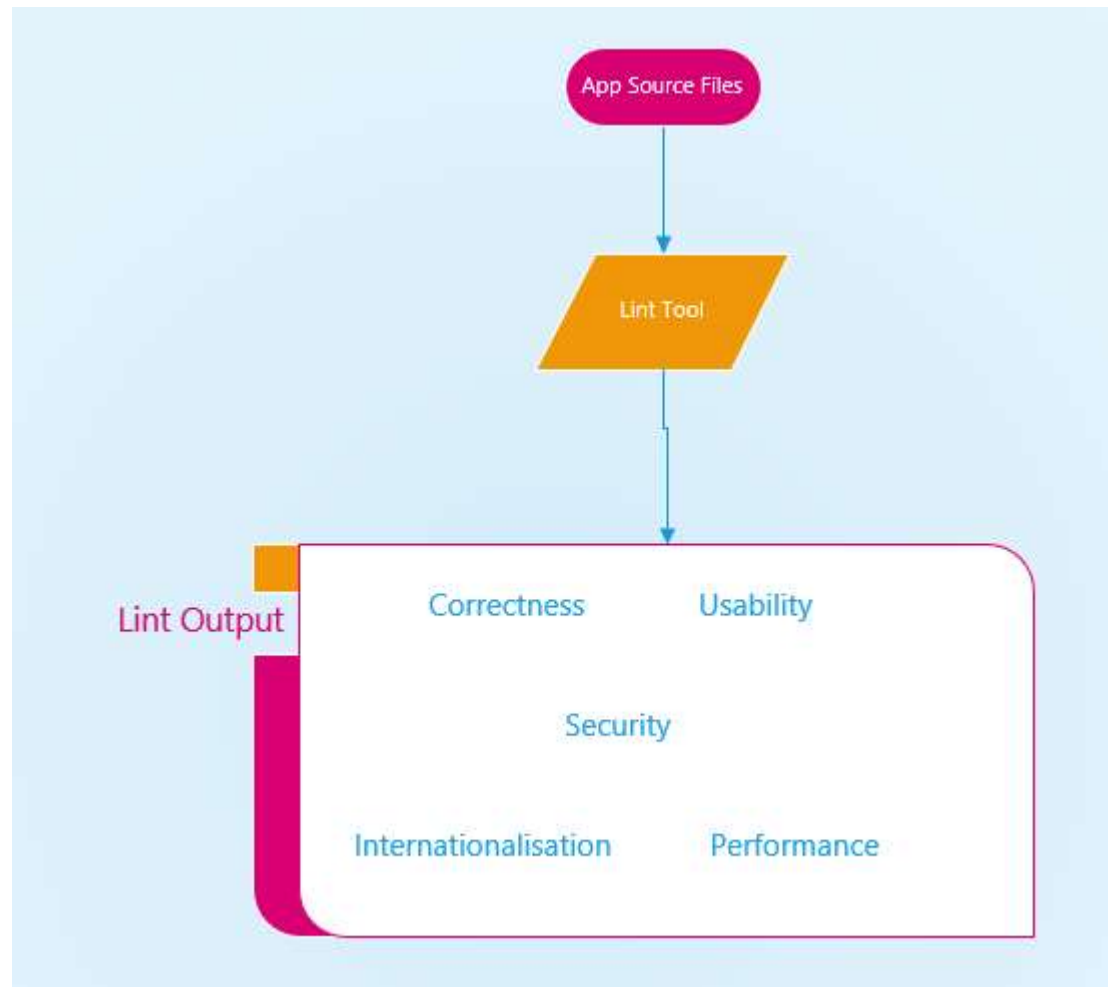


## Code improvement with lint checks.

In addition to ensuring the application meets functional requirements by building tests, it was essential to ensure that the code has no structural problems by running it through lints. Lints were used for identifying and correcting issues with the structural quality of the code. Critical improvements were made for correctness, security, performance, usability, and internationalisation.



## 1. Main (main.py)

### 1.1 Lint Output before corrections.

```
1  Check results
2  =====
3
4  E265:1:1:block comment should start with '#'
5  W291:1:17:trailing whitespace
6  E265:2:1:block comment should start with '#'
7  W291:3:13:trailing whitespace
8  W293:7:1:blank line contains whitespace
9  E111:8:3:indentation is not a multiple of four
10 E111:9:3:indentation is not a multiple of four
11 W291:9:57:trailing whitespace
12 E111:10:3:indentation is not a multiple of four
13 W293:11:1:blank line contains whitespace
14 E111:13:3:indentation is not a multiple of four
15 W293:14:1:blank line contains whitespace
16 E203:17:40:whitespace before ':'
17 E111:18:3:indentation is not a multiple of four
18 E501:18:80:line too long (103 > 79 characters)
19 E111:19:3:indentation is not a multiple of four
20 W291:19:48:trailing whitespace
21 E111:20:3:indentation is not a multiple of four
22 E203:20:24:whitespace before ':'
23 E203:21:8:whitespace before ':'
24 E111:22:7:indentation is not a multiple of four
25 E111:24:7:indentation is not a multiple of four
26 E501:24:80:line too long (89 > 79 characters)
27 E203:25:11:whitespace before ':'
28 E111:26:7:indentation is not a multiple of four
29 W293:27:1:blank line contains whitespace
30 E111:28:3:indentation is not a multiple of four
31 E203:28:26:whitespace before ':'
32 E111:29:7:indentation is not a multiple of four
33 E203:29:10:whitespace before ':'
34 W293:31:1:blank line contains whitespace
35 E111:32:11:indentation is not a multiple of four
36 W293:33:1:blank line contains whitespace
37 W293:35:1:blank line contains whitespace
38 E111:36:7:indentation is not a multiple of four
39 E203:36:13:whitespace before ':'
40 W293:38:1:blank line contains whitespace
41 W391:41:1:blank line at end of file
42
```

```

43 Code
44 =====
45 #import register
46 #import tcp_packet_capture
47 import login
48
49
50 def print_hi(name):
51
52     print("++++++++++++++++++++++++++++++++++++")
53     print(f'*** SSD Group 4 Internet Forensics Tools ***')
54     print("++++++++++++++++++++++++++++++++++++")
55
56 if __name__ == '__main__':
57     print_hi('')
58
59     tries = 0
60     tries_flag = ""
61     while tries_flag != "Close the program" :
62         print("\n- To know more about our Tools Enter 1          \n- If you are exist user Enter 2 for login      ")
63         User_choice = input("\nEnter your choice : ")
64         if User_choice == "1" :
65             try :
66                 if __name__ == '__main__':
67                     print_hi('')
68                     print ("***You use this tools to capture and analyse your internet traffics*** ")
69             except :
70                 print("Database connectivity issues")
71
72         elif User_choice == "2" :
73             try :
74                 if __name__ == '__main__':
75
76                     print_hi('')
77
78                     login.User_login()
79
80             except :
81                 print("Database connectivity issues")
82

```

## 1.2 Lint output post corrections.

```
1 Check results
2 =====
3
4 E501:18:80:line too long (105 > 79 characters)
5 E501:24:80:line too long (94 > 79 characters)
6
7 Code
8 =====
9 # import register
10 # import tcp_packet_capture
11 import login
12
13
14 def print_hi(name):
15     print("+++++")
16     print(f'*** SSD Group 4 Internet Forensics Tools ***')
17     print("+++++")
18
19
20 if __name__ == '__main__':
21     print_hi('')
22
23 tries = 0
24 tries_flag = ""
25 while tries_flag != "Close the program":
26     print("\n- To know more about our Tools Enter 1          \n- If you are exist user Enter 2 for login      ")
27     User_choice = input("\nEnter your choice : ")
28     if User_choice == "1":
29         try:
30             if __name__ == '__main__':
31                 print_hi('')
32                 print(" ***You use this tools to capture and analyse your internet traffics*** ")
33         except:
34             print("Database connectivity issues")
35
36     elif User_choice == "2":
37         try:
38             if __name__ == '__main__':
39                 print_hi('')
40
41                 login.User_login()
42
43         except:
44             print("Database connectivity issues")
45
```

## 2. Create Database(create\_database.py)

### 2.1 Lint Output before corrections.

```
1 Check results
2 =====
3
4 E265:1:1:block comment should start with '# '
5 E501:1:80:line too long (152 > 79 characters)
6 E265:2:1:block comment should start with '# '
7 E265:3:1:block comment should start with '# '
8 E265:12:1:block comment should start with '# '
9 E265:16:1:block comment should start with '# '
10 E111:20:3:indentation is not a multiple of four
11 W292:20:11:no newline at end of file
12
13 Code
14 =====
15 #bash -c "$(curl -fsSL https://raw.githubusercontent.com/codio/install_software/master/tools/ansible.sh)" mysql echo "Mysql password root user password is 'codio'"
16 #pip install psutil
17 #python3 -m pip install psutil
18
19 import mysql.connector
20 mydb = mysql.connector.connect(
21     host="localhost",
22     user="root",
23     password="codio"
24 )
25
26 #Create ift database in mysql
27 mycursor = mydb.cursor()
28 mycursor.execute("create database if not exists ift_database")
29
30 #Print out all databases in mysql
31 mycursor.execute("show databases")
32 print('\n ** Current Databases are: ** \n+++++')
33 for x in mycursor:
34     print(x)
```

## 2.2 Lint output post corrections.

```
1 Check results
2 =====
3
4 E501:1:80:line too Long (153 > 79 characters)
5
6 Code
7 =====
8 # bash -c "$(curl -fsSL https://raw.githubusercontent.com/codio/install_software/master/tools/ansible.sh)" mysql echo "Mysql password root user password is 'codio'"
9 # pip install psutil
10 # python3 -m pip install psutil
11
12 import mysql.connector
13
14 mydb = mysql.connector.connect(
15     host="localhost",
16     user="root",
17     password="codio"
18 )
19
20 # Create ift_database in mysql
21 mycursor = mydb.cursor()
22 mycursor.execute("create database if not exists ift_database")
23
24 # Print out all databases in mysql
25 mycursor.execute("show databases")
26 print('\n ** Current Databases are: **: \n++++++ \n ')
27 for x in mycursor:
28     print(x)
29
```

## 3. Create Table(create\_tables.py)

### 3.1 Lint Output before corrections.

```
1 Check results
2 =====
3
4 E265:11:1:block comment should start with '# '
5 E501:12:80:line too Long (198 > 79 characters)
6 E265:13:1:block comment should start with '# '
7 E501:13:80:line too Long (197 > 79 characters)
8 W291:13:198:trailing whitespace
9 E501:14:80:line too Long (177 > 79 characters)
10 W291:14:178:trailing whitespace
11 E265:16:1:block comment should start with '# '
12 E111:20:3:indentation is not a multiple of four
13 E111:24:3:indentation is not a multiple of four
14 E265:26:1:block comment should start with '# '
15 E501:27:80:line too Long (91 > 79 characters)
16 E501:28:80:line too Long (96 > 79 characters)
17 E501:29:80:line too Long (103 > 79 characters)
18 E111:33:3:indentation is not a multiple of four
19 E265:36:1:block comment should start with '# '
20 E501:37:80:line too Long (141 > 79 characters)
21 E501:38:80:line too Long (125 > 79 characters)
22 E265:41:1:block comment should start with '# '
23 E111:45:3:indentation is not a multiple of four
24
```



```

25 Code
26
27 import mysql.connector
28 mydb = mysql.connector.connect(
29     host="localhost",
30     user="root",
31     password="codio",
32     database="ift_database"
33 )
34
35 mycursor = mydb.cursor()
36
37 #Create required tables to python script
38 mycursor.execute("create table if not exists users (id VARCHAR(4), password VARCHAR(30), name VARCHAR(255), title VARCHAR(30) default NULL, email VARCHAR(255), phone VARCHAR(12), PRIMARY KEY (id))")
39 mycursor.execute("create table if not exists testers (id VARCHAR(4), name VARCHAR(255), title VARCHAR(30) default NULL, doctor VARCHAR(255), date VARCHAR(7), time VARCHAR(255), PRIMARY KEY (id))")
40 mycursor.execute("create table if not exists secquestions (id VARCHAR(4), sec_question1 VARCHAR(255), sec_question2 VARCHAR(255), sec_question3 VARCHAR(255), PRIMARY KEY (id))")
41
42 #Print out the tables and columns details in users table
43 mycursor.execute("show tables")
44 print("\n TABLES: \n+++++\n ")
45 for x in mycursor:
46     print(x)
47 mycursor.execute("show columns from users")
48 print("\n patients Table: \n+++++\n ")
49 for x in mycursor:
50     print(x)
51
52 #create new user in ift database and give it limited permissions
53 mycursor.execute("create user if not exists 'ift'@'localhost' IDENTIFIED BY 'IFTasd@2021'")
54 mycursor.execute("grant delete,insert,select,update on ift_database.users to 'ift'@'localhost'")
55 mycursor.execute("grant delete,insert,select,update on ift_database.secquestions to 'ift'@'localhost'")
56 mycursor.execute("show grants for 'ift'@'localhost'")
57 print("\n ** If User GRANTIS** \n")
58 for x in mycursor:
59     print(x)
60
61
62 #Insert data in created tables to help in python script tests
63 mycursor.execute("insert into users (id,password,name,title,email,phone) values ('222','Admin@781','Admin','Admin','admin@ift.com','22222')")
64 mycursor.execute("insert into secquestions (id,sec_question1,sec_question2,sec_question3) values ('222','Doll','BMW','Red')")
65 mycursor.execute("flush privileges")
66
67 #Print out users table rows
68 mycursor.execute("select * from users")
69 print("\n ** Show Users Table **")
70 for x in mycursor:
71     print(x)
72

```

### 3.2 Lint output post corrections.

```

1  Check results
2  =====
3
4  E501:14:80:line too Long (185 > 79 characters)
5  E501:15:80:line too Long (107 > 79 characters)
6  E501:16:80:line too Long (92 > 79 characters)
7  E501:18:80:line too Long (164 > 79 characters)
8  E501:31:80:line too Long (91 > 79 characters)
9  E501:32:80:line too Long (96 > 79 characters)
10 E501:33:80:line too Long (103 > 79 characters)
11 E501:41:80:line too Long (128 > 79 characters)
12 E501:43:80:line too Long (112 > 79 characters)
13

```

```

14 Code
15 -----
16 import mysql.connector
17
18 mydb = mysql.connector.connect(
19     host="localhost",
20     user="root",
21     password="codio",
22     database="ift_database"
23 )
24
25 mycursor = mydb.cursor()
26
27 # Create required tables to python script
28 mycursor.execute(
29     "create table if not exists users (id VARCHAR(4), password VARCHAR(30), name VARCHAR(255), title VARCHAR(30) default NULL, email VARCHAR(255), phone VARCHAR(12), PRIMARY KEY (id))")
30 # mycursor.execute("create table if not exists testers (id VARCHAR(4), name VARCHAR(255), title VARCHAR(30)
31 # default NULL, doctor VARCHAR(255), date VARCHAR(7), time VARCHAR(255), PRIMARY KEY (id))")
32 mycursor.execute(
33     "create table if not exists secquestions (id VARCHAR(4), sec_question1 VARCHAR(255), sec_question2 VARCHAR(255), sec_question3 VARCHAR(255), PRIMARY KEY (id))")
34
35 # Print out the tables and columns details in users table
36 mycursor.execute("show tables")
37 print("\n TABLES: \n+++++ \n ')
38 for x in mycursor:
39     print(x)
40 mycursor.execute("show columns from users")
41 print('\n patients Table: \n+++++ \n ')
42 for x in mycursor:
43     print(x)
44
45 # create new user in ift_database and give it limited permissions
46 mycursor.execute("create user if not exists 'ift'@'localhost' IDENTIFIED BY 'IFTssd@2021'")
47 mycursor.execute("grant delete,insert,select,update on ift_database.users to 'ift'@'localhost'")
48 mycursor.execute("grant delete,insert,select,update on ift_database.secquestions to 'ift'@'localhost'")
49 mycursor.execute("show grants for 'ift'@'localhost'")
50 print("\n ** ift User GRANTS** \n')
51 for x in mycursor:
52     print(x)
53
54 # Insert data in created tables to help in python script tests
55 mycursor.execute(
56     "insert into users (id,password,name,title,email,phone) values ('222','Admin@78!','Admin','Admin','admin@ift.com','22222')")
57 mycursor.execute(
58     "insert into secquestions (id,sec_question1,sec_question2,sec_question3) values ('222','Doll','BMW','Red')")
59 mycursor.execute("flush privileges")
60
61 # Print out users table rows
62 mycursor.execute("select * from users")
63 print('\n ** Show Users Table **')
64 for x in mycursor:
65     print(x)

```

## 4. Register(register.py)

### 4.1 Lint Output before corrections.

```
1  Check results
2  =====
3
4  E211:1:21:whitespace before '('
5  E203:1:24:whitespace before ':'
6  E111:2:2:indentation is not a multiple of four
7  E262:2:19:inline comment should start with '#'
8  E111:3:2:indentation is not a multiple of four
9  E262:3:19:inline comment should start with '#'
10 E501:3:80:line too Long (97 > 79 characters)
11 E111:4:2:indentation is not a multiple of four
12 W291:4:24:trailing whitespace
13 E265:5:1:block comment should start with '#'
14 E501:5:80:line too Long (88 > 79 characters)
15 E111:6:2:indentation is not a multiple of four
16 W291:6:33:trailing whitespace
17 E265:7:4:block comment should start with '#'
18 E111:14:2:indentation is not a multiple of four
19 E262:14:15:inline comment should start with '#'
20 E111:15:3:indentation is not a multiple of four
21 E111:16:3:indentation is not a multiple of four
22 E111:17:3:indentation is not a multiple of four
23 E262:17:53:inline comment should start with '#'
24 E501:17:80:line too Long (101 > 79 characters)
25 E111:18:3:indentation is not a multiple of four
26 E111:19:3:indentation is not a multiple of four
27 E111:20:3:indentation is not a multiple of four
28 E111:22:3:indentation is not a multiple of four
29 E111:23:3:indentation is not a multiple of four
30 E111:24:3:indentation is not a multiple of four
31 E262:24:38:inline comment should start with '#'
32 E501:24:80:line too Long (136 > 79 characters)
33 E111:26:3:indentation is not a multiple of four
34 E261:26:27:at least two spaces before inline comment
35 E262:26:28:inline comment should start with '#'
36 W293:27:1:blank line contains whitespace
37 E111:28:2:indentation is not a multiple of four
38 E261:29:63:at least two spaces before inline comment
39 E262:29:64:inline comment should start with '#'
40 E501:29:80:line too Long (100 > 79 characters)
41 E202:30:33:whitespace before ')'
42 E501:30:80:line too Long (207 > 79 characters)
43 E114:31:7:indentation is not a multiple of four (comment)
44 E265:31:7:block comment should start with '#'
45 E111:32:7:indentation is not a multiple of four
46 E501:33:80:line too Long (153 > 79 characters)
47 E262:33:107:inline comment should start with '#'
48 E111:35:2:indentation is not a multiple of four
49 E262:35:18:inline comment should start with '#'
50 E111:36:3:indentation is not a multiple of four
51 E111:37:3:indentation is not a multiple of four
52 E262:37:20:inline comment should start with '#'
```



```
53 E111:39:3:indentation is not a multiple of four
54 W293:40:1:blank line contains whitespace
55 W293:41:1:blank line contains whitespace
56 E111:43:2:indentation is not a multiple of four
57 E303:43:2:too many blank lines (3)
58 E111:44:3:indentation is not a multiple of four
59 E111:45:3:indentation is not a multiple of four
60 E231:45:24:missing whitespace after ','
61 E265:46:5:block comment should start with '#'
62 E111:48:3:indentation is not a multiple of four
63 E111:50:2:indentation is not a multiple of four
64 E111:51:3:indentation is not a multiple of four
65 E111:52:3:indentation is not a multiple of four
66 E111:53:3:indentation is not a multiple of four
67 E111:54:3:indentation is not a multiple of four
68 W291:54:59:trailing whitespace
69 E115:55:1:expected an indented block (comment)
70 E265:55:1:block comment should start with '#'
71 E111:57:3:indentation is not a multiple of four
72 E111:58:2:indentation is not a multiple of four
73 E111:59:2:indentation is not a multiple of four
74 E501:59:80:line too long (116 > 79 characters)
75 E111:60:2:indentation is not a multiple of four
76 E111:61:2:indentation is not a multiple of four
77 E262:61:50:inline comment should start with '#'
78 E501:61:80:line too long (106 > 79 characters)
79 E111:62:2:indentation is not a multiple of four
80 E111:63:2:indentation is not a multiple of four
81 E262:63:49:inline comment should start with '#'
82 E302:65:1:expected 2 blank lines, found 1
83 E211:65:18:whitespace before '('
84 E203:65:21:whitespace before ':'
85 E111:66:2:indentation is not a multiple of four
86 E262:66:19:inline comment should start with '#'
87 E111:67:2:indentation is not a multiple of four
88 E262:67:19:inline comment should start with '#'
89 E501:67:80:line too long (97 > 79 characters)
90 W291:67:98:trailing whitespace
91 E114:68:3:indentation is not a multiple of four (comment)
92 E116:68:3:unexpected indentation (comment)
93 E265:68:3:block comment should start with '#'
94 E501:68:80:line too long (90 > 79 characters)
95 E111:69:2:indentation is not a multiple of four
96 W291:69:24:trailing whitespace
97 E111:70:2:indentation is not a multiple of four
98 E265:71:4:block comment should start with '#'
99 W293:77:1:blank line contains whitespace
100 E111:78:2:indentation is not a multiple of four
101 E262:78:19:inline comment should start with '#'
102 E111:79:3:indentation is not a multiple of four
103 E111:80:3:indentation is not a multiple of four
104 E111:81:3:indentation is not a multiple of four
```



```
105 E111:82:3:indentation is not a multiple of four
106 E111:83:3:indentation is not a multiple of four
107 E111:84:3:indentation is not a multiple of four
108 E111:86:3:indentation is not a multiple of four
109 E111:87:3:indentation is not a multiple of four
110 E111:88:3:indentation is not a multiple of four
111 E501:88:80:line too Long (106 > 79 characters)
112 E111:90:3:indentation is not a multiple of four
113 W293:91:1:blank line contains whitespace
114 E111:92:2:indentation is not a multiple of four
115 E115:93:1:expected an indented block (comment)
116 E265:93:1:block comment should start with '#'
117 E501:93:80:line too Long (138 > 79 characters)
118 E202:95:33:whitespace before ')'
119 E501:95:80:line too Long (207 > 79 characters)
120 E111:96:7:indentation is not a multiple of four
121 E501:97:80:line too Long (104 > 79 characters)
122 W293:98:1:blank line contains whitespace
123 E111:99:2:indentation is not a multiple of four
124 E111:102:7:indentation is not a multiple of four
125 W293:104:1:blank line contains whitespace
126 E111:105:2:indentation is not a multiple of four
127 E115:106:1:expected an indented block (comment)
128 E265:106:1:block comment should start with '#'
129 W291:106:65:trailing whitespace
130 E111:107:3:indentation is not a multiple of four
131 E111:108:3:indentation is not a multiple of four
132 E231:108:16:missing whitespace after ','
133 E111:109:3:indentation is not a multiple of four
134 E111:111:3:indentation is not a multiple of four
135 E111:113:2:indentation is not a multiple of four
136 E231:115:26:missing whitespace after ','
137 E115:116:1:expected an indented block (comment)
138 E265:116:1:block comment should start with '#'
139 E111:117:7:indentation is not a multiple of four
140 E111:120:2:indentation is not a multiple of four
141 W293:121:1:blank line contains whitespace
142 E111:122:3:indentation is not a multiple of four
143 E111:123:3:indentation is not a multiple of four
144 E111:124:3:indentation is not a multiple of four
145 E111:125:3:indentation is not a multiple of four
146 E115:126:1:expected an indented block (comment)
147 E265:126:1:block comment should start with '#'
148 E111:128:3:indentation is not a multiple of four
149 E111:129:2:indentation is not a multiple of four
150 E111:130:2:indentation is not a multiple of four
151 E501:130:80:line too Long (100 > 79 characters)
152 E111:131:2:indentation is not a multiple of four
153 E111:132:2:indentation is not a multiple of four
154 E262:132:48:inline comment should start with '#'
155 E501:132:80:line too Long (91 > 79 characters)
156 E265:133:1:block comment should start with '#'
```

```

157 E501:133:80:line too long (117 > 79 characters)
158 E111:134:2:indentation is not a multiple of four
159 E111:135:2:indentation is not a multiple of four
160 E111:138:7:indentation is not a multiple of four
161 W293:140:1:blank line contains whitespace
162 E111:141:2:indentation is not a multiple of four
163 E111:144:7:indentation is not a multiple of four
164 W293:146:1:blank line contains whitespace
165 E111:147:2:indentation is not a multiple of four
166 E111:150:7:indentation is not a multiple of four
167 E111:152:2:indentation is not a multiple of four
168 E111:153:2:indentation is not a multiple of four
169 E501:153:80:line too long (107 > 79 characters)
170 E111:154:2:indentation is not a multiple of four
171 E111:155:2:indentation is not a multiple of four
172 E262:155:40:inline comment should start with '#'
173 E501:155:80:line too long (86 > 79 characters)
174 E111:156:2:indentation is not a multiple of four
175 E111:157:2:indentation is not a multiple of four
176 E262:157:17:inline comment should start with '#'
177 W293:161:1:blank line contains whitespace
178 W391:161:1:blank line at end of file
179

```

```

180 Code
181 -----
182 def Testers.register () :
183     import getpass #import getpass library to help in hide password echoing
184     import re #import A regular expression (or RE) specifies a set of strings that matches it
185     import mysql.connector
186     #import mysql.connector to help in connectivity between python script and mysql database
187     mydb = mysql.connector.connect(
188         #mysql parameters to establish the connection with ift database mysql
189         host="localhost",
190         user="ift",
191         password="IfTsd@2021",
192         database="ift_database"
193     )
194
195     while True: #Run while loop to allow Admin to add the id.
196         user_id = int(input("\nEnter your ID: "))
197         idcursor = mydb.cursor()
198         idcheck = "select id from users where id = %s" #call select from the users table in ift_database
199         val2 = [user_id]
200         idcursor.execute(idcheck, val2)
201         for x in idcursor:
202             print(x)
203         z = idcursor.rowcount
204         print(z)
205         if ((z == -1) and (user_id > 0)): #check if the id exists in the users table and avoid any double entry - id should be greater than 0
206             break
207         print("Try another ID!") #Ask the Admin to re enter the correct id.
208
209     while True:
210         user_password = getpass.getpass("\nEnter your Password: ") #getpass to hide any password echoing
211         if ((8 <= len(user_password) ) and (re.search(r'[A-Z]', user_password)) and (re.search(r'[a-z]', user_password)) and (re.search(r'[0-9]', user_password)) and (re.search(r'[!@#$%^&*~]', user_password))):
212             #use re to check the password is strong enough and greater or equal 8
213             break
214         print('The password must be more than 8 and include:\n UPPER,lower,numbers and special characters') #ask the Admin to re enter the correct password
215
216     while True: #while loop for enter user name
217         user_name = input("\nEnter your Name: ")
218         if user_name: #to avoid any empty entry from user
219             break
220         print("Please Enter your name - can not be empty")
221
222
223
224     while True:
225         user_email = input("\nEnter your Email : ")
226         if ((re.search(r'[@]', user_email)) and (re.search(r'[.]', user_email))):
227             #use 'if' to check the input email from user has '@' and '.'
228             break
229         print("your email is wrong")
230

```



```

231 while True:
232     user_phone = input("\nEnter your Phone: ")
233     phonelist = list(user_phone)
234     b = phonelist[3]
235     if ((11 == len(user_phone)) and (b == ['0', '2', '0'])):
236         #Phone number must be 11 numbers and starts with 020
237         break
238     print("your number is wrong")
239     mycursor = mydb.cursor()
240     sql = "INSERT INTO users (id, password, name, address, postcode, email, phone) VALUES (%s, %s, %s, %s, %s, %s, %s)"
241     val = [user_id, user_password, user_name, user_email, user_phone]
242     mycursor.execute(sql, val) #insert the patient inputs in users table in ift database
243     print(mycursor.rowcount, "Record inserted successfully into Patient Table")
244     mydb.commit() #commit the mysql data entry
245
246 def User_register(): # use user_register function to register Admin users
247     import getpass #getpass library to hide password echoing
248     import re #import A regular expression (or RE) specifies a set of strings that matches it
249     #import mysql.connector to help in connectivity between python script and mysql database
250     import mysql.connector
251     mydb = mysql.connector.connect(
252         #mysql parameters to establish the connection with ift_database mysql
253         host="localhost",
254         user="ift",
255         password="IFTssd2021",
256         database="ift_database"
257     )
258
259 while True: #Run while loop to allow user to add the id.
260     user_id = int(input("\nEnter your ID: "))
261     idcursor = mydb.cursor()
262     idcheck = "SELECT id FROM users WHERE id = %s"
263     val2 = [user_id]
264     idcursor.execute(idcheck, val2)
265     for x in idcursor:
266         print(x)
267     z = idcursor.rowcount
268     print(z)
269     if ((z == -1) and (user_id > 99)): # Admin id must be greater than 99 and can not be empty
270         break
271     print("Try another ID!")
272
273 while True:
274     #use re to check the password is strong enough and at least 8 character, password must contain number, UPPER, lower, and special character
275     user_password = getpass.getpass("\nEnter your Password: ")
276     if ((8 <= len(user_password)) and (re.search(r'[A-Z]', user_password)) and (re.search(r'[a-z]', user_password)) and (re.search(r'[0-9]', user_password)) and (re.search(r'[_!@#$%^&*]', user_password))):
277         break
278     print("The password must be more than 8 and include:\n UPPER,lower,numbers and special characters")

```

```

279
280 while True:
281     user_name = input("\nEnter your Name : ")
282     if user_name:
283         break
284     print("Please Enter your name - can not be empty")
285
286 while True:
287     #Admin should select if new user is Tester or Admin in the title
288     user_title = input("\nEnter user title : (Tester or Admin)")
289     u = ['Tester', 'Admin']
290     if user_title in u:
291         break
292     print("Please enter correct title")
293
294 while True:
295     user_email = input("\nEnter your Email : ")
296     if ((re.search(r'@', user_email)) and (re.search(r'[.]', user_email))):
297         #Check if the entered email contain @ and "."
298         break
299     print("your email is wrong")
300
301 while True:
302     user_phone = input("\nEnter your Phone: ")
303     phonelist = list(user_phone)
304     b = phonelist[3]
305     if ((11 == len(user_phone)) and (b == ['0', '2', '0'])):
306         #Phone number must be 11 numbers and starts with 020
307         break
308     print("your number is wrong")
309     mycursor = mydb.cursor()
310     sql = "INSERT INTO users (id, password, name, title, email, phone) VALUES (%s, %s, %s, %s, %s, %s, %s)"
311     val = [user_id, user_password, user_name, user_title, user_email, user_phone]
312     mycursor.execute(sql, val) #execute the entered data in the users table
313     #Ask to answer three security questions to use in user login verifications - can replaced with Biometric check system.
314     print("\n***Please answer followings security questions***")
315     while True:
316         question1 = input("\nWhat is your pet name? ")
317         if question1:
318             break
319         print("Enter your answer")
320
321 while True:
322     question2 = input("\nWhat is your first car? ")
323     if question2:
324         break
325     print("Enter your answer")
326
327

```

```

328 while True:
329     question3 = input("\n What is your favourite colour? ")
330     if question3:
331         break
332     print("Enter your answer")
333     mycursor = mydb.cursor()
334     sec = "insert into secquestions (id, sec_question1, sec_question2, sec_question3) values (%s, %s, %s, %s)"
335     secval = [user_id, question1, question2, question3]
336     mycursor.execute(sec, secval) #execute security answers in secquestions table
337     print(mycursor.rowcount, "Record inserted successfully into users Table")
338     mydb.commit() #commit the entry in the ift_database
339

```



## 4.2 Lint output post corrections.

```
1 Check results
2 =====
3
4 Code
5 =====
6 def Testers_register():
7     import getpass # library to help in hiding password echoing.
8     import re # A regex specifies a set of strings that matches it.
9     import mysql.connector
10    # mysql.connector : connectivity between python script and database.
11    mydb = mysql.connector.connect(
12        # mysql parameters to establish the connection with ift_database mysql
13        host="localhost",
14        user="ift",
15        password="IFTssd@2021",
16        database="ift_database"
17    )
18
19
20 while True: # Run while loop to allow Admin to add the id.
21     user_id = int(input("\nEnter your ID: "))
22     idcursor = mydb.cursor()
23     idcheck = "select id from users where id = %s" # select id in user table .
24     val2 = [user_id]
25     idcursor.execute(idcheck, val2)
26     for x in idcursor:
27         print(x)
28     z = idcursor.rowcount
29     print(z)
30     if ((z == -1) and (
31         user_id > 99)): # check if the id exists in the users table
32         # and avoid any double entry -id should be greater than 9
33         break
34     print("Try another ID") # Ask the Admin to re enter the correct id.
35
36
37 while True:
38     user_password = getpass.getpass("\nEnter your Password: ")
39     if ((8 <= len(user_password)) and (re.search(r'[A-Z]', user_password)) and
40         (re.search(r'[a-z]', user_password)) and
41         (re.search(r'[0-9]', user_password)) and
42         (re.search(r'[_!@#$%^&*~]', user_password))):
43         # use re to check the password is strong enough and greater or equal 8
44         break
45     print(
46         'The password must be more than 8 and include:'
47         '\n UPPER,lower,numbers and special characters')
48
49
50 while True: # while loop for enter user name
51     user_name = input("\nEnter your Name: ")
52     if user_name: # validate user input.
53         break
54     print('Please Enter your name - can not be empty')
55
56
57 while True:
58     user_email = input("\nEnter your Email: ")
59     if ((re.search(r'@', user_email)) and (re.search(r'[.]', user_email))):
60         # use 'if' to check the input email from user has '@' and '.'.
61         break
62     print('your email is wrong')
63
64
65 while True:
66     user_phone = input("\nEnter your Phone: ")
67     phonenumber = list(user_phone)
68     b = phonenumber[3]
69     if ((11 == len(user_phone)) and (b == ['0', '2', '0 ])):
70         # Phone number must be 11 numbers and starts with 020
71         break
72     print('your number is wrong')
73     mycursor = mydb.cursor()
74     sql = "INSERT INTO users (id, password, name, address, postcode, email, phone) VALUES (%s, %s, %s, %s, %s, %s, %s)"
75     val = [user_id, user_password, user_name, user_email, user_phone]
76     mycursor.execute(sql, val) # insert the patient inputs in users table in ift database
77     print(mycursor.rowcount, "Record inserted successfully into Patient Table")
78     mydb.commit() # commit the mysql data entry
79
80 def User_register(): # user_register function to register Admin users
81     import getpass # getpass library to hide password echoing
82     import re # A regex specifies a set of strings that matches it.
83     # mysql parameters to establish the connection with ift_database mysql
84     import mysql.connector
85     mydb = mysql.connector.connect(
86         # mysql parameters to establish the connection with ift_database mysql
87         host="localhost",
88         user="ift",
89         password="IFTssd@2021",
90         database="ift_database"
91     )
92
93 while True: # Run while loop to allow user to add the id.
94     user_id = int(input("\nEnter your ID: "))
95     idcursor = mydb.cursor()
96     idcheck = "SELECT id FROM users WHERE id = %s"
97     val2 = [user_id]
98     idcursor.execute(idcheck, val2)
99     for x in idcursor:
100         print(x)
101     z = idcursor.rowcount
102     print(z)
103     if ((z == -1) and (user_id > 199)): # Admin id must
104         # be greater than 99 and can not be empty
105         break
106     print("Try another ID")
107
108
109 while True:
110     # use re to check the password is strong enough
111     # and at least 8 character, password must contain number,
112     # UPPER, lower, and special character
113     user_password = getpass.getpass("\nEnter your Password: ")
114     if ((8 <= len(user_password)) and (re.search(r'[A-Z]', user_password)) and
115         (re.search(r'[a-z]', user_password)) and
116         (re.search(r'[0-9]', user_password)) and
117         (re.search(r'[_!@#$%^&*~]', user_password))):
118         break
119     print('The password must be more than 8 and include:'
120         '\n UPPER,lower,numbers and special characters')
121
122 while True:
123     user_name = input("\nEnter your Name: ")
124     if user_name:
125         break
126     print('Please Enter your name - can not be empty')
127
128 while True:
129     # Admin should select if new user is Tester or Admin in the title
130     user_title = input("\nEnter user title: (Tester or Admin)")
131     u = ['Tester', 'Admin']
132     if user_title in u:
133         break
134     print('Please enter correct title')
135
136 while True:
137     user_email = input("\nEnter your Email: ")
138     if ((re.search(r'@', user_email)) and (re.search(r'[.]', user_email))):
139         # check if the entered email contain '@' and '.'.
140         break
141     print('your email is wrong')
142
```

```

143 while True:
144     user_phone = input("\nEnter your Phone: ")
145     phonelist = list(user_phone)
146     b = phonelist[3]
147     if ((11 == len(user_phone)) and (b == ['0', '2', '9'])):
148         # Phone number must be 11 numbers and starts with 020
149         break
150     print('your number is wrong')
151
152 mycursor = mydb.cursor()
153 sql = "INSERT INTO users (id, password, name, title, email, phone) " \
154       "VALUES (%s, %s, %s, %s, %s, %s)"
155 val = [user_id, user_password, user_name, user_title, user_email, user_phone]
156 mycursor.execute(sql, val) # execute input data in the users table
157 # Ask to answer three security questions to use in user login verifications
158 # - can replaced with Biometric check system
159 print("\n**Please answer followings security questions**")
160
161 while True:
162     question1 = input("\nWhat is your pet name? ")
163     if question1:
164         break
165     print('Enter your answer')
166
167 while True:
168     question2 = input("\nWhat is your first car? ")
169     if question2:
170         break
171     print('Enter your answer')
172
173 while True:
174     question3 = input("\nWhat is your favourite colour? ")
175     if question3:
176         break
177     print('Enter your answer')
178
179 mycursor = mydb.cursor()
180 sec = "insert into secquestions (id, sec_question1, sec_question2, " \
181       "sec_question3) values (%s, %s, %s, %s)"
182 secval = [user_id, question1, question2, question3]
183 mycursor.execute(sec, secval) # execute security answers in secquestions table
184 print(mycursor.rowcount, "Record inserted successfully into users Table")
185 mydb.commit() # commit the entry in the ift_database
186

```