

PROJECT DEVELOPMENT PHASE

Database DB2

SPRINT-2

<i>DATE</i>	<i>05 October 2022</i>
<i>Team ID</i>	<i>PNT2022TMID24347</i>
<i>Project name</i>	<i>Customer Care Registry</i>

SQL Query to create table:

– complaint Table

```
CREATE TABLE complaints(  
  id INT NOT NULL AUTO_INCREMENT,  
  username TEXT NOT NULL, email TEXT NOT NULL,  
  against_person TEXT NOT NULL,  
  des TEXT NOT NULL, date TEXT NOT NULL,  
  solved TEXT NOT NULL,  
  PRIMARY KEY (complaint_id)  
);
```

– customer details Table

```
CREATE TABLE customerdeatils (  
  id INT NOT NULL AUTO_INCREMENT,  
  name TEXT NOT NULL,  
  email TEXT NOT NULL,  
  passwrđ TEXT NOT NULL,  
  PRIMARY KEY (id)  
);
```

-- Insert data into customer table

```
INSERT INTO customerdeatils VALUES (null, 'Clark', 'abc@abc.com', 'abc');  
INSERT INTO customerdeatils VALUES (null, 'Clark2', 'abc2@abc.com', 'abc');  
INSERT INTO customerdeatils VALUES (null, 'Clark3', 'abc3@abc.com', 'abc');
```

-- Insert data into customer table

```
INSERT INTO complaints VALUES ( 'Clark', 'kent' , 'abc@abc.com', 'abc','water','0');
```

```
INSERT INTO complaints VALUES ( 'Clark', 'kent' , 'abc@abc.com', 'abc','water','1');
```

-- Fetch data from customer table

```
SELECT * FROM customerdeatils;
```

– Fetch data from Complaints

```
SELECT * FROM Complaints;
```

IBM cloud Db2:

The screenshot displays the IBM Db2 on Cloud management console. The top navigation bar includes 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is active, showing a list of tables in the 'DMT13873' schema: 'COMPLAINTS' and 'CUSTOMERDEATILS'. The 'Table definition' panel for the 'COMPLAINTS' table is open, showing the following columns:

Name	Data type	Nullable	Length	Scale
ID	INTEGER	N		0
USERNAME	VARCHAR	N	150	0
EMAIL	VARCHAR	N	150	0
AGAINST_PERSON	VARCHAR	N	150	0
DEF	VARCHAR	N	150	0

A 'View data' button is visible at the bottom of the table definition panel. The console also shows a search bar for schemas or tables and a 'Refresh' button.

DMT13873.COMPLAINTS

Back



Export to CSV



ID	USERNAME	EMAIL	AGAINST_PERSON	DES	DATE	SOLVED
1	ranjith	ranjith.2k01@gmail.com	walter	sdsds	12/11/2022	1
3	ranjith	ranjith.2k01@gmail.com	BSNL	sadsad	13/11/2022	0
4	ranjith	ranjith.2k01@gmail.com	BSNL	sfdnfjdngn	14.11.2022	0
5	ranjith	ranjith.2k01@gmail.com	BSNL	klkljkh	14.11.2022	0

DMT13873.CUSTOMERDEATILS

Back



Export to CSV



ID	USERNAME	EMAIL	PASSWRD
1	ranjith	ranjith.2k01@gmail.com	abc
2	summa	19i340@psgtech.ac.in	abc
3	ranjith	rakshan486@gmail.com	abc
4	shankar	palaniarthi123@gmail.com	Palanivel1

DataBase Schema

Table definition



COMPLAINTS

No statistics available.

Name	Data type	Nullable	Length	Scale	
ID	INTEGER	N		0	
USERNAM E	VARCHAR	N	150	0	
EMAIL	VARCHAR	N	150	0	
AGAINST_ PERSON	VARCHAR	N	150	0	
DES	VARCHAR	N	150	0	

Table definition



COMPLAINTS

No statistics available.

Name	Data type	Nullable	Length	Scale	
AGAINST_PERSON	VARCHAR	N	150	0	👁
DES	VARCHAR	N	150	0	👁
DATE	VARCHAR	N	150	0	👁
SOLVED	VARCHAR	N	150	0	👁

Table definition



CUSTOMERDEATILS

No statistics available.

Name	Data type	Nullable	Length	Scale	
ID	INTEGER	N		0	👁
USERNAME	VARCHAR	N	150	0	👁
EMAIL	VARCHAR	N	150	0	👁
PASSWRD	VARCHAR	N	150	0	👁

Connection configuration resources

Host name: b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud

With SSL: Yes

Port number: 31249

Database name: bludb

User ID: <user name>

Password: *****

Version: Compatible with Db2, Version 11.5.0 or later

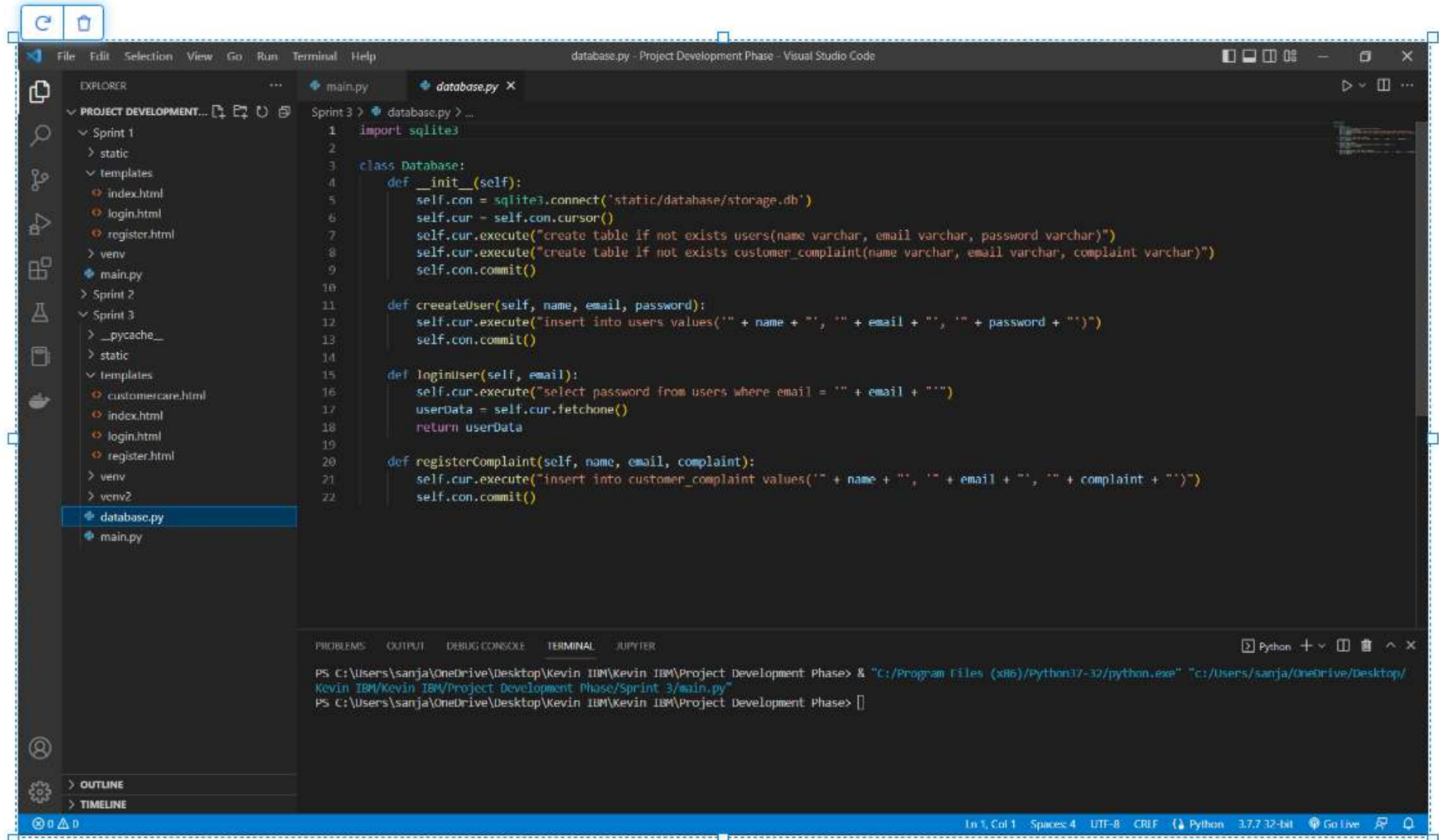
[Download SSL Certificate](#)

Connection to Db2:

```
4 # dsn_hostname = "b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud"
5 # dsn_uid = "dmt13873"
6 # dsn_pwd = "740yZ1Yq8Uj2E4qm"
7 # dsn_database = 'bludb'
8 # dsn_port = 31249
9 conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=b0aebb68-94fa-46ec-a1fc-1c999edb6187.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=31249;SE
10 print(conn)
11 print("connection successful...")
```

```
query = "SELECT * FROM customerdeatils WHERE email=? AND passwr=?"
stmt = ibm_db.prepare(conn, query) # type:ignore
ibm_db.bind_param(stmt,1,mail) # type:ignore
ibm_db.bind_param(stmt,2,password) # type:ignore
ibm_db.execute(stmt) # type:ignore
user = ibm_db.fetch_assoc(stmt) # type:ignore
print(user,password)
```

Database.py



The screenshot displays the Visual Studio Code interface with a Python project named "Project Development Phase". The Explorer sidebar on the left shows the project structure, including folders for "Sprint 1", "Sprint 2", and "Sprint 3", and files like "index.html", "login.html", "register.html", "main.py", and "database.py". The "database.py" file is selected and open in the editor.

The code in "database.py" defines a `Database` class that interacts with a SQLite database. The class methods include `__init__` for connecting to the database, `createUser` for inserting new users, `loginUser` for retrieving user data, and `registerComplaint` for inserting complaints.

```
1 import sqlite3
2
3 class Database:
4     def __init__(self):
5         self.con = sqlite3.connect('static/database/storage.db')
6         self.cur = self.con.cursor()
7         self.cur.execute("create table if not exists users(name varchar, email varchar, password varchar)")
8         self.cur.execute("create table if not exists customer_complaint(name varchar, email varchar, complaint varchar)")
9         self.con.commit()
10
11     def createUser(self, name, email, password):
12         self.cur.execute("insert into users values('" + name + "', '" + email + "', '" + password + "')")
13         self.con.commit()
14
15     def loginUser(self, email):
16         self.cur.execute("select password from users where email = '" + email + "'")
17         userData = self.cur.fetchone()
18         return userData
19
20     def registerComplaint(self, name, email, complaint):
21         self.cur.execute("insert into customer_complaint values('" + name + "', '" + email + "', '" + complaint + "')")
22         self.con.commit()
```

The bottom of the interface shows the TERMINAL panel with the command prompt output, indicating the successful execution of the Python script.

```
PS C:\Users\sanja\OneDrive\Desktop\Kevin IBM\Kevin IBM\Project Development Phase> & "C:/Program Files (x86)/Python37-32/python.exe" "c:/Users/sanja/OneDrive/Desktop/
Kevin IBM/Kevin IBM/Project Development Phase/Sprint 3/main.py"
PS C:\Users\sanja\OneDrive\Desktop\Kevin IBM\Kevin IBM\Project Development Phase>
```