

# Create IBM DB2 and Connect with Python

Team ID	PNT2022TMID11592
Project Name	Project - Skill and Job Recommender

The screenshot shows the IBM Cloud dashboard for a Db2-8i instance. The top navigation bar includes the IBM Cloud logo, a search bar, and links to Catalog, Manage, and the user's account (Loganayagi P's Account). The main content area is titled "Db2-8i" and shows the instance is "Active". A sidebar on the left lists "Manage" options: "Getting started", "Service credentials", and "Connections". The main content area has two sections: "Getting started" with instructions on finding credentials and a "Go to UI" button, and "Need help?" with a "Support case" button. The bottom of the screen shows a Windows taskbar with various application icons and the system clock.

The screenshot shows the IBM Db2 on Cloud console. The top navigation bar includes the "IBM Db2 on Cloud" logo and a search bar. The main content area is titled "Schemas" and shows a table with one row: "RPV66344" of type "User" with 7 tables. The table has columns for "Name", "Type", and "Tables". The bottom of the screen shows a Windows taskbar with various application icons and the system clock.

Name	Type	Tables
RPV66344	User	7

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

Find schemas or tables Refresh

**Tables** New table

Name	Schema	Properties
AI_JOBS	RPV66344	...
APPLICANT	RPV66344	...
APPLY_JOBS	RPV66344	...
DATA_SCI_JOBS	RPV66344	...
FULLSTACK_JOBS	RPV66344	...
RECRUITER_INFO	RPV66344	...

Total: 7, selected: 0

**Table definition** APPLICANT Approximate -1 rows (4.00 MB) Updated on

Name	Data type	Nullable	Length	Scale
PHOTO	BLOB	Y	1048576	0
FNAME	VARCHAR	Y	32	0
LNAME	VARCHAR	Y	32	0
DOB	VARCHAR	Y	32	0

View data

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

RPV66344.APPLICANT Back

Export to CSV

PHOTO	FNAME	LNAME	DOB	GENDER	EMAIL	PASSWORD	PHONE	ADDRESS	RESUME	HIGHEST_QUAL	DEGREE
06e0061 0079006 1006700 69002e0 070006e 006700	Loganaya gi	P	2002- 02-18	3	ploganay agi@gmail. il.com	18022002	8765436 789	Alagarkoil,M adurai	90061006 70069005f 00720065 00730075 006d0065 002e0070 00640066 00	2	2
7000720 0610074 0068007 5002e00 6a00700 0650067	PRATHU	J S	2001- 11-11	3	prathush adevi200 1@gmail. com	11112001	9008078 654	Madurai	70007200 61007400 68007500 50072006 50073007 5006d006 5002e007	4	3

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

RPV66344.WEB\_DEV\_JOBS

Back

Export to CSV

COMP_NAME	POSITION	LOCATION	DEGREE	JOB_TYPE	TECH_AREA	EXPERIENCE	JOB_DESC
Albot software	Web developer	2	2	2	2	2	Should be responsible for the design and construction of websites. Must ensure that sites meet user expectations by ensuring they look good, run smoothly and offer easy access points with no loading issues between pages or error messages.
Albot software	Web developer	2	2	2	2	2	Should be responsible for the design and construction of websites. Must ensure that sites meet user expectations by ensuring they look good, run smoothly and offer easy access points with no loading issues between pages or error messages.

Type here to search

23:24 24-11-2022

```

1 from flask import Flask, render_template, redirect, request, session
2 import ibm_db, re
3 import smtplib
4 import sendgrid
5 import os
6 from sendgrid import SendgridAPIClient
7 from sendgrid.helpers.mail import Mail, Email, To, Content
8 #SUBJECT = "Interview Call"
9 #s = smtplib.SMTP('smtp.gmail.com', 587)
10
11 app = Flask(__name__)
12 app.secret_key = "a"
13
14 conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-8fc0-d6a8c9f7a0ef.c1ogj3sd0gtu0lqde00.databases.appdomain.cloud;PORT=31321;SECURITY=SSL;SSL_VERIFY_REQUIRED=false;SSL_KEYPATH=/etc/ssl/certs/ssl-cert-snakeoil.pem;SSL_CERTPATH=/etc/ssl/certs/ssl-cert-snakeoil.pem;UID=root;PWD=root;ENCIPHERMENT=NONE")
15
16
17 #app.config['SECRET_KEY'] = 'top-secret!'
18 #app.config['MAIL_SERVER'] = 'smtp.sendgrid.net'
19 #app.config['MAIL_PORT'] = 587
20 #app.config['MAIL_USE_TLS'] = True
21 #app.config['MAIL_USERNAME'] = 'apikey'
22 #app.config['MAIL_PASSWORD'] = os.environ.get('SG_RK3hMHPwQcmICFXxPQIGw.1lLaiz2SHuuzICBzuVVBtd5WahT7GQx9u3_xdTvYtHQ')
23 #app.config['MAIL_DEFAULT_SENDER'] = os.environ.get('imbskillsandjob@gmail.com')
24 #mail = Mail(app)
25
26 @app.route('/')
27 def home():
28     return render_template('home.html')
29
30 @app.route('/learning_module', methods=["GET", "POST"])
31 def learning_module():
32     return render_template('learning_module.html')
33
34 @app.route('/applicants_list', methods=["GET", "POST"])
35 def applicants_list():
36     return render_template('applicants_list.html')
37

```

Ln 14, Col 143 Spaces: 4 UTF-8 LF Python 3.10.8 64-bit

This screenshot shows a code editor with a Python script for user registration. The script uses a database connection to check if an email exists and to insert a new user if it doesn't. The user data is collected from a form and stored in a database table named 'APPLICANT'.

```
111 sql="SELECT * FROM APPLICANT WHERE EMAIL=?"
112 stmt=ibm_db.prepare(conn,sql)
113 ibm_db.bind_param(stmt,1,email)
114 ibm_db.execute(stmt)
115 account = ibm_db.fetch_assoc(stmt)
116 print(account)
117 if account:
118     msg = "Account already exists!"
119 elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
120     msg = "Invalid Email Address."
121 else:
122     insert_sql = "INSERT INTO APPLICANT VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)"
123     prep_stmt = ibm_db.prepare(conn,insert_sql)
124     ibm_db.bind_param(prep_stmt,1,photo)
125     ibm_db.bind_param(prep_stmt,2,fname)
126     ibm_db.bind_param(prep_stmt,3,lname)
127     ibm_db.bind_param(prep_stmt,4,dob)
128     ibm_db.bind_param(prep_stmt,5,gender)
129     ibm_db.bind_param(prep_stmt,6,email)
130     ibm_db.bind_param(prep_stmt,7,password)
131     ibm_db.bind_param(prep_stmt,8,phone)
132     ibm_db.bind_param(prep_stmt,9,address)
133     ibm_db.bind_param(prep_stmt,10,resume)
134     ibm_db.bind_param(prep_stmt,11,highest_qual)
135     ibm_db.bind_param(prep_stmt,12,degree)
136     ibm_db.bind_param(prep_stmt,13,branch)
137     ibm_db.bind_param(prep_stmt,14,tenth)
138     ibm_db.bind_param(prep_stmt,15,twelfth)
139     ibm_db.bind_param(prep_stmt,16,ug_cgpa)
140     ibm_db.bind_param(prep_stmt,17,ug_percent)
141     ibm_db.bind_param(prep_stmt,18,diploma)
142     ibm_db.bind_param(prep_stmt,19,skillset)
143
144     ibm_db.execute(prep_stmt)
145     msg = "You have successfully registered."
146     to_email = To(email)
147 #
```

This screenshot shows a code editor with a Python script for job application. The script uses a database connection to insert a new job application. The user data is collected from a form and stored in a database table named 'APPLY\_JOBS'.

```
407 twelfth = request.form['twelfth']
408 domain = request.form['domain']
409 ug_percent = request.form['ug_percent']
410 email = request.form['email']
411 phone = request.form['phone']
412 resume = request.form['resume']
413 comp_name = request.form['comp_name']
414 position = request.form['position']
415
416 sql="INSERT INTO APPLY_JOBS VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)"
417 stmt=ibm_db.prepare(conn,sql)
418 ibm_db.bind_param(stmt,1,fname)
419 ibm_db.bind_param(stmt,2,lname)
420 ibm_db.bind_param(stmt,3,degree)
421 ibm_db.bind_param(stmt,4,branch)
422 ibm_db.bind_param(stmt,5,tenth)
423 ibm_db.bind_param(stmt,6,twelfth)
424 ibm_db.bind_param(stmt,7,domain)
425 ibm_db.bind_param(stmt,8,ug_percent)
426 ibm_db.bind_param(stmt,9,email)
427 ibm_db.bind_param(stmt,10,phone)
428 ibm_db.bind_param(stmt,11,resume)
429 ibm_db.bind_param(stmt,12,comp_name)
430 ibm_db.bind_param(stmt,13,position)
431
432 ibm_db.execute(stmt)
433 msg="You have successfully applied!"
434 return render_template('applicant_domain.html',msg=msg)
435
436 elif request.method == 'POST': msg="Please fill out the form."
437 return render_template('apply.html')
438
439 if __name__ == "__main__":
440     app.run(debug=True)
441
442
```

V