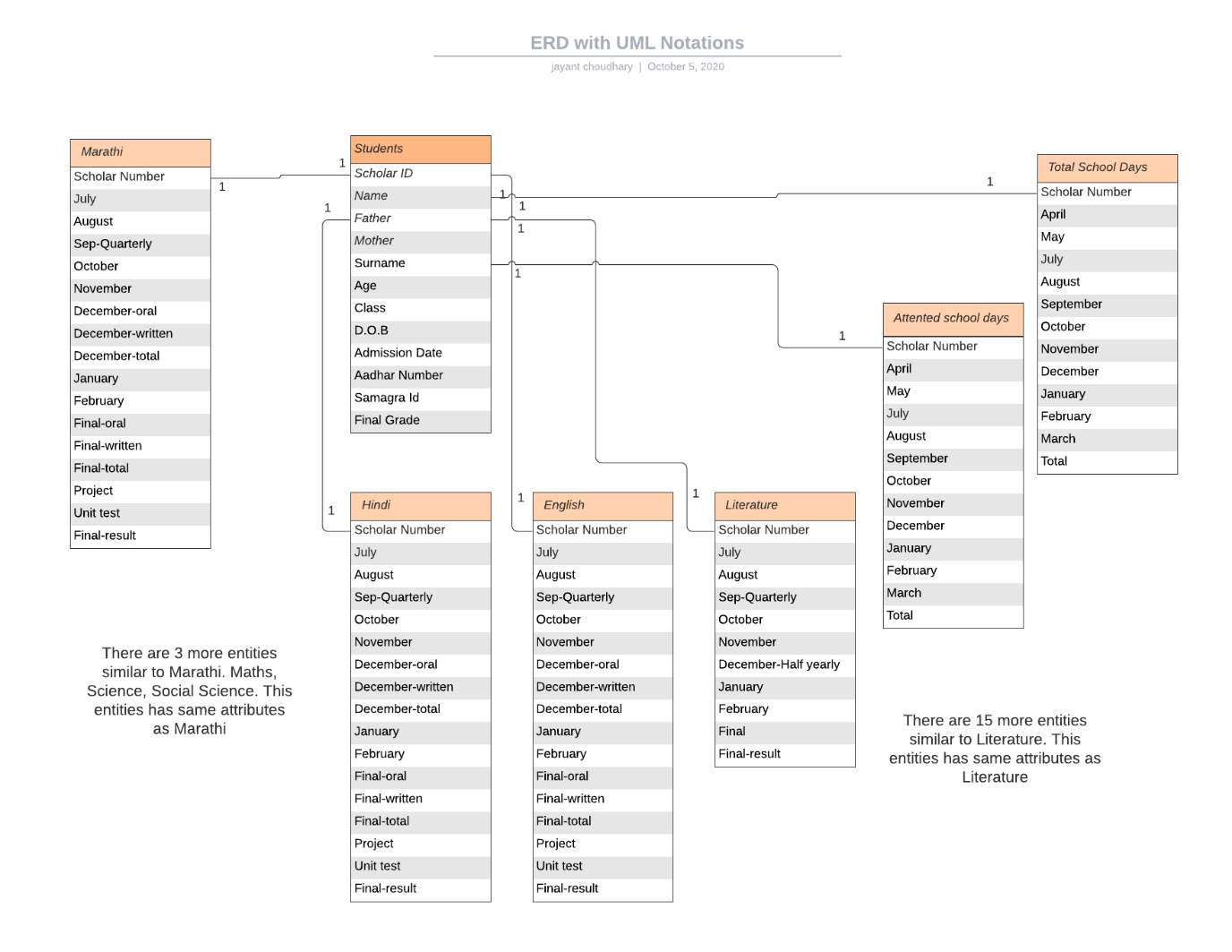
****

Figure ER diagram for the database using UML notations

**Flow of data**:

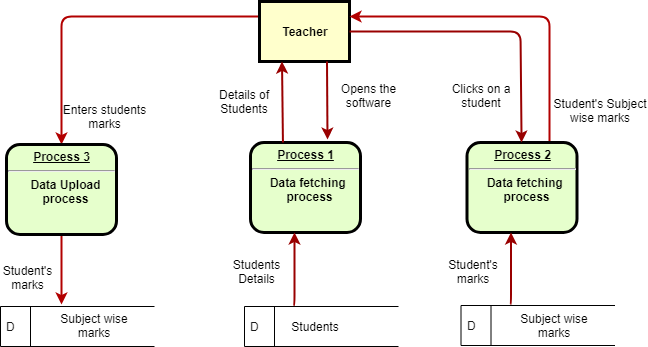
****

Figure Data flow diagram to show the flow of data

**Database Functionality:**

This Code fetches the data from the students table and stores it in dictionary and returns that dictionary.

*import mysql.connector  
from mysql.connector import Error  
  
try:  
 connection = mysql.connector.connect(host='localhost',  
 database='desktop\_app',  
 user='root',  
 password='')  
  
 sql\_select\_Query = "select \* from students where class = 6 ORDER BY name"  
 cursor = connection.cursor()  
 cursor.execute(sql\_select\_Query)  
 records = cursor.fetchall()  
 print("Total number of rows in Students for class 6 is: ", cursor.rowcount)  
 no\_rows = cursor.rowcount  
 students6 = {}  
 for row in records:  
 students6[row[0]] = list(row)  
except Error as e:*

*print("Error reading data from MySQL table", e)*

*finally:*

*if (connection.is\_connected()):*

*connection.close()*

*cursor.close()*

*print("MySQL connection is closed").*

This code fetches, insert and updates the data in the subject(Marathi, English, Maths, etc.) tables.

*import mysql.connector*

*from mysql.connector import Error*

*def fetchStudent(enroll\_performance):*

*print(enroll\_performance)*

*arr = ['Marathi', 'English', 'Hindi', 'Maths', 'Science', 'So\_science','Extra\_sub', 'Literature', 'Culture', 'Scientific', 'Creativity', 'Sports', 'Regularity', 'punctuality', 'Cleanliness', 'Discipline', 'Helping\_nature', 'Leadership', 'Truthfulness', 'Honesty', 'Presentation', 'student\_p\_days']*

*try:*

*connection = mysql.connector.connect(host='localhost',*

*database='desktop\_app',*

*user='root',*

*password='')*

*sql\_select\_Query\_0 = """select \* from marathi where scholar\_no = %s"""*

*sql\_select\_Query\_1= """select \* from english where scholar\_no = %s"""*

*sql\_select\_Query\_2 = """select \* from hindi where scholar\_no = %s"""*

*sql\_select\_Query\_3 = """select \* from maths where scholar\_no = %s"""*

*sql\_select\_Query\_4 = """select \* from science where scholar\_no = %s"""*

*sql\_select\_Query\_5 = """select \* from so\_science where scholar\_no = %s"""*

*arr\_qry=[sql\_select\_Query\_0,*

*sql\_select\_Query\_1,*

*sql\_select\_Query\_2,*

*sql\_select\_Query\_3,…]*

*for x in range(22):*

*cursor = connection.cursor(prepared=True)*

*data\_tuple = (enroll\_performance,)*

*cursor.execute(arr\_qry[x], data\_tuple)*

*records = cursor.fetchall()*

*print("Total number of rows in", arr[x] , "is: ", cursor.rowcount)*

*for row in records:*

*student\_marks[arr[x]] = list(row)*

*cursor = connection.cursor()*

*cursor.execute(sql\_select\_Query\_21)*

*records = cursor.fetchall()*

*print("Total number of rows in t\_school\_days is: ", cursor.rowcount)*

*for row in records:*

*student\_marks['t\_school\_days'] = list(row)*

*print(student\_marks)*

*except Error as e:*

*print("Error reading data from MySQL table", e)*

*finally:*

*if (connection.is\_connected()):*

*connection.close()*

*cursor.close()*

*print("MySQL connection is closed")*

*return student\_marks*

*def updateMarks(enroll\_performance, tup):*

*arr = ['Marathi', 'English', 'Hindi', 'Maths', 'Science', 'So\_science','Extra\_sub', 'Literature', 'Culture',*

*'Scientific', 'Creativity', 'Sports', 'Regularity', 'punctuality', 'Cleanliness', 'Discipline',*

*'Helping\_nature', 'Leadership', 'Truthfulness', 'Honesty', 'Presentation', 'student\_p\_days']*

*print(enroll\_performance)*

*tup\_marathi = tuple(tup[1:17])*

*tup\_english = tuple(tup[18:34])*

*tup\_hindi = tuple(tup[35:51])*

*tup\_maths = tuple(tup[52:68])*

*tup\_science = tuple(tup[69:85])*

*tup\_soscience = tuple(tup[86:102])*

*try:*

*connection = mysql.connector.connect(host='localhost',*

*database='desktop\_app',*

*user='root',*

*password='')*

*sql\_select\_Query\_0 = """select \* from marathi where scholar\_no = %s"""*

*sql\_select\_Query\_1= """select \* from english where scholar\_no = %s"""*

*sql\_select\_Query\_2 = """select \* from hindi where scholar\_no = %s"""*

*……*

*arr\_qry=[sql\_select\_Query\_0,*

*sql\_select\_Query\_1,*

*sql\_select\_Query\_2,*

*sql\_select\_Query\_3,*

*sql\_select\_Query\_4, …]*

*for x in range(22):*

*cursor = connection.cursor(prepared=True)*

*data\_tuple = (enroll\_performance,)*

*cursor.execute(arr\_qry[x], data\_tuple)*

*records = cursor.fetchall()*

*print("Total number of rows in", arr[x] , "is: ", cursor.rowcount)*

*if records:*

*update = """UPDATE"""*

*ending = """SET july= %s, august = %s, sep\_quarterly = %s,october = %s,*

*november=%s, december\_o=%s, december\_w =%s, december\_total =%s, january = %s, february = %s,*

*final\_o= %s, final\_w = %s, final\_total= %s, project= %s, unit\_test= %s, final\_result = %s*

*WHERE scholar\_no = %s"""*

*cursor = connection.cursor(prepared=True)*

*if arr[x] == "Marathi":*

*sub = " marathi "*

*query = update + sub + ending*

*data\_tuple = tup\_marathi +(enroll\_performance,)*

*print("data tuple", data\_tuple)*

*cursor.execute(query, data\_tuple)*

*connection.commit()*

*elif arr[x] == "English":*

*sub = " english "*

*query = update + sub + ending*

*data\_tuple =tup\_english +(enroll\_performance,)*

*print(data\_tuple)*

*cursor.execute(query, data\_tuple)*

*connection.commit()*

*……*

*else:*

*print("Records are empty")*

*insert = """insert into """*

*ending = """(july, august, sep\_quarterly, october, november,*

*december\_o, december\_w, december\_total, january, february, final\_o, final\_w, final\_total,*

*project, unit\_test, final\_result, scholar\_no) values (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,*

*%s, %s, %s, %s)"""*

*if arr[x] == "Marathi":*

*sub = " marathi "*

*sql\_insert\_Query = insert + sub + ending*

*print(sql\_insert\_Query)*

*cursor = connection.cursor(prepared=True)*

*data\_tuple = tup\_marathi + (enroll\_performance,)*

*print(data\_tuple)*

*cursor.execute(sql\_insert\_Query, data\_tuple)*

*connection.commit()*

*elif arr[x] == "English":*

*sub = " english "*

*sql\_insert\_Query = insert + sub + ending*

*print(sql\_insert\_Query)*

*cursor = connection.cursor(prepared=True)*

*data\_tuple = tup\_english + (enroll\_performance,)*

*print(data\_tuple)*

*cursor.execute(sql\_insert\_Query, data\_tuple)*

*connection.commit()*

*……*

*except Error as e:*

*print("Error reading data from MySQL table", e)*

*finally:*

*if (connection.is\_connected()):*

*connection.close()*

*cursor.close()*

*print("MySQL connection is closed")*

**Database Integration in UI:-**

This code uses the dictionary passed from the above functions and displayed the data in table widget. This code is used at all the places where we want to display the data in table widget with changes in variables.

*r\_index = 0*

*for rows in students8:*

*item = self.tableWidget\_2.verticalHeaderItem(r\_index)*

*item.setText(\_translate("MainWindow", str(r\_index+1)))*

*std = students8[rows]*

*for lenth in range(12):*

*details = str(std[lenth])*

*print(r\_index, ", ", lenth, " - ", details)*

*item = self.tableWidget\_3.item(r\_index, lenth)*

*item.setText(\_translate("MainWindow", details))*

*r\_index += 1*

This code is used for displaying the marks.

*mar = fetchStudent(enroll\_performance=enrollId)*

*data\_marathi = mar['Marathi']*

*for iterator in range(15):*

*data = data\_marathi[iterator]*

*data = str(data)*

*item = self.tableWidget\_5.item(0, iterator+1)*

*item.setText(\_translate("Form", data))*

The function gets all the data from the table widgets and stores it in an array and then passes that array into another function which has the upload and insert code written.

*def uploadDetails(self,enrollId):*

*tup = []*

*for y in range(self.tableWidget\_5.rowCount()-1):*

*for x in range(self.tableWidget\_5.columnCount()-1):*

*data = self.tableWidget\_5.item(y, x)*

*if data.text() != "" and len(data.text()) <= 3:*

*tup.append(int(data.text()))*

*else:*

*tup.append(data.text())*

*updateMarks(enroll\_performance=enrollId,tup=tup)*

**Testing**

Testing techniques used-

1. Unit testing- Performed 5 test cases per logic in the code.
2. Integration testing - Integration of the windows and buttons are tested. This testing is also done with the user.
3. Acceptance testing - I gave the software to one of the user for using. It took 20 min for the user to get familiarized with the software. The difficulties the user faced is the speed of the software. It takes a few seconds to load the window. Sometimes the user double clicked.

**Deployment:**

For compilation of the software I used pyinstaller. I compiled the system into a single exe file. After compilation I tested it. It worked successfully. Then for deployment I used advanced installer to create a MSI file for the installation of the software on the machine.

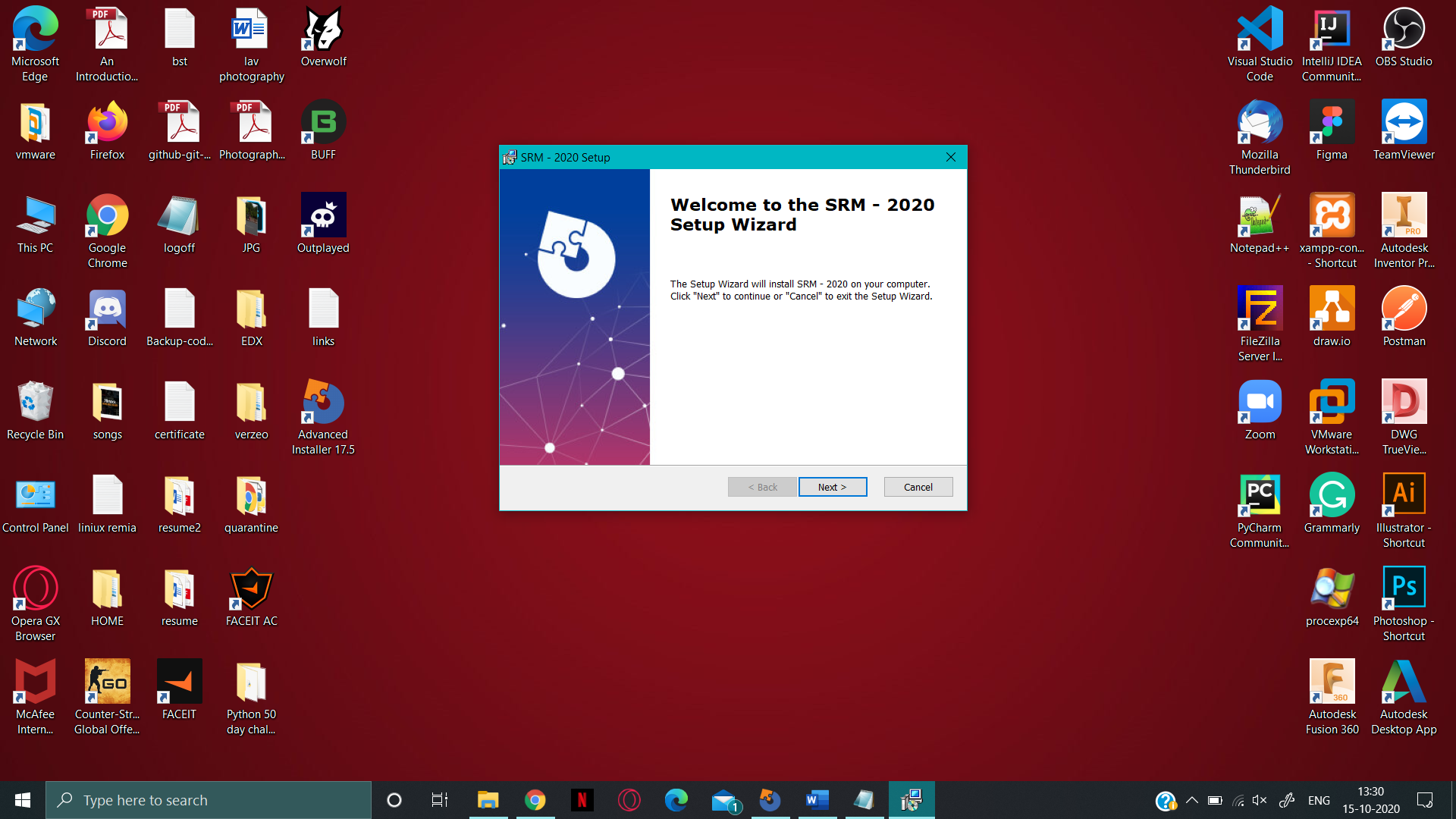
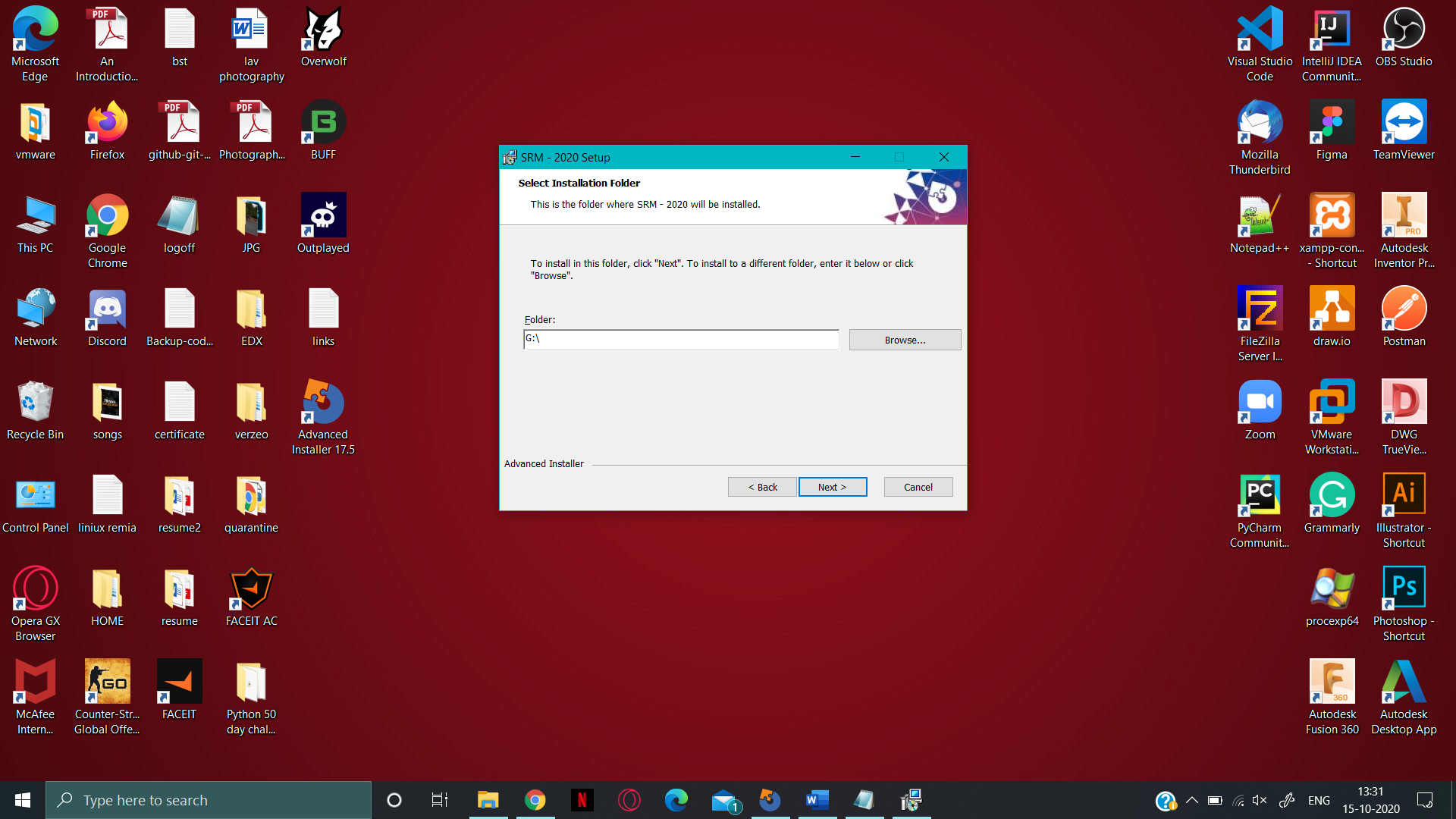
Compilation:-

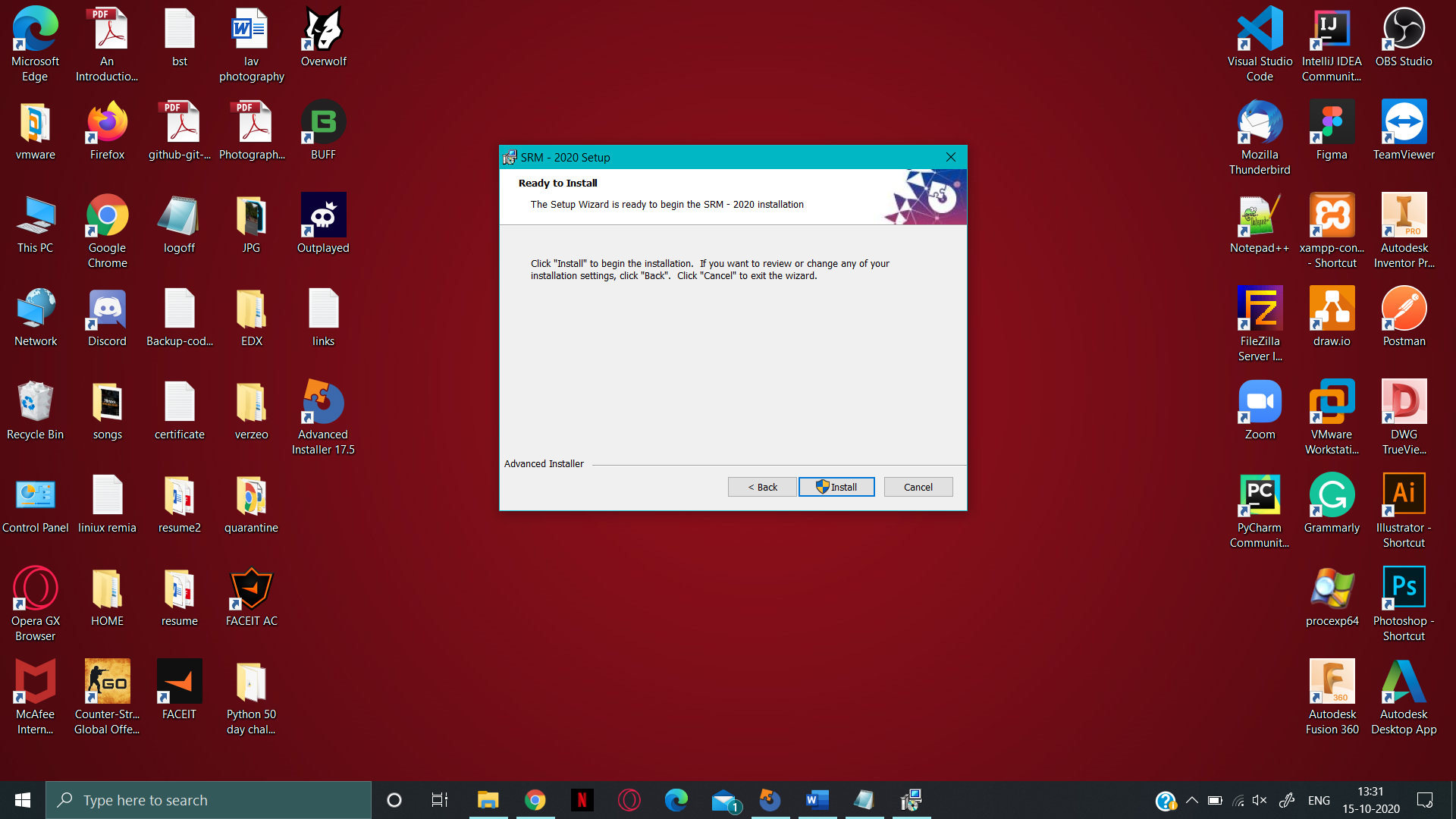
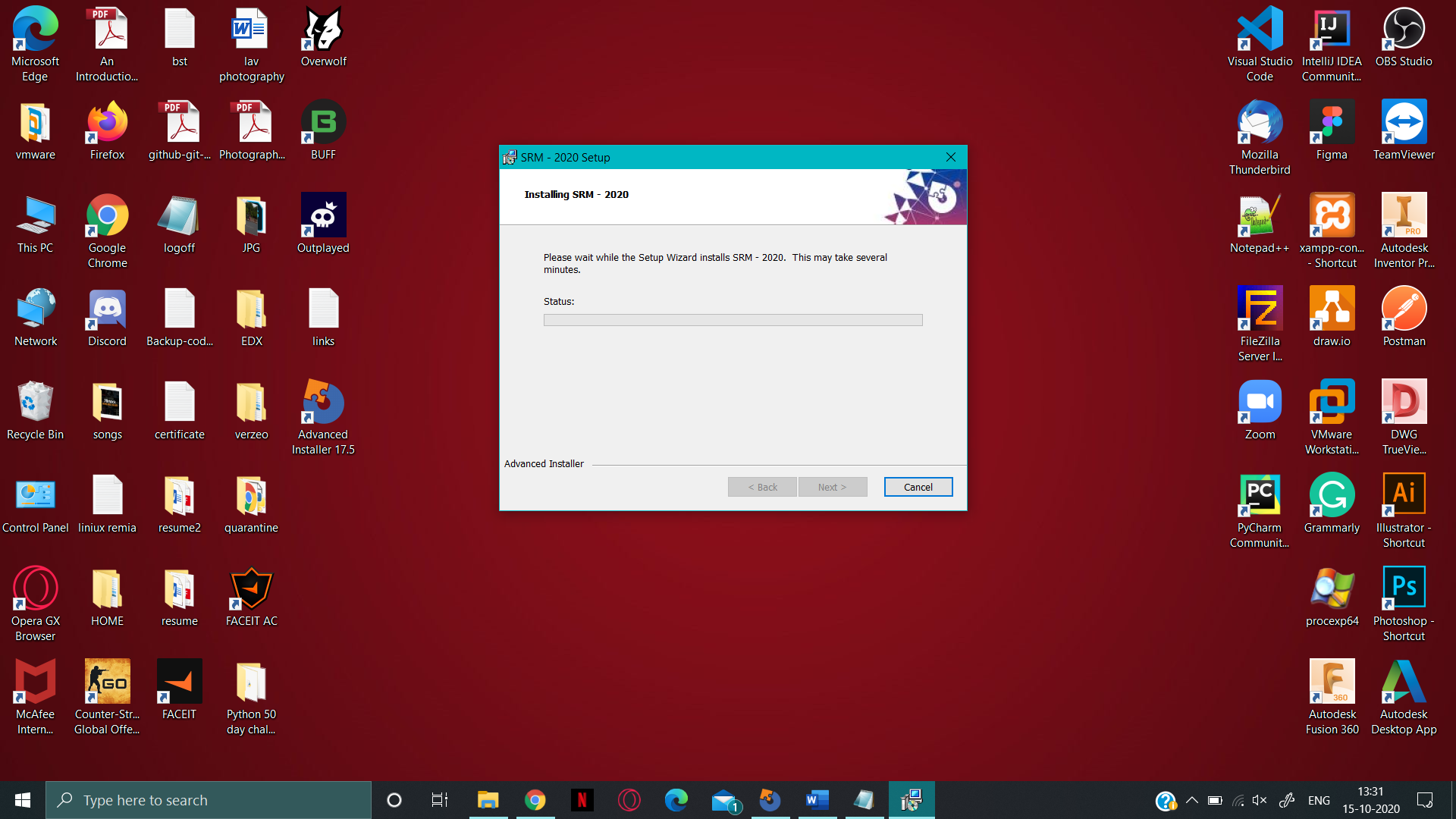
I used pyinstaller for the compilation of the program. This the command I used, to create the executable.

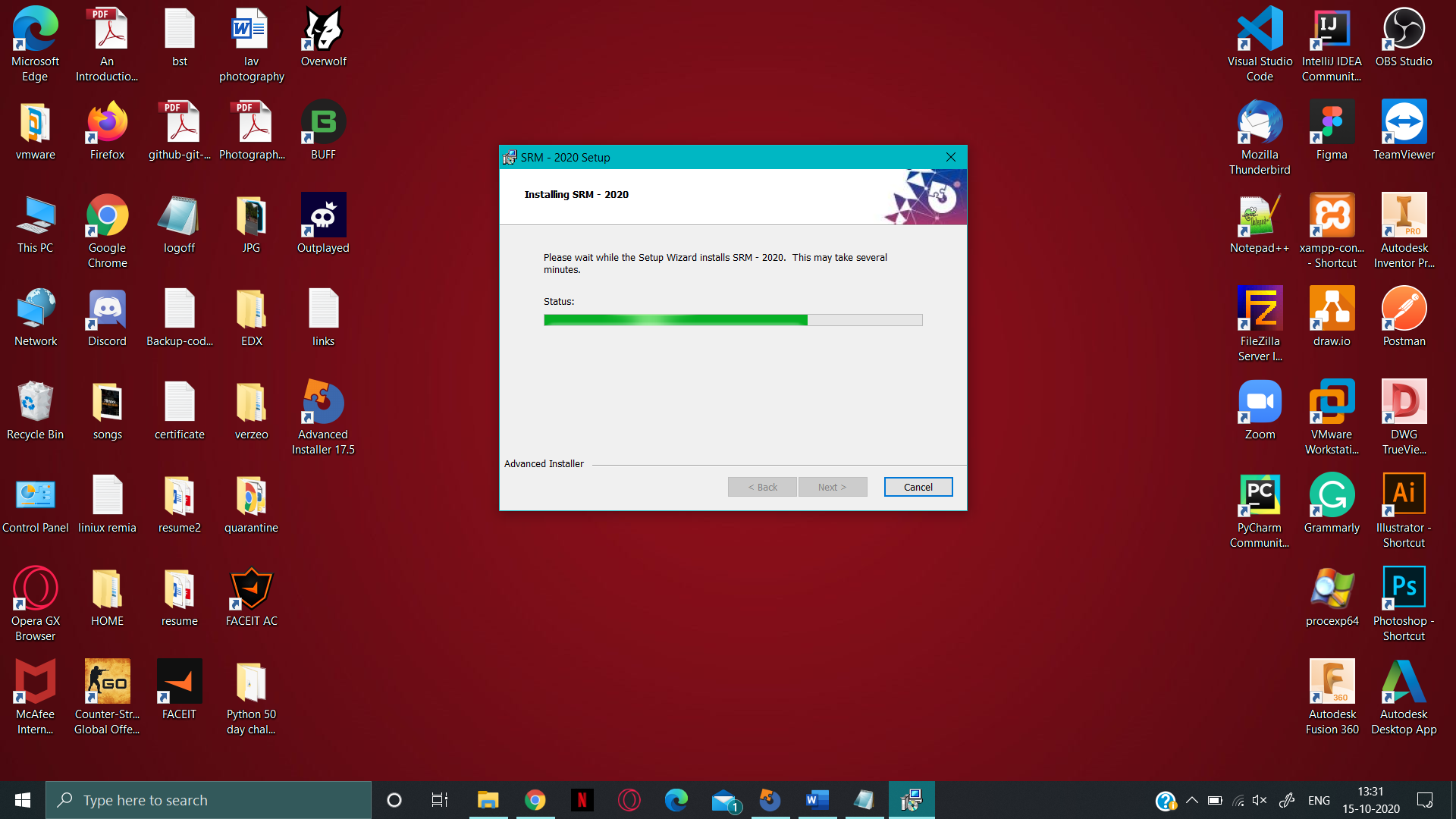
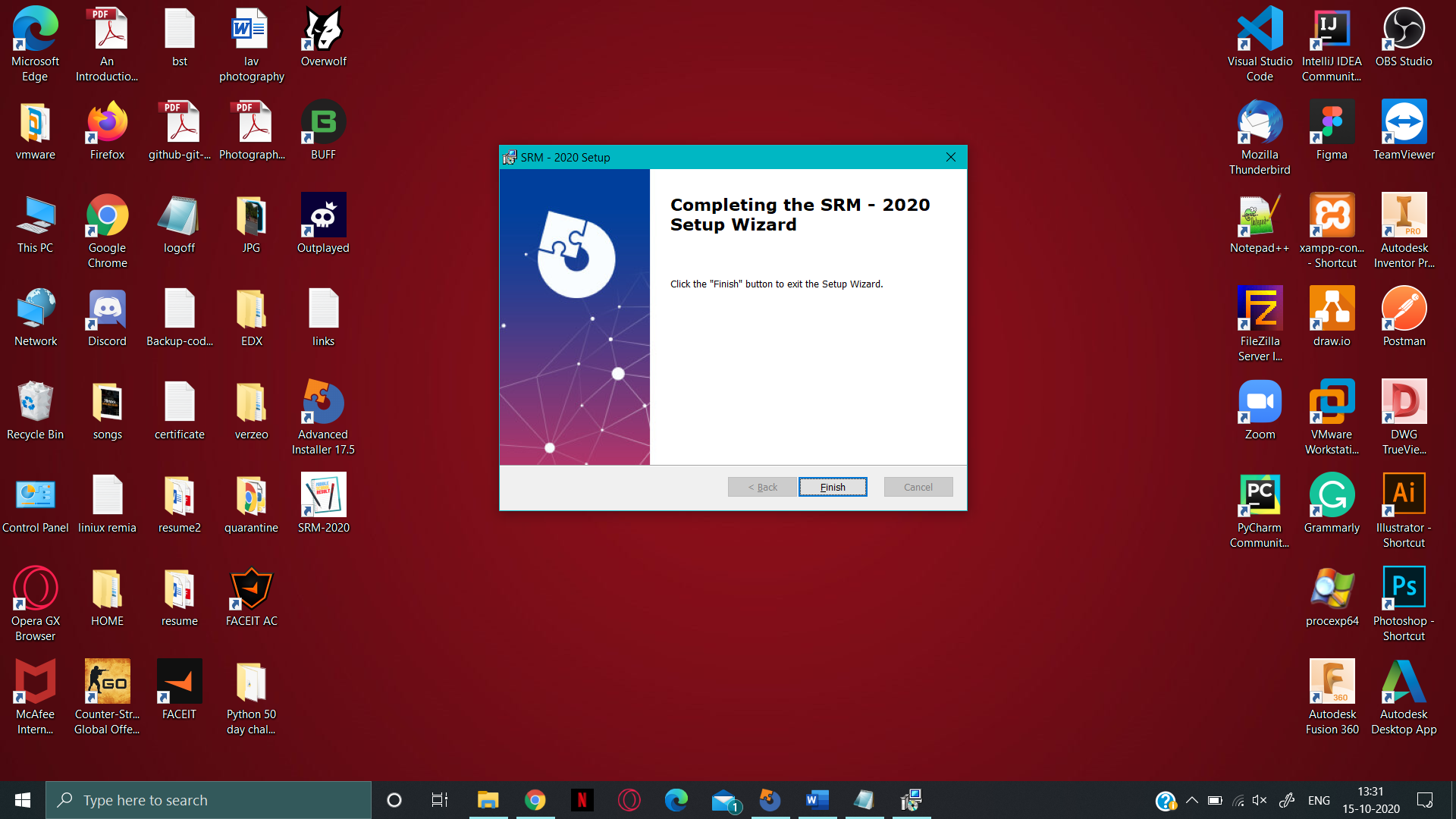
*pyinstaller main.py --add-data "C:\Users\jaypr\AppData\Roaming\Python\Python37\site-packages\mysql";"mysql" --noconsole --onefile --name "SRM"*

Deployment:

I packed the exe file into MSI file using advanced installer software. Then when you double click on the MSI file This installation wizard will open. It will guide you through the installation procedure.

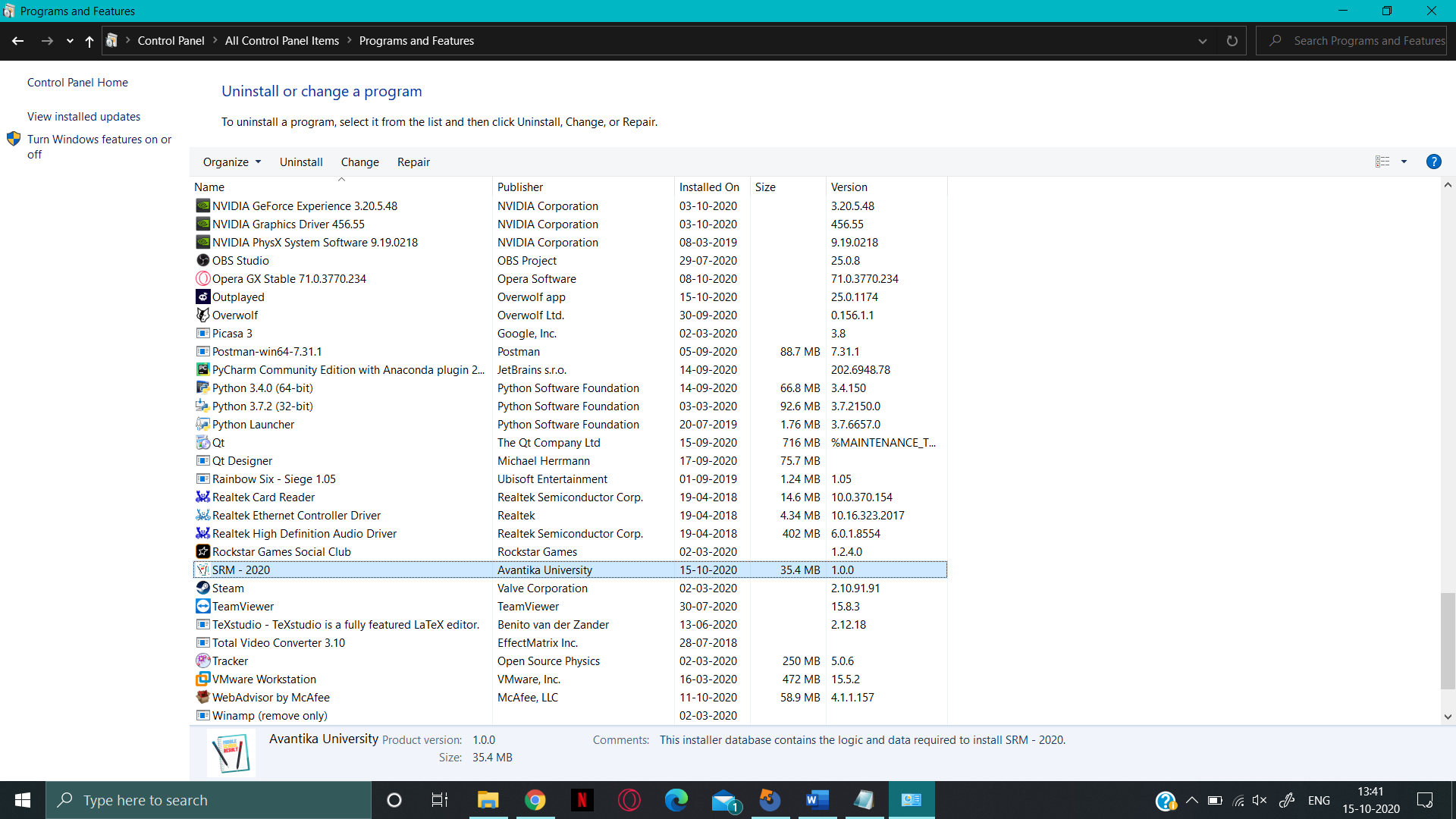
 

Figure This is the image of Desktop and control panel where you can see the SRM - 2020 software