DESIGN DOC

Structuer of the folder:

```
-- images
        -- contains test images
-- myvenvpy
        -- virtual env
-- src
        -- connection.py
                -- has a class to create connection to the postgresql database
        -- db.sql
                -- contains create table command
        -- list_sublist.py
                -- contains function to sort a list of list on the basis of the first element and the forth
                   element
        -- main.py
                -- contains the api routes and is the main file with all api routes defined and calls
                   internally connection.py and list_sublist.py to establish a connection and sort a
                   nested list
        -- test1.py
                -- contains tests to test all aspects of search_image api route
        -- test2.py
                -- contains tests to test all aspects of add_face api route
        -- test3.py
                -- contains tests to test all aspects of add_faces_in_bulk api route
        -- test4.py
                -- contains tests to test all aspects of get_face_info api route
        -- test5.py
                -- contains tests to test all aspects of  add_meta_data api route
```

Structure of the Table used to store all the face records:

```
                         Table "public.faces"
   Column    |          Type          | Collation | Nullable |             Default
-------------+------------------------+-----------+----------+----------------------------------
 id          | integer                |           | not null | nextval('faces_id_seq'::regclass)
 name        | character varying(200) |           |          |
 vector      | double precision[]     |           |          |
 person_name | character varying(200) |           |          |
 version     | integer                |           |          |
 date        | character varying(200) |           |          |
```

Description of the fields are:

- id: Stores face_id, is serial, taht is auto_increment.
- Name: Stores name of the image file
- vector: Stores the encodings of the image file
- person_name: Stores name of the person whose face is in the image
- version: Stores version fo the image
- date: Stores date the image was taken

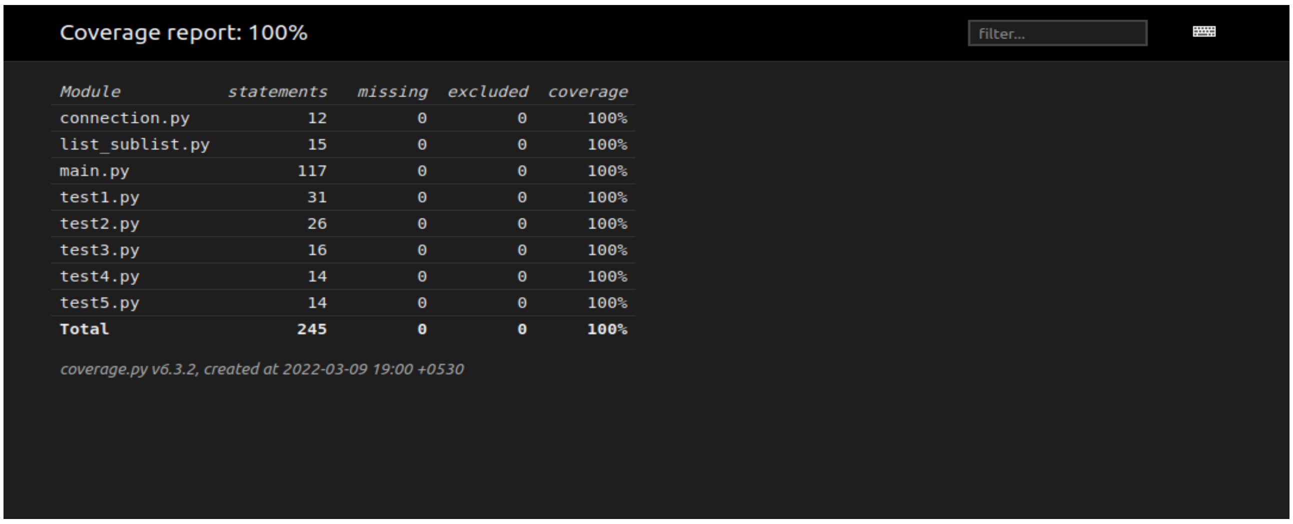
Structure of api routes:

```
-- search_faces
        -- Parameters:
                -- file: UploadFile : image file to be searched from
                -- k: int = Form(...) : maximum number of matches to display
                -- confidence: float = Form(...) : minimum confidence in the match
        -- Responses:
                -- Not an image file
                        -- {"status": "ERROR", "body": "Not an image file"}
                -- No face in image
                        -- {"status": "ERROR", "body": "No face in the given image"}
                -- Sucessful serach
                        -- return {"status": "OK", "body": body}
-- add_face
        -- Parameters:
                -- file: UploadFile : image file to be added
        -- Responses:
                -- Not an image file
                        -- {"status": "ERROR", "body": "Not an image file"}
                -- No face in image
                        -- {"status": "ERROR", "body": "No face in the given image"}
                -- Multiple faces in image
                        -- {"status": "ERROR", "body": "Multiple faces found"}
                -- Successful Addition
                        -- return {"status": "OK", "body": "successfully added " + file.filename}
-- add_faces_in_bulk
        -- Parameters
                -- file: UploaqdFile : zip file from which images are to be added
        -- Responses:
                -- Not a zip file
                        -- {"status": "ERROR", "body": "Not a zip file"}
                -- No image in the zip file
                        -- {"status": "ERROR", "body": "No image file in given zip file"}
                -- Successful Addition
                        -- {"status": "OK", "body": " Successfully added "+ str(count) + " images"}
-- get_face_info
        -- Parameters
                -- apikey: str = Form(...)
                -- face_id: str = Form(...) : id of the face of which information needs to be sent
        -- Responses
                -- No face with given id
                        -- {"status": "ERROR", "body": "No face with the given id"}
                -- Successful retrieval
                        -- {"status": "OK", "body": body}
-- add_meta_data
        -- Parameters
                -- person_name: str = Form(...) : Name of the person to be added as meta-data
                -- face_id: str = Form(...) : id of face whose meta-data is to be added
                -- version_face: int = Form(...) : version of the image to be added
                -- date: str = Form(...) : date the image was taken, to be added as meta-data
        -- Responses
                -- No face with given id
```

-- {"status": "ERROR", "body": "No face with the given id"}
-- Successul updation
-- {"status":"OK", "body" : "Meta Data added for "+ face_id}

Coverage:

-- index.html

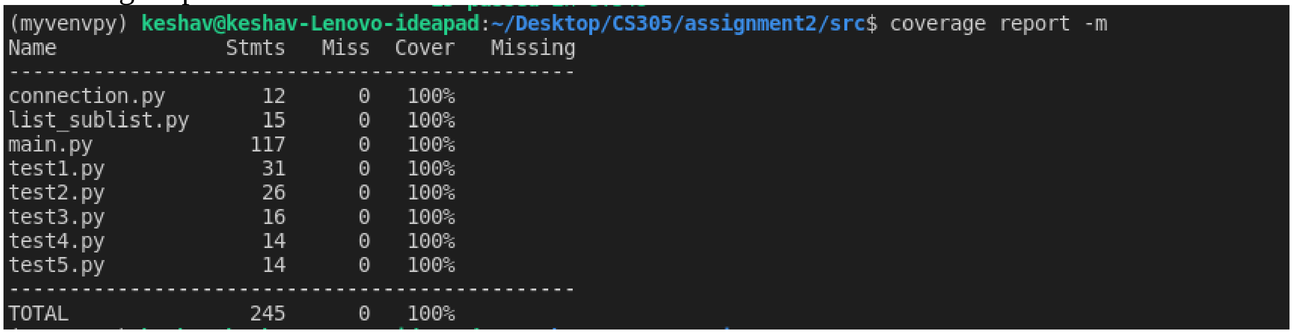


```
Coverage report: 100%                                            filter...

Module               statements   missing   excluded   coverage
connection.py                12         0          0       100%
list_sublist.py              15         0          0       100%
main.py                     117         0          0       100%
test1.py                     31         0          0       100%
test2.py                     26         0          0       100%
test3.py                     16         0          0       100%
test4.py                     14         0          0       100%
test5.py                     14         0          0       100%
Total                       245         0          0       100%

coverage.py v6.3.2, created at 2022-03-09 19:00 +0530
```

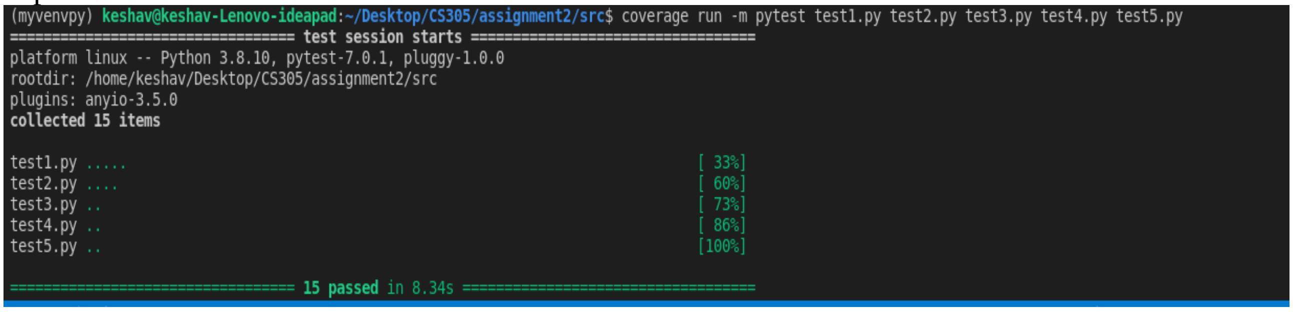
-- coverage report



```
(myvenvpy) keshav@keshav-Lenovo-ideapad:~/Desktop/CS305/assignment2/src$ coverage report -m
Name                Stmts   Miss  Cover   Missing
-------------------------------------------------
connection.py          12      0   100%
list_sublist.py        15      0   100%
main.py               117      0   100%
test1.py               31      0   100%
test2.py               26      0   100%
test3.py               16      0   100%
test4.py               14      0   100%
test5.py               14      0   100%
-------------------------------------------------
TOTAL                 245      0   100%
```

Tests:

-- passed all tests



```
(myvenvpy) keshav@keshav-Lenovo-ideapad:~/Desktop/CS305/assignment2/src$ coverage run -m pytest test1.py test2.py test3.py test4.py test5.py
================================== test session starts ==================================
platform linux -- Python 3.8.10, pytest-7.0.1, pluggy-1.0.0
rootdir: /home/keshav/Desktop/CS305/assignment2/src
plugins: anyio-3.5.0
collected 15 items

test1.py .....                                                                  [ 33%]
test2.py ....                                                                   [ 60%]
test3.py ..                                                                     [ 73%]
test4.py ..                                                                     [ 86%]
test5.py ..                                                                     [100%]

================================== 15 passed in 8.34s ===================================
```