



PROGRESS ON FILE STORAGE USING FACIAL PATTERN- BASED ENCRYPTION

John Pocasangre Jacob Acosta

Andrew Ibrahim Carlos Sandoval

Database tables progression

Established our database schema.

Will start with inserting mock data into the database and testing it.

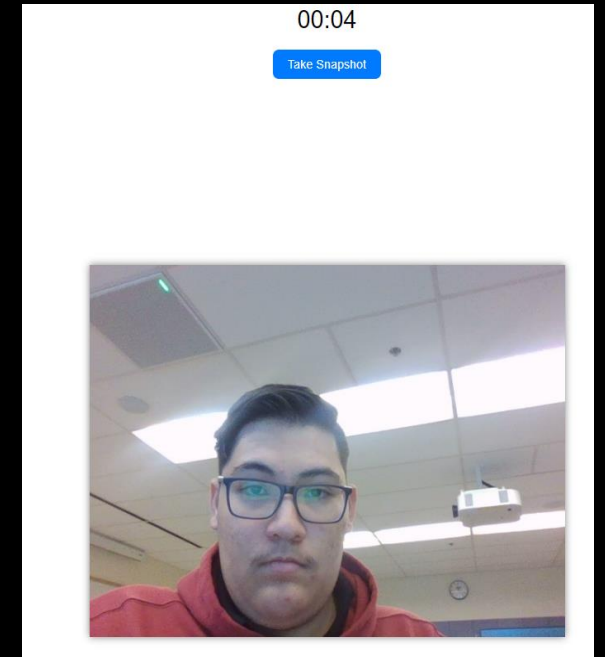
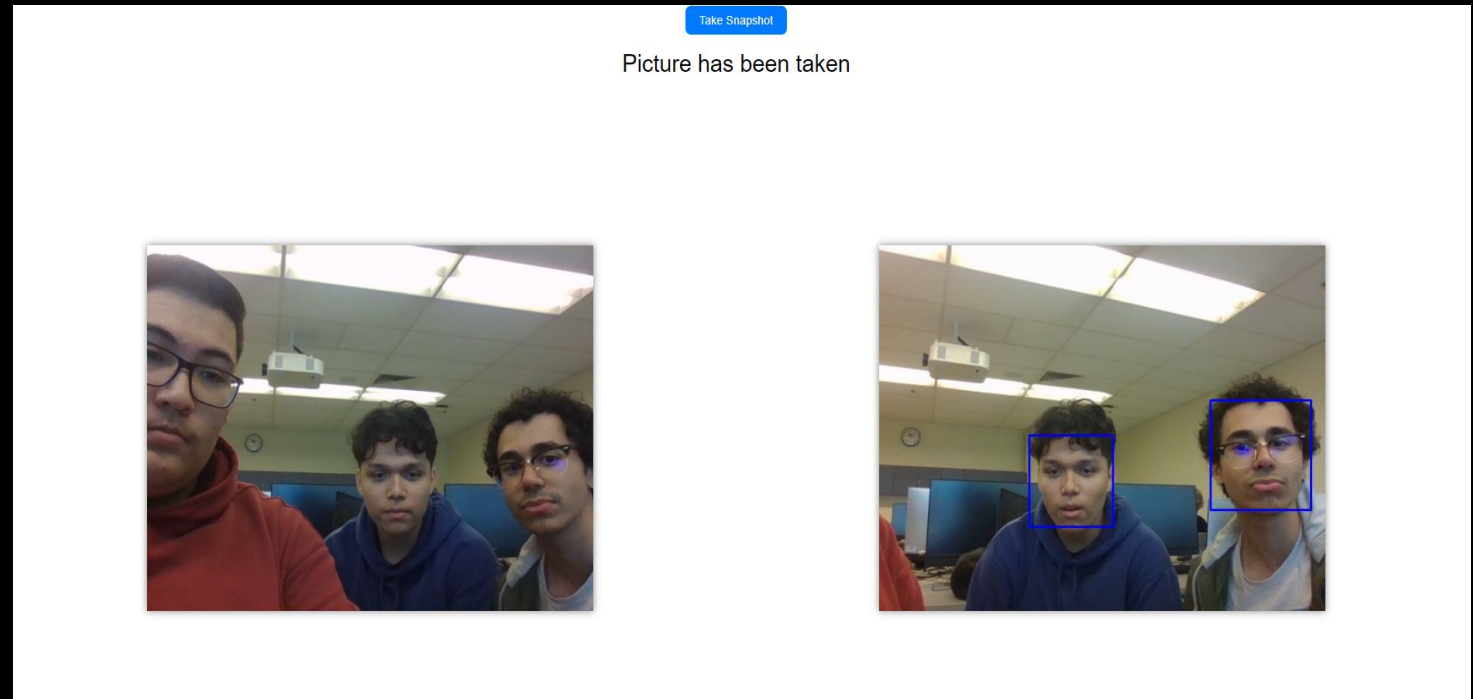
```
facial=> \dt
                List of relations
 Schema |          Name          | Type  | Owner
-----+-----+-----+-----
 public | faceauthentication    | table | facial
 public | files                  | table | facial
 public | folder                 | table | facial
 public | userinfo               | table | facial
(4 rows)

facial=>
```

Progress on analyzing Images from video stream to analyze if a Face is present

The camera stream that will be shown starts once camera access has been given, which then starts a 5 second count-down timer.

Once the timer reaches 0, a snapshot is taken of the current frame from the stream, and it is then analyzed by our python script. If a face is in the frame of said snapshot, a blue box is place on the face of the user, signaling that a face was detected/identified.



Python script for facial landmarks

- We have 2 different python scripts in use for our project. This python script was designed to scan the face and return key values that we can use to apply to the encryption
- It gathers 128 plot points on the face, and it returns variables in an array that we can use as a base
- Our plan is to merge our two python scripts into one to manage both functionalities of detecting if a face is present, and then to analyze the face that was detected to collect the data.

```
Facial Encodings: [array([-0.06060635, 0.10174929, 0.01941249, 0.02724024, -0.03361638,
0.0164675, -0.03375981, -0.04141923, 0.16718613, -0.10031269,
0.25343937, -0.04001725, -0.23539205, -0.08029607, 0.00533456,
0.0749456, -0.14844503, -0.07513712, -0.12009215, -0.00357656,
0.02560661, -0.01824091, 0.05074649, 0.04222828, -0.20660993,
-0.42820147, -0.06964193, -0.15126459, 0.01738106, -0.10917728,
-0.04653131, 0.03305087, -0.13118213, -0.0388199, -0.02867581,
0.05542243, -0.05966679, -0.0330907, 0.26813519, 0.06479893,
-0.17173457, 0.01750953, -0.01064712, 0.32452053, 0.17978048,
0.02066524, 0.09670069, -0.11797647, 0.13564862, -0.24755797,
0.05238116, 0.21025997, 0.07710855, 0.09180203, 0.11032964,
-0.07932048, 0.0421046, 0.08803082, -0.19081338, 0.07540342,
0.04334394, 0.01900539, -0.02830791, -0.02365316, 0.18003166,
0.08272086, -0.08757217, -0.07496215, 0.14059207, -0.18065797,
-0.02553346, 0.08733747, -0.12865219, -0.25934675, -0.29048204,
0.04990864, 0.46362171, 0.18099125, -0.16311243, 0.00417472,
-0.10214896, -0.12447654, 0.12483213, 0.08306618, -0.09005607,
-0.03854111, -0.06170053, 0.11482343, 0.22592987, -0.01355261,
0.00406403, 0.17404921, 0.0585909, -0.02400492, 0.0513254,
0.04182218, -0.11650316, 0.01492986, -0.10679778, 0.03084203,
0.06905235, -0.14029846, 0.00612603, 0.04680334, -0.24230307,
0.16059329, 0.03657659, -0.1010024, 0.07584623, 0.03226487,
-0.1330774, -0.04536681, 0.18656257, -0.28030977, 0.15146963,
0.1914424, 0.0452369, 0.14349791, 0.03459471, 0.10090941,
-0.04727468, 0.01020074, -0.1479747, -0.07910535, -0.01082899,
0.04719732, 0.01638329, -0.03914913]])]
```