CORAL BLOCK CHAIN WEB FORM

It was my first experience to create an API, so first task for me was to choose an appropriate language to build the API.

There were two options for me to create the API:
First One is -> Node JS and
Second One -> Python

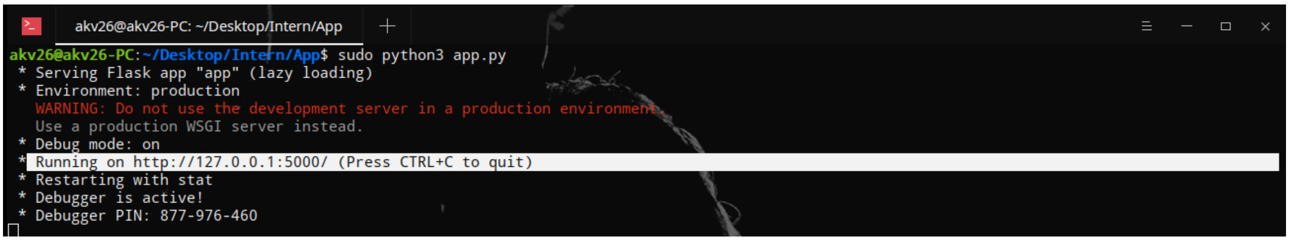
Since I am comfortable with Python instead of Node JS, Hence I chose Python and then I found the Python Framework Flask that is a micro-framework for Web Development. Django was another option for me, but since Flask is micro-framework (not so much dependencies), hence I went for Flask.

Now it was the time to learn Flask and I learned it through YouTube Videos.
Here are some YouTube Play-lists that were very helpful to learn Flask.
1. Corey Schafer
2. Traversy Media

Other than these, StackOverFlow answers, W3Schools JavaScript and HTML Tutorials also helped me a lot during doing this Assessment.

Dependencies:
1. Python
2. Flask Python Framework (pip install flask)
3. Python MySQL Connector (pip install mysql-connector-python)
4. JINJA 2 (included with Flask)
5. JavaScript

How to Run on your PC:
1. Install All Dependencies
2. Run app.py file (python app.py)
3. Open Browser and go to the url given into terminal (as shown in figure http://127.0.0.1:5000/)

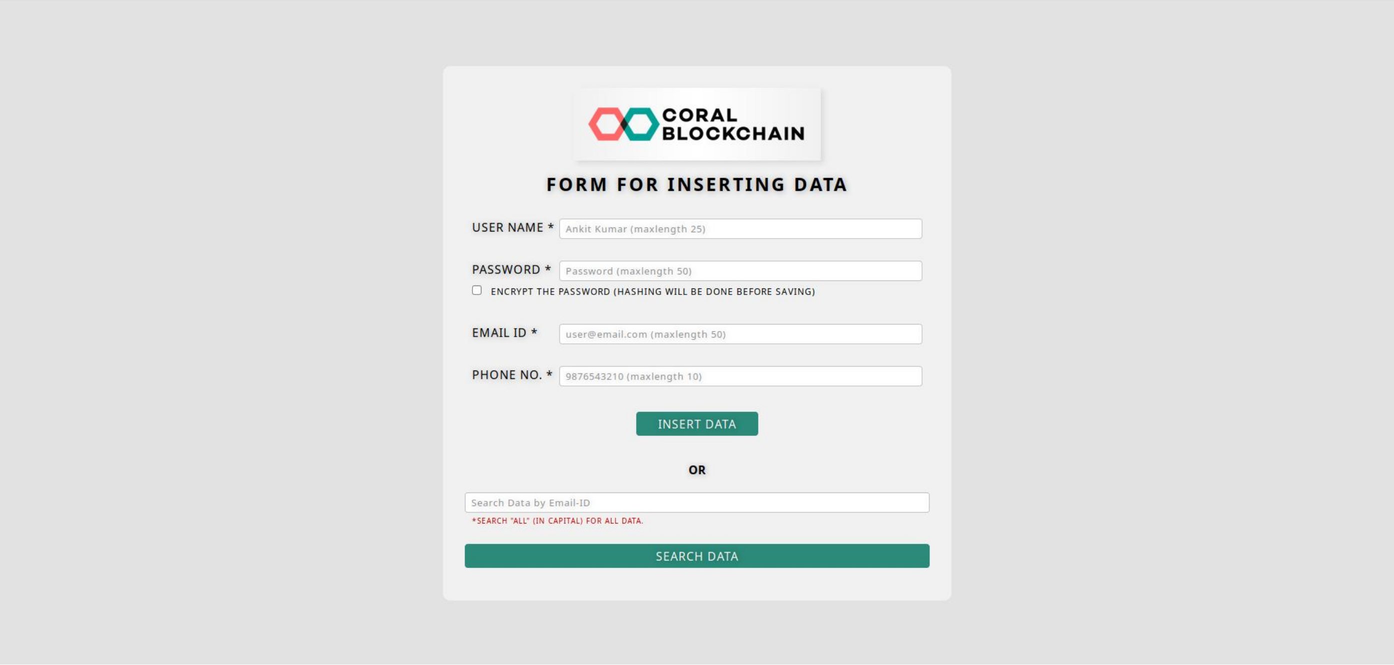


```
akv26@akv26-PC:~/Desktop/Intern/App$ sudo python3 app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 877-976-460
```

What Steps I followed during my Project:

1. First I learned a little about Flask that how it works, how Flask APP is created, how Flask APP is linked to HTML, how Data Flow will take place and these kind of basic things.

2. Then I created a HTML Form and implemented Input Validation using JavaScript.



CORAL BLOCKCHAIN

FORM FOR INSERTING DATA

USER NAME * Ankit Kumar (maxlength 25)

PASSWORD * Password (maxlength 50)

□ ENCRYPT THE PASSWORD (HASHING WILL BE DONE BEFORE SAVING)

EMAIL ID * user@email.com (maxlength 50)

PHONE NO. * 9876543210 (maxlength 10)

INSERT DATA

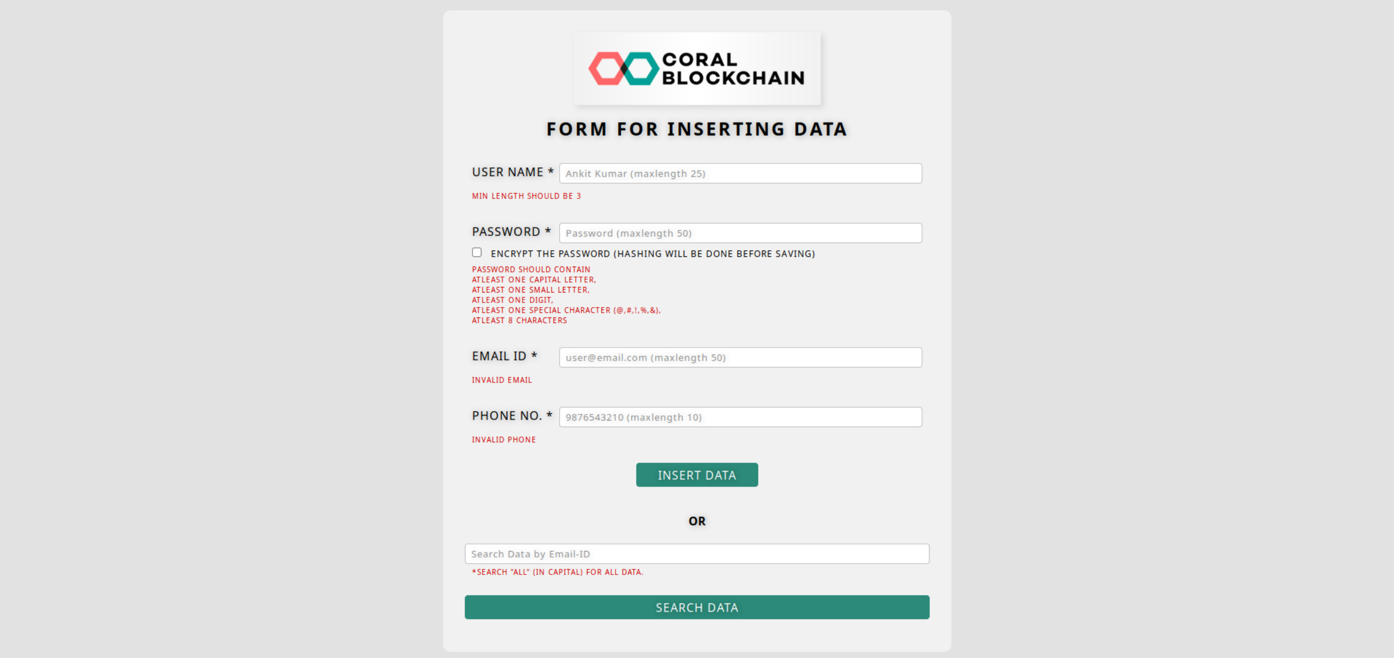
OR

Search Data by Email-ID

*SEARCH "ALL" (IN CAPITAL) FOR ALL DATA.

SEARCH DATA

Fig: How Page Looks when first time it is loaded



CORAL BLOCKCHAIN

FORM FOR INSERTING DATA

USER NAME * Ankit Kumar (maxlength 25)

MIN LENGTH SHOULD BE 3

PASSWORD * Password (maxlength 50)

□ ENCRYPT THE PASSWORD (HASHING WILL BE DONE BEFORE SAVING)

PASSWORD SHOULD CONTAIN
ATLEAST ONE CAPITAL LETTER,
ATLEAST ONE SMALL LETTER,
ATLEAST ONE DIGIT,
ATLEAST ONE SPECIAL CHARACTER (@,#,!,%,&),
ATLEAST 8 CHARACTERS

EMAIL ID * user@email.com (maxlength 50)

INVALID EMAIL

PHONE NO. * 9876543210 (maxlength 10)

INVALID PHONE

INSERT DATA

OR

Search Data by Email-ID

*SEARCH "ALL" (IN CAPITAL) FOR ALL DATA.

SEARCH DATA

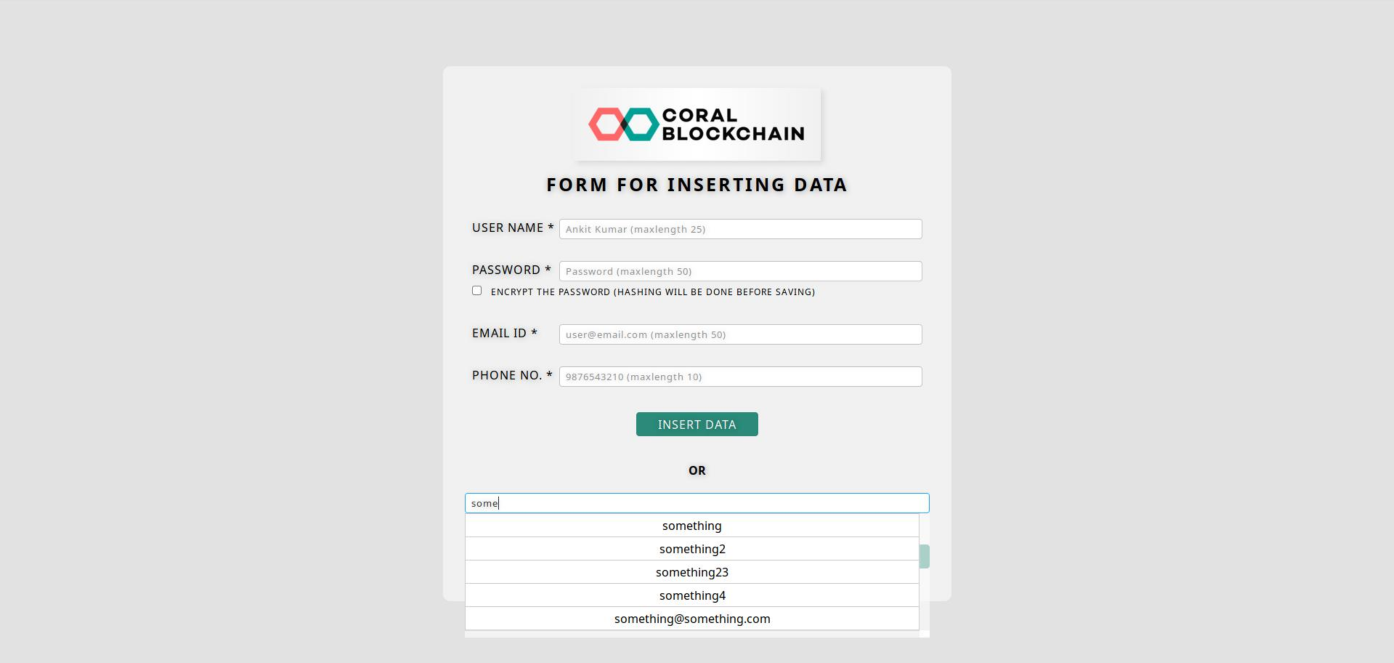
Fig: When User click at Input Field, Respective Validation Criteria is displayed

3. Then I created a Python Module that can access the MySQL Database and insert and retrieve the Data into/from this Database.

4. Now it was the time to create the Flask APP and link the HTML file with this APP. This Flask APP is the REST API that is linked to HTML form as well as the MySQL Database, that takes the Data and Request from HTML form and sends it to the MySQL Module to insert the Data into the Database.

5. Then I created a search bar that takes email-id as the input and shows the respective record.

6. Then I decided to include some extra feature that is search suggestion. And this part was a little challenging for me but all the parts were fun to do and I enjoyed doing this assessment.



USER NAME * Ankit Kumar (maxlength 25)

PASSWORD * Password (maxlength 50)

□ ENCRYPT THE PASSWORD (HASHING WILL BE DONE BEFORE SAVING)

EMAIL ID * user@email.com (maxlength 50)

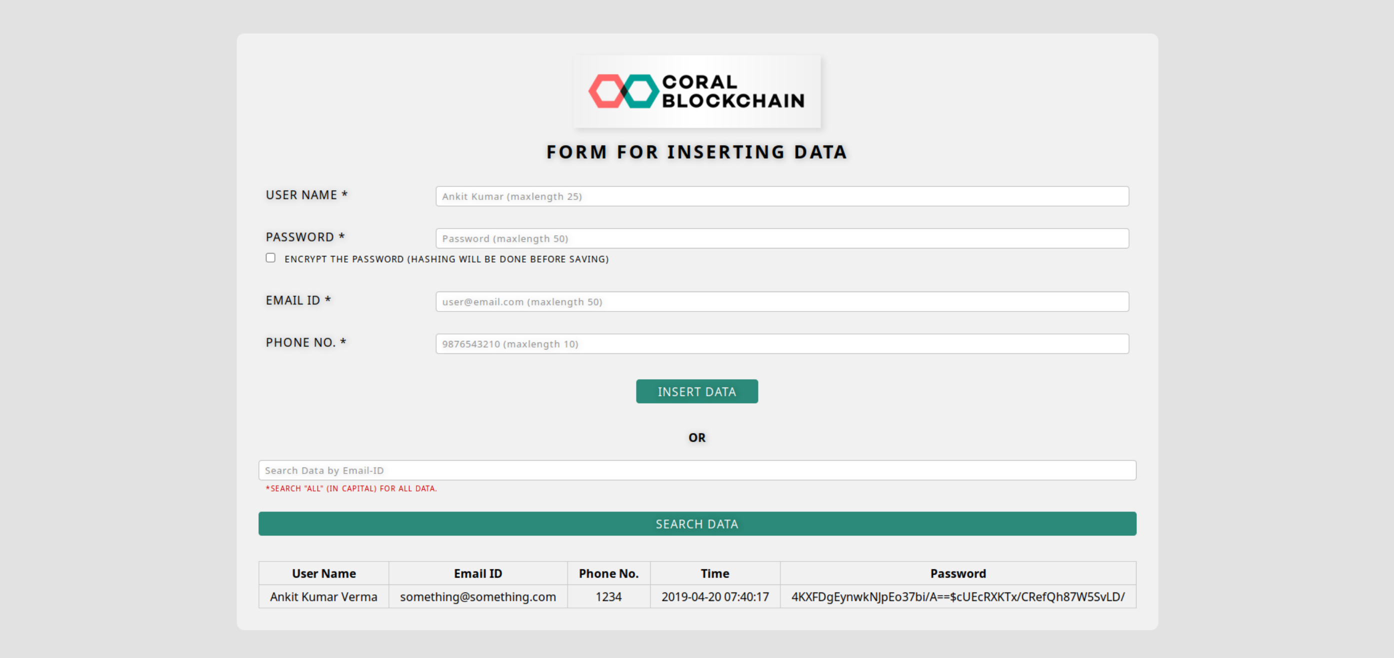
PHONE NO. * 9876543210 (maxlength 10)

some

something
something2
something23
something4
something@something.com

Fig: Search Suggestion Displaying as User starts Typing

Finally, I am very happy as I completed this assessment as I wanted it to be.



CORAL BLOCKCHAIN

FORM FOR INSERTING DATA

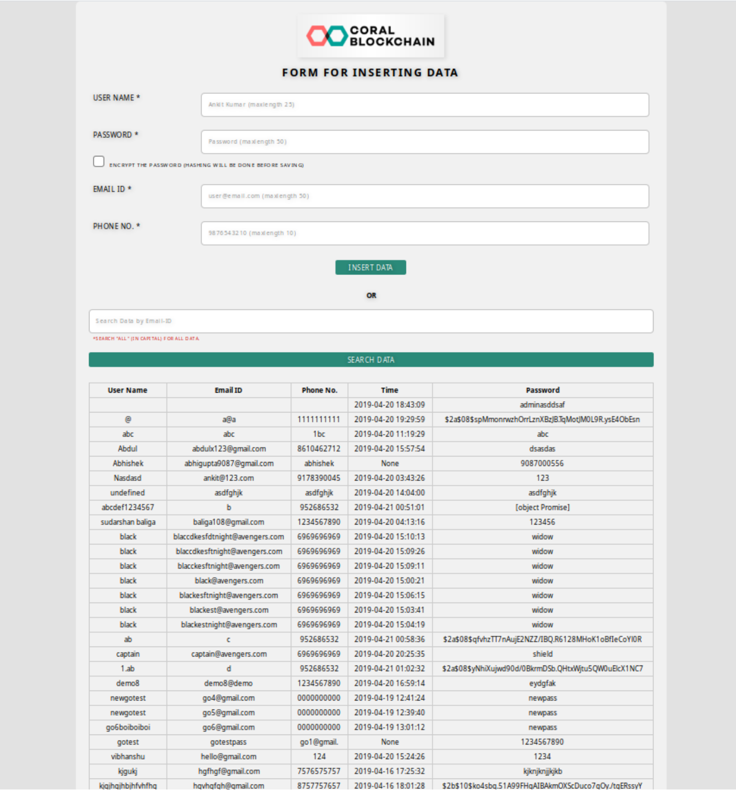
USER NAME *	Ankit Kumar (maxlength 25)
PASSWORD *	Password (maxlength 50)
□ ENCRYPT THE PASSWORD (HASHING WILL BE DONE BEFORE SAVING)	
EMAIL ID *	user@email.com (maxlength 50)
PHONE NO. *	9876543210 (maxlength 10)

Search Data by Email-ID

*SEARCH "ALL" (IN CAPITAL) FOR ALL DATA.

User Name	Email ID	Phone No.	Time	Password
Ankit Kumar Verma	something@something.com	1234	2019-04-20 07:40:17	4KXFDgEynwkNJpEo37bi/A==$cUEcRXKTx/CRefQh87W5SvLD/

Fig: Displaying Result for entered Email_ID



CORAL BLOCKCHAIN

FORM FOR INSERTING DATA

USER NAME *	Ankit Kumar (maxlength 25)
PASSWORD *	Password (maxlength 50)
□ ENCRYPT THE PASSWORD (HASHING WILL BE DONE BEFORE SAVING)	
EMAIL ID *	user@email.com (maxlength 50)
PHONE NO. *	9876543210 (maxlength 10)

Search Data by Email-ID

*SEARCH "ALL" (IN CAPITAL) FOR ALL DATA.

User Name	Email ID	Phone No.	Time	Password
			2019-04-20 18:43:09	adminasddsaf
@	a@a	1111111111	2019-04-20 19:29:59	$2a$08$spMmonrwzhOrrLznXBzJB.TqMotJM0L9R.ysE4ObEsn
abc	abc	1bc	2019-04-20 11:19:29	abc
Abdul	abdulx123@gmail.com	8610462712	2019-04-20 15:57:54	dsasdas
Abhishek	abhigupta9087@gmail.com	abhishek	None	9087000556
Nasdasd	ankit@123.com	9178390045	2019-04-20 03:43:26	123
undefined	asdfghjk	asdfghjk	2019-04-20 14:04:00	asdfghjk
abcdef1234567	b	952686532	2019-04-21 00:51:01	[object Promise]
sudarshan baliga	baliga108@gmail.com	1234567890	2019-04-20 04:13:16	123456
black	blaccdkesfdtnight@avengers.com	6969696969	2019-04-20 15:10:13	widow
black	blaccdkesftnight@avengers.com	6969696969	2019-04-20 15:09:26	widow
black	blacckesftnight@avengers.com	6969696969	2019-04-20 15:09:11	widow
black	black@avengers.com	6969696969	2019-04-20 15:00:21	widow
black	blackesftnight@avengers.com	6969696969	2019-04-20 15:06:15	widow
black	blackest@avengers.com	6969696969	2019-04-20 15:03:41	widow
black	blackestnight@avengers.com	6969696969	2019-04-20 15:04:19	widow
ab	c	952686532	2019-04-21 00:58:36	$2a$08$qfvhzTT7nAujE2NZZ/IBQ.R6128MHoK1oBfIeCoYI0R
captain	captain@avengers.com	6969696969	2019-04-20 20:25:35	shield
1.ab	d	952686532	2019-04-21 01:02:32	$2a$08$yNhiXujwd90d/0BkrmDSb.QHtxWjtu5QW0uElcX1NC7
demo8	demo8@demo	1234567890	2019-04-20 16:59:14	eydgfak
newgotest	go4@gmail.com	0000000000	2019-04-19 12:41:24	newpass
newgotest	go5@gmail.com	0000000000	2019-04-19 12:39:40	newpass
go6boiboiboi	go6@gmail.com	0000000000	2019-04-19 13:01:12	newpass
gotest	gotestpass	go1@gmail.	None	1234567890
vibhanshu	hello@gmail.com	124	2019-04-20 15:24:26	1234
kjgukj	hgfhgf@gmail.com	7576575757	2019-04-16 17:25:32	kjknjknjjkjkb
kjgjhgjhbjhfvhfhg	hgvhgfgh@gmail.com	8757757657	2019-04-16 18:01:28	$2b$10$ko4sbq.51A99FHqAIBAkmOXScDuco7qOy./tqERssyY

Fig: Displaying Result when User type ALL to see all Data

GitHub Repository Link

https://github.com/akverma26/REST-App-in-Python-Flask-for-Coral-Block-Chain

My Portfolio:

https://akverma26.github.io/

Thank You