

Term Id: 220211

Subject: INT213



L OVELY
P ROFESSIONAL
U NIVERSITY

CA-1

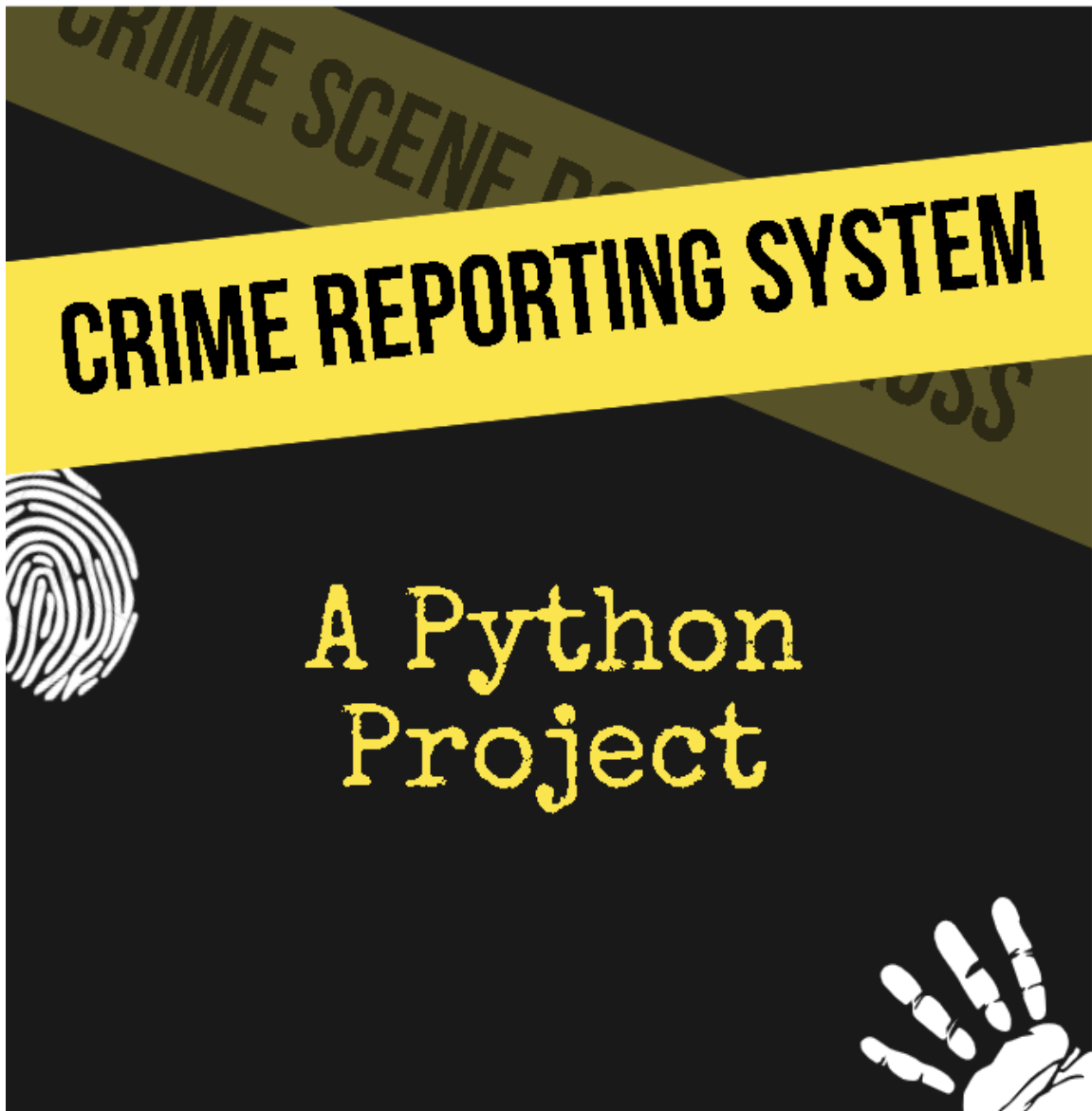
Group Members:

| Name | Roll No | Reg. No. |
|-------------|----------------|-----------------|
| Alan James | 20 | 11903972 |
| Joy Garg | 62 | 11918595 |

Project Name: Crime Reporting System

Section: K19KH

Submitted to: Mrs. Ankita Wadhawan



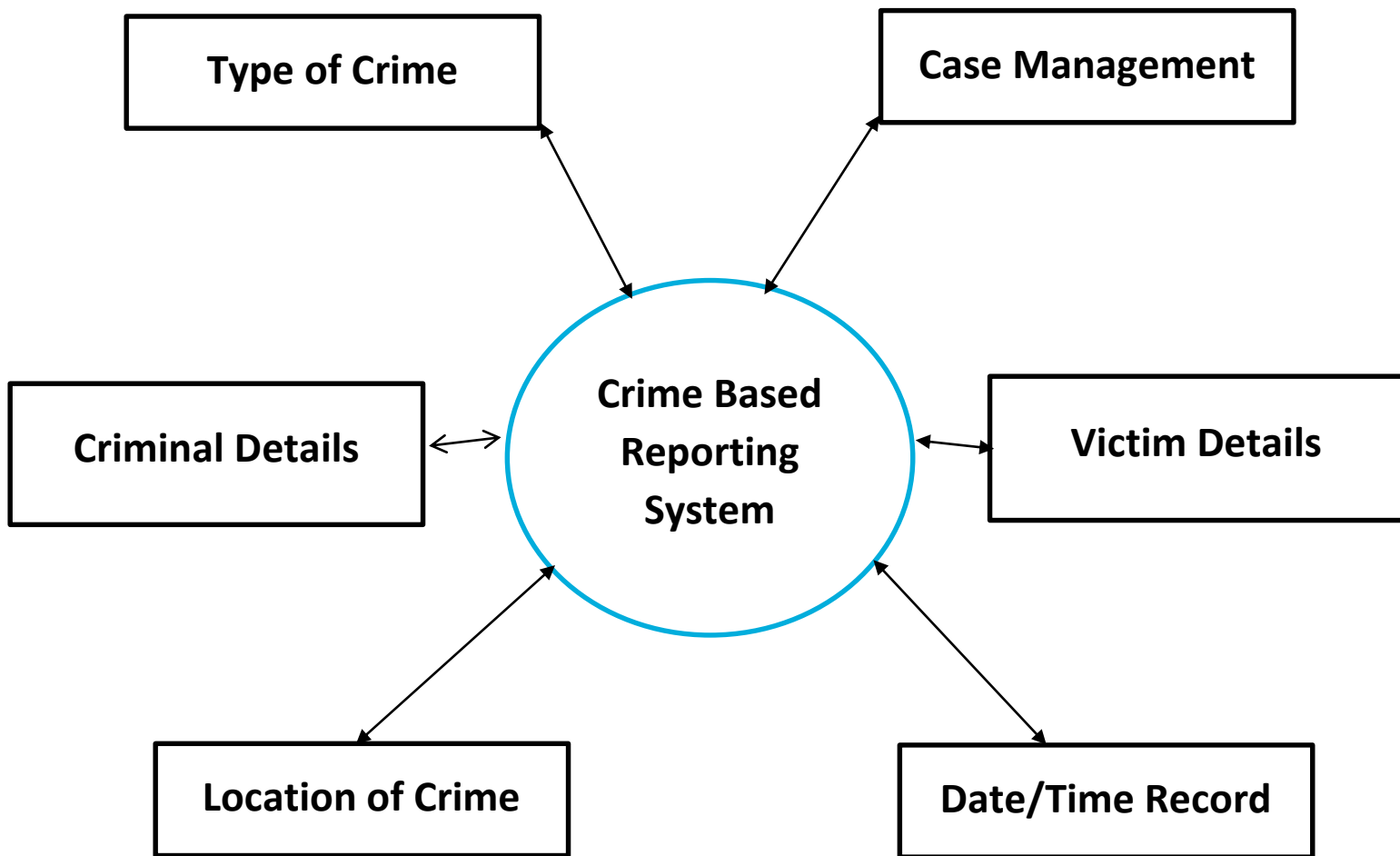
CRIME REPORTING SYSTEM

Built using Python with the additional libraries tkinter and sqlite.

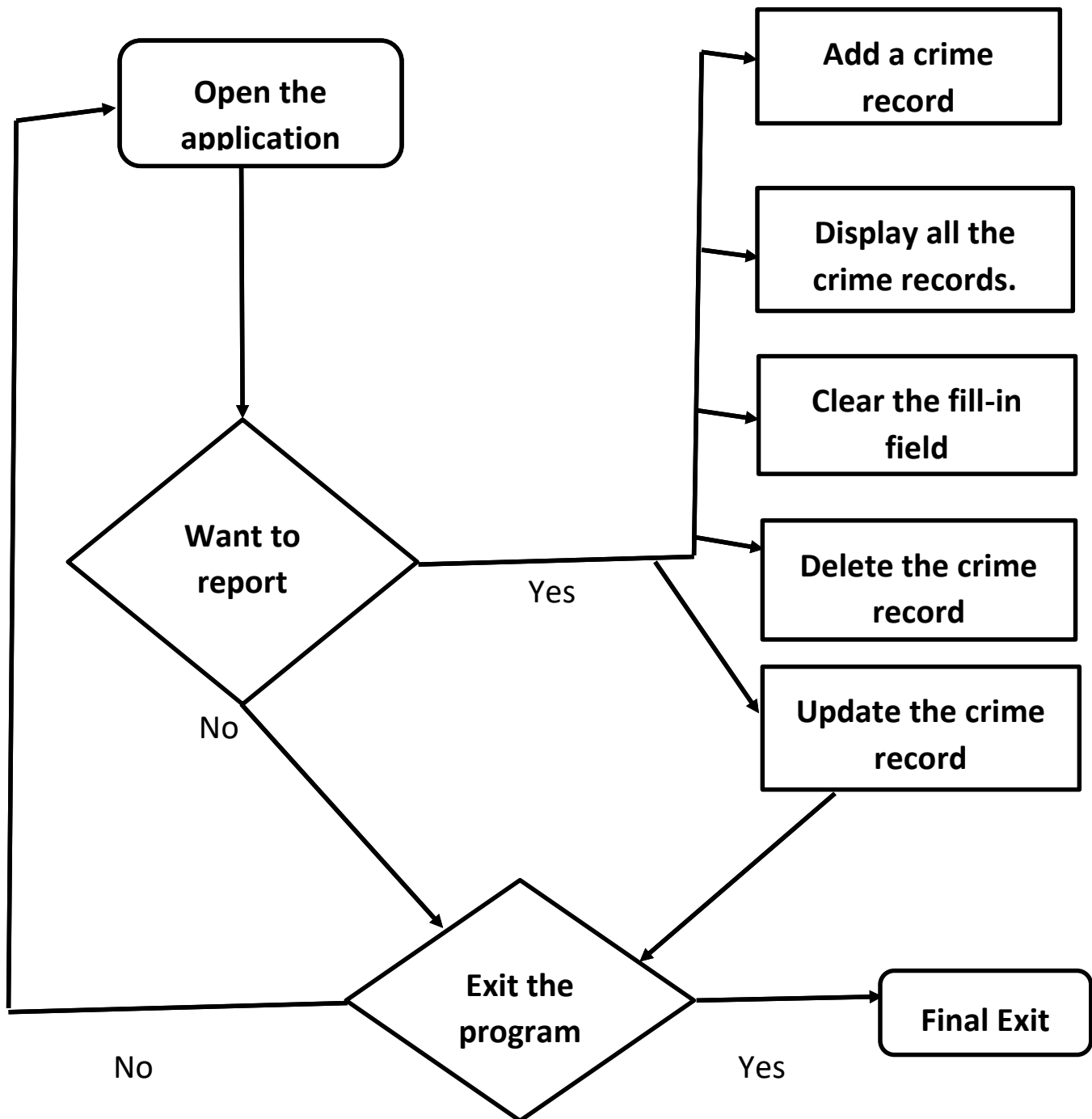
INTRODUCTION

Crime is an unlawful and punishable offence under law, and the state. Crime Reporting is an actual respective reporting of the incident for better investigation, prosecution and judgement. Over the years, crime rate has significantly grown to stooping levels. The Crime Reporting System discussed here is a simple approach for any basic level application to complex. It helps you to carefully fill in the first investigation process findings, with details. This helps the police forces to capture, identify the criminal or person behind the crime. This Python project also uses database to its advantage to store the details permanently and do the necessary updating or clarification later on.

DATA FLOW DIAGRAM

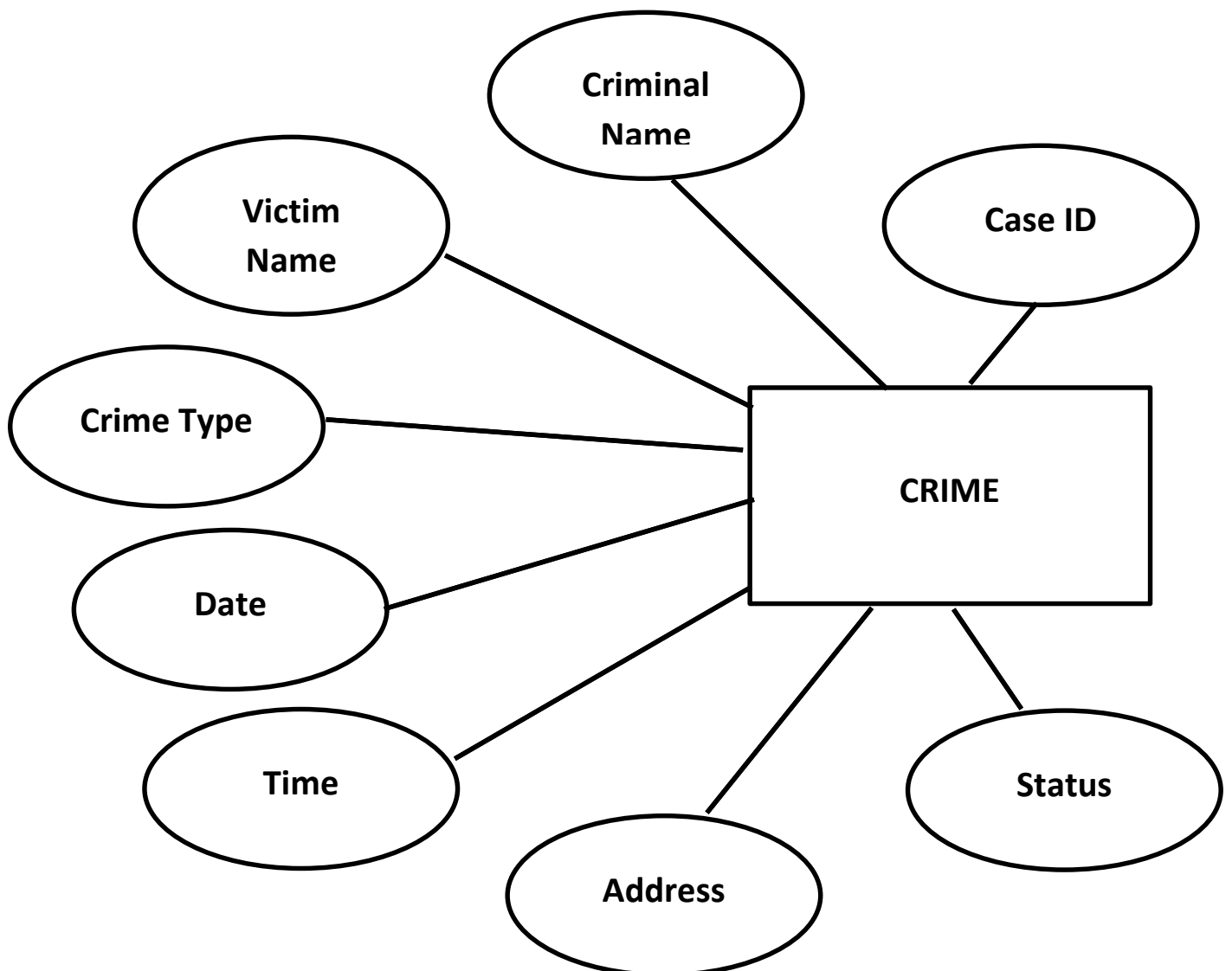


FLOW CHART



ENTITY-RELATIONSHIP

DIAGRAM



SCREENSHOTS

Programming Code: Frontend

```
StudentsDatabaseManagementSystems.py X StdDatabase.py
C:\Users\alaan\Desktop>INT208 Project > StudentsDatabaseManagementSystems.py > Student > __init__
1 from tkinter import*
2 import tkinter.messagebox
3 import StdDatabase
4 #Frontend
5
6
7 class Student:
8
9     def __init__(self,root):
10         self.root =root
11         self.root.title("Crime Reporting System")
12         self.root.geometry("1350x750+0+0")
13         self.root.config(bg="maroon")
14
15         StdID = StringVar()
16         Firstname = StringVar()
17         Surname = StringVar()
18         DoB = StringVar()
19         Age = StringVar()
20         Gender = StringVar()
21         Address = StringVar()
22         Mobile = StringVar()
23
24         #=====Function Declaration=====
25
26     def iExit():
27         iExit = tkinter.messagebox.askyesno("Crime Reporting Sytems","Confirm if you want to exit")
28         if iExit > 0:
29             root.destroy()
30             return
31
32     def ClearData():
33         self.txtStdID.delete(0,END)
34         self.txtfna.delete(0,END)
35         self.txtsna.delete(0,END)
36         self.txtDoB.delete(0,END)
37         self.txtAge.delete(0,END)
38         self.txtGender.delete(0,END)
39
40         self.txtAdr.delete(0,END)
41         self.txtMobile.delete(0,END)
42
43     def addData():
44         if(len(StdID.get())!=0):
45             StdDatabase.addStdRec(StdID.get(), Firstname.get(), Surname.get(), DoB.get(), Age.get(), Gender.get(), Address.get(), Mobile.get())
46             studentlist.delete(0,END)
47             studentlist.insert(END,(StdID.get(), Firstname.get(), Surname.get(), DoB.get(), Age.get(), Gender.get(), Address.get(), Mobile.get()))
48
49     def DisplayData():
50         studentlist.delete(0,END)
51         for row in StdDatabase.viewData():
52             studentlist.insert(END,row,str(""))
53
54     def StudentRec(event):
55         global sd
56         searchStd = studentlist.curselection()[0]
57         sd = studentlist.get(searchStd )
58
59         self.txtStdID.delete(0,END)
60         self.txtStdID.insert(END,sd[1])
61         self.txtfna.delete(0,END)
62         self.txtfna.insert(END,sd[2])
63         self.txtsna.delete(0,END)
64         self.txtsna.insert(END,sd[3])
65         self.txtDoB.delete(0,END)
66         self.txtDoB.insert(END,sd[4])
67         self.txtAge.delete(0,END)
68         self.txtAge.insert(END,sd[5])
69         self.txtGender.delete(0,END)
70         self.txtGender.insert(END,sd[6])
71         self.txtAdr.delete(0,END)
72         self.txtAdr.insert(END,sd[7])
73         self.txtMobile.delete(0,END)
74         self.txtMobile.insert(END,sd[8])
```

```

StudentsDatabaseManagementSystems.py X StdDatabase.py
C:\Users\alaan\Desktop>INT208 Project>StudentsDatabaseManagementSystems.py>Student>__init__

75     def DeleteData():
76         if(len(stdID.get())!=0):
77             StdDatabase.deleteRec(sd[0])
78             ClearData()
79             DisplayData()
80
81     def searchDatabase():
82         studentlist.delete(0,END)
83         for row in StdDatabase.searchData(stdID.get(), Firstname.get(),Surname.get(),DoB.get(), Age.get(), Gender.get(), Address.get(), Mobile.get()):
84             studentlist.insert(END,row,str(""))
85
86     #def update():
87         #if(len(stdID.get())!=0):
88             #StdDatabase.dataUpdate(sd[0],(stdID.get(), Firstname.get(),Surname.get(),DoB.get(), Age.get(),Gender.get(),Address.get(),Mobile.get()))
89
90     def update():
91         if(len(stdID.get())!=0):
92             StdDatabase.deleteRec(sd[0])
93         if(len(stdID.get())!=0):
94             StdDatabase.addStdRec(stdID.get(), Firstname.get(), Surname.get(), DoB.get(), Age.get(), Gender.get(), Address.get(), Mobile.get())
95             studentlist.delete(0,END)
96             studentlist.insert(END,(stdID.get(), Firstname.get(), Surname.get(), DoB.get(), Age.get(), Gender.get(), Address.get(), Mobile.get()))
97
98
99     #=====Frame=====
100     MainFrame =Frame(self.root, bg="maroon")
101     MainFrame.grid()
102
103     TitFrame = Frame(MainFrame, bd=2,  padx=54,pady=8, bg="Ghost White", relief=RIDGE)
104     TitFrame.pack(side=TOP)
105
106     self.lblTit =Label(TitFrame,font=('arial', 47,'bold'),text="Crime Reporting System",bg="Ghost White")
107     self.lblTit.grid(sticky=W)
108
109     ButtonFrame =Frame(MainFrame, bd=2, width=1350, height=70, padx=18,pady=10, bg="Ghost White", relief=RIDGE)
110     ButtonFrame.pack(side=BOTTOM)

```

```

StudentsDatabaseManagementSystems.py X StdDatabase.py
C:\Users\alaan\Desktop>INT208 Project>StudentsDatabaseManagementSystems.py>Student>__init__

112     DataFrame =Frame(MainFrame, bd=1, width=1300, height=400, padx=20, pady=20, relief=RIDGE, bg="maroon")
113     DataFrame.pack(side=BOTTOM)
114
115     DataFrameLEFT =LabelFrame(DataFrame, bd=1, width=1000, height=600, padx=20, relief=RIDGE
116     , font=('arial', 20,'bold'), text="Crime Info:\n",bg="Ghost White")
117     DataFrameLEFT.pack(side=LEFT)
118
119     DataFrameRIGHT =LabelFrame(DataFrame, bd=1, width=450, height=300, padx=31,pady=3, relief=RIDGE
120     , font=('arial', 20,'bold'),bg="Ghost White", text="Crime Details:\n",)
121     DataFrameRIGHT.pack(side=RIGHT)
122     #=====Labels and Entrys=====
123     self.lblStdID =Label(DataFrameLEFT,font=('arial',20,'bold'),text="Case ID:",padx=2,pady=2,bg="Ghost White")
124     self.lblStdID.grid(row=0, column =0,sticky=W)
125     self.txtStdID =Entry(DataFrameLEFT,font=('arial',20,'bold'), textvariable=stdID, width=39)
126     self.txtStdID.grid(row=0, column =1)
127
128     self.blbfna =Label(DataFrameLEFT, font=('arial', 20,'bold'), text="Name:",padx=2,pady=2 ,bg="Ghost White")
129     self.blbfna.grid(row=1, column=0,sticky=W)
130     self.txtfna=Entry(DataFrameLEFT, font=('arial', 20,'bold'),textvariable =Firstname , width=39)
131     self.txtfna.grid(row=1, column=1)
132
133     self.blbsna =Label(DataFrameLEFT, font=('arial', 20,'bold'), text="Type of Crime:",padx=2,pady=2 ,bg="Ghost White")
134     self.blbsna.grid(row=2, column=0,sticky=W)
135     self.txtsna=Entry(DataFrameLEFT, font=('arial', 20,'bold'),textvariable = Surname, width=39)
136     self.txtsna.grid(row=2, column=1)
137
138     self.blldoB =Label(DataFrameLEFT, font=('arial', 20,'bold'),text="Date of Crime:",padx=2,pady=3 ,bg="Ghost White")
139     self.blldoB.grid(row=3, column=0,sticky=W)
140     self.txtdoB=Entry(DataFrameLEFT,font=('arial', 20,'bold'),textvariable = DoB,width=39)
141     self.txtdoB.grid(row=3, column=1)
142
143     self.blbAge =Label(DataFrameLEFT, font=('arial', 20,'bold'), text="Time:",padx=2,pady=3 ,bg="Ghost White")
144     self.blbAge.grid(row=4, column=0,sticky=W)
145     self.txtAge=Entry(DataFrameLEFT, font=('arial', 20,'bold'),textvariable = Age, width=39)
146     self.txtAge.grid(row=4, column=1)

```



```

148 self.lblGender =Label(DataFrameLEFT, font=('arial', 20,'bold'),text="VictimsName:",padx=2,pady=3 ,bg="Ghost White")
149 self.lblGender.grid(row=5, column=0,sticky=W)
150 self.txtGender=Entry(DataFrameLEFT, font=('arial', 20,'bold'),textvariable = Gender, width=39)
151 self.txtGender.grid(row=5, column=1)
152
153 self.lblAdr =Label(DataFrameLEFT, font=('arial', 20,'bold'), text="Address:",padx=2,pady=3 ,bg="Ghost White")
154 self.lblAdr.grid(row=6, column=0,sticky=W)
155 self.txtAdr=Entry(DataFrameLEFT, font=('arial', 20,'bold'),textvariable = Address, width=39)
156 self.txtAdr.grid(row=6, column=1)
157
158 self.lblMobile =Label(DataFrameLEFT, font=('arial', 20,'bold'),text="Status :",padx=2,pady=3 ,bg="Ghost White")
159 self.lblMobile.grid(row=7, column=0,sticky=W)
160 self.txtMobile =Entry(DataFrameLEFT, font=('arial', 20,'bold'),textvariable = Mobile , width=39)
161 self.txtMobile.grid(row=7, column=1)
162 #=====Listbox and Scrollbar=====
163
164 scrollbar = Scrollbar(DataFrameRIGHT)
165 scrollbar.grid(row=0, column=1, sticky ='ns')
166
167 studentlist = Listbox(DataFrameRIGHT, width = 41, height=16, font=('arial', 12,'bold'), yscrollcommand=scrollbar.set)
168 studentlist.bind('<<ListboxSelect>>', StudentRec)
169 studentlist.grid(row=0, column=0, padx=8)
170 scrollbar.config(command=studentlist.yview)
171 #=====Buttons Widget=====
172 self.btnAddData=Button(ButtonFrame, text='Add New', font=('arial', 20,'bold'),height=1, width=10, bd=4,
173                        command=addData)
174 self.btnAddData.grid(row=0,column=0)
175
176 self.btnDisplayData=Button(ButtonFrame, text='Display', font=('arial', 20,'bold'),height=1, width=10,
177                        bd=4, command=DisplayData)
178 self.btnDisplayData.grid(row=0,column=1)
179
180 self.btnClearData=Button(ButtonFrame, text='Clear', font=('arial', 20,'bold'),height=1, width=10, bd=4,
181                        command=ClearData)
182 self.btnClearData.grid(row=0,column=2)

```

```

184 self.btnDeleteData=Button(ButtonFrame, text='Delete', font=('arial', 20,'bold'),height=1, width=10, bd=4,
185                        command=DeleteDate)
186 self.btnDeleteData.grid(row=0,column=3)
187
188 self.btnSearchData=Button(ButtonFrame, text='Search', font=('arial', 20,'bold'),height=1, width=10, bd=4,
189                        command =searchDatabase)
190 self.btnSearchData.grid(row=0,column=4)
191
192 self.btnUpdateData=Button(ButtonFrame, text='Update', font=('arial', 20,'bold'),height=1, width=10, bd=4,
193                        command = update)
194 self.btnUpdateData.grid(row=0,column=5)
195
196 self.btnExit=Button(ButtonFrame, text='Exit', font=('arial', 20,'bold'),height=1, width=10, bd=4, command=iExit)
197 self.btnExit.grid(row=0,column=6)
198
199
200
201
202 if __name__=='__main__':
203     root = Tk()
204     application = Student(root)
205     root.mainloop()

```

Backend

```
StudentsDatabaseManagementSystems.py StdDatabase.py X
C:\Users\alaan\Desktop\INT208 Project> StdDatabase.py > ...
1  import sqlite3
2  #backend
3
4  def studentData():
5      con=sqlite3.connect("student.db")
6      cur = con.cursor()
7      cur.execute("CREATE TABLE IF NOT EXISTS student (id INTEGER PRIMARY KEY, StdID text,Firstname text,Surname text,DoB text, \
8          Age text,Gender text,Address text,Mobile text)")
9      con.commit()
10     con.close()
11
12     def addStdRec(StdID, Firstname,Surname,DoB, Age,Gender, Address, Mobile):
13         con=sqlite3.connect("student.db")
14         cur = con.cursor()
15         cur.execute("INSERT INTO student VALUES (NULL,?,?,?,?,?,?,?,?)", \
16             (StdID, Firstname,Surname,DoB, Age,Gender, Address, Mobile))
17         con.commit()
18         con.close()
19
20     def viewData():
21         con=sqlite3.connect("student.db")
22         cur = con.cursor()
23         cur.execute("SELECT * FROM student")
24         rows=cur.fetchall()
25         con.close
26         return rows
27
28     def deleteRec(id):
29         con=sqlite3.connect("student.db")
30         cur = con.cursor()
31         cur.execute("DELETE FROM student WHERE id=?", (id,))
32         con.commit()
33         con.close
34
35
36     def searchData(StdID="", Firstname="",Surname="",DoB="", Age="",Gender="", Address="", Mobile=""):
37         con=sqlite3.connect("student.db")
38         cur = con.cursor()
39         cur.execute("SELECT * FROM student WHERE StdID=? OR Firstname=? OR Surname=? OR DoB=? OR Age=? OR Gender=? OR Address=? OR Mobile=? ", \
40             (StdID, Firstname,Surname,DoB, Age,Gender, Address, Mobile))
41         rows=cur.fetchall()
42         con.close()
43         return rows
44
45     def dataUpdate(id,StdID="", Firstname="",Surname="",DoB="", Age="",Gender="", Address="", Mobile=""):
46         con=sqlite3.connect("student.db")
47         cur = con.cursor()
48         cur.execute("UPDATE student SET StdID=?, Firstname=?,Surname=?,DoB=?, Age=?,Gender=?, Address=?, Mobile=?, WHERE id=?", \
49             (StdID, Firstname,Surname,DoB, Age,Gender, Address, Mobile, id))
50         con.commit()
51         con.close()
52
53
54
55
56
57     studentData()
```

Output:

- Adding some records to the database.
After clicking the **add** button, the record gets stored onto the database. This is indicated by the display of record in the Crime Details Space.

The screenshot displays the 'Crime Reporting System' window. It features a title bar with the text 'Crime Reporting System' and standard window controls. The main area is divided into two sections: 'Crime Info:' and 'Crime Details:'. The 'Crime Info:' section contains a form with the following fields and values:

| | |
|----------------|-----------------|
| Case ID: | 100 |
| Name: | John Wick |
| Type of Crime: | Multiple Murder |
| Date of Crime: | 13/05/2019 |
| Time: | 18:00 |
| VictimsName: | Nemo Jarvis |
| Address: | Bangalore |
| Status : | Open |

The 'Crime Details:' section contains a text area displaying the record: '100 {John Wick} {Multiple Murder} 13/05/2019 18:00'. Below the form, there is a row of buttons: 'Add New', 'Display', 'Clear', 'Delete', 'Search', 'Update', and 'Exit'.

- After adding some records, when **Display** button is clicked. We retrieve all the data from database and display it at the Crime Details space on the right.

Crime Reporting System

Crime Reporting System

Crime Info:

Case ID:
Name:
Type of Crime:
Date of Crime:
Time:
VictimsName:
Address:
Status :

Crime Details:

1 100 {John Wick} {Multiple Murder} 13/05/2019 16:00
2 101 {Raj Kumar} Theft 09/12/2010 13:29 {Arpit Y
3 102 {Santhosh Kumar} Assault 12/04/2015 17:00
4 103 {Abishek Singh} Trafficking 15/08/2018 21:20
5 104 {Abhimanyu Tripathi} {Illegal Drugs Use} 15/08/2018 21:20
6 105 {Aditi Singh} Fraud 25/11/2010 14:37 {Coop

Add New

Display

Clear

Delete

Search

Update

Exit

- When we click the **clear** button, the added entry automatically clears from every field.

Crime Reporting System

Crime Info:

Case ID: 100
 Name: John Wick
 Type of Crime: Multiple Murder
 Date of Crime: 13/05/2019
 Time: 18:00
 VictimsName: Nemo Jarvis
 Address: Bangalore
 Status : Open

Crime Details:

1 100 {John Wick} {Multiple Murder} 13/05/2019 18:00
 2 101 {Raj Kumar} Theft 09/12/2010 13:29 {Arpit Y
 3 102 {Santhosh Kumar} Assault 12/04/2015 17:01
 4 103 {Abhishek Singh} Trafficking 15/08/2018 21:12
 5 104 {Abhimanyu Tripathi} {Illegal Drugs Use} 15/08/2018 21:12
 6 105 {Aditi Singh} Fraud 25/11/2010 14:37 {Coop

Add New Display Clear Delete Search Update Exit

- By giving in the variable, we **search** for record shown in Crime Details space.

Crime Reporting System

Crime Info:

Case ID: 100
 Name:
 Type of Crime:
 Date of Crime:
 Time:
 VictimsName:
 Address:
 Status :

Crime Details:

1 100 {John Wick} {Multiple Murder} 13/05/2019 18:00

Add New Display Clear Delete Search Update Exit

- When we click the **delete** button, that particular record is deleted from database as well as it vanishes from the Crime Details Space.

Crime Reporting System

| Crime Info: | Crime Details: |
|------------------------------------|--|
| Case ID: 102 | 1 100 {John Wick} {Multiple Murder} 13/05/2019 11:00 |
| Name: Santhosh Kumar | 2 101 {Raj Kumar} Theft 09/12/2010 13:29 {Arpit Y} |
| Type of Crime: Assault | 3 102 {Santhosh Kumar} Assault 12/04/2015 17:05 |
| Date of Crime: 12/04/2015 | 5 104 {Abhimanyu Tripathi} {Illegal Drugs Use} 15/08/2010 14:37 {Coop} |
| Time: 17:05 | 6 105 {Aditi Singh} Fraud 26/11/2010 14:37 {Coop} |
| VictimsName: Harsh Yadav | |
| Address: Near SBI Branch, Mainpuri | |
| Status : Open | |

Add New Display Clear Delete Search Update Exit

Crime Reporting System

| Crime Info: | Crime Details: |
|----------------|--|
| Case ID: | 1 100 {John Wick} {Multiple Murder} 13/05/2019 11:00 |
| Name: | 2 101 {Raj Kumar} Theft 09/12/2010 13:29 {Arpit Y} |
| Type of Crime: | 5 104 {Abhimanyu Tripathi} {Illegal Drugs Use} 15/08/2010 14:37 {Coop} |
| Date of Crime: | 6 105 {Aditi Singh} Fraud 26/11/2010 14:37 {Coop} |
| Time: | |
| VictimsName: | |
| Address: | |
| Status : | |

Add New Display Clear Delete Search Update Exit

- We select the record to be updated, and click **Update** button to make changes.

Crime Reporting System

| Crime Info: | Crime Details: |
|--------------------------------|--|
| Case ID: 101 | 1 '00 (John Wick)-[Multiple Murder] 13062215:14 |
| Name: Raj Kumar | 2 '01 (Raj Kumar) Theft 09/12/2010 13:29 [Arpit Y] |
| Type of Crime: Theft | 3 '04 (Abhimanyu Tripathi)-[Illegal Drugs Use] 15 |
| Date of Crime: 09/12/2010 | 4 '05 (Aadi Singh) Fraud 25/11/2010 14:37 [Coop |
| Time: 13:29 | |
| VictimsName: Arpit Yadav | |
| Address: Children's Park, Agra | |
| Status : Closed | |

Crime Reporting System

| Crime Info: | Crime Details: |
|--------------------------------|--|
| Case ID: 101 | 101 (Raj Kumar) Cyberbullying 09/12/2010 13:29 |
| Name: Raj Kumar | |
| Type of Crime: Cyberbullying | |
| Date of Crime: 09/12/2010 | |
| Time: 13:29 | |
| VictimsName: Arpit Yadav | |
| Address: Children's Park, Agra | |
| Status : Closed | |

- When we select the delete button, a confirmation opens up asking Yes/No. Depending on your decision the program proceeds.

The screenshot displays the 'Crime Reporting System' application window. The title bar reads 'Crime Reporting System'. The main window has a red header with the title 'Crime Reporting System'. Below the header, there are two main sections: 'Crime Info:' and 'Crime Details:'. The 'Crime Info:' section contains a form with the following fields: Case ID (101), Name (Raj Kumar), Type of Crime (Cyberbullying), Date of Crime (09/12/2010), Time (13:29), VictimsName (Arpit Yadav), Address (Children's Park, Agra), and Status (Closed). The 'Crime Details:' section shows a list of crime records, with the first entry being '101 {Raj Kumar} Cyberbullying 09/12/2010 13:29'. A confirmation dialog box is open in the center of the screen, titled 'Crime Reporting Systems', with a question mark icon and the text 'Confirm if you want to exit'. It has 'Yes' and 'No' buttons. At the bottom of the application window, there is a row of buttons: 'Add New', 'Display', 'Clear', 'Delete', 'Search', 'Update', and 'Exit'.

| Crime Info: | | Crime Details: |
|----------------|-----------------------|--|
| Case ID: | 101 | 101 {Raj Kumar} Cyberbullying 09/12/2010 13:29 |
| Name: | Raj Kumar | |
| Type of Crime: | Cyberbullying | |
| Date of Crime: | 09/12/2010 | |
| Time: | 13:29 | |
| VictimsName: | Arpit Yadav | |
| Address: | Children's Park, Agra | |
| Status : | Closed | |

Buttons: Add New, Display, Clear, Delete, Search, Update, Exit

RESULTS

After preparing this basic Crime Reporting System, one is able to combine the details accurately and is user-friendly. Crime Reporting depends on the victim's mind and style. It is a necessary thing to address since we all require a decent crime-free neighborhood. Through this project we also came about the different challenges one might face, when reporting a crime. The essential process being reporting, and management is possible through flexible supervision and agreement. This calls for many opportunities by integrating this project with API calls to make it more interactive and easiness for the user. Also, we can use some ML models and algorithms to better study from the existing database, and make decisions and predictions to better configure and define the security and crime-free environment in our lives.

