Problem statement:

Let's suppose a small startup wants to build a Lost and Found platform which would help the end user to search and view the lost items and upload the found items. Let's suppose you got the opportunity to join this startup as a backend Intern. You must provide clear and complete documentation about how to run your program. You should be able to handle different routes in your app. The following features Restful APIs implementation is given to you as a first task to build for a lost and found platform.

Approach towards finding adequate Solution:

The first step took towards the destination is a milestone. The first and foremost requirement for this application is the implementation of models which include user and items. Furthermore, Modular approach is followed and implemented on the given task. Fast Api and SqlAlchemy are used and all the end points are defined. The documentation of Fast Api helped me a lot throughout the project. Fast Api server is used for the testing part of the application.

Problems faced during the task:

Fast Api is something never heard before the implementation of this task. Several technical issues were faced during the implementation.

Docker "This run on my machine" in order to overcome this problem as far as I have learned about Docker. Docker implementation is bit tricky and annoying as a beginner I faced a lot of issues while implementing Docker so in at the end I removed implementation of docker from my application. Several other technical issues are also faced during implementation regarding object relational mapping (ORM) and the concepts to grasp on includes Synchronous and Asynchronous tasks. Figures of the project including fast Api server testing are attached.

Fig :1

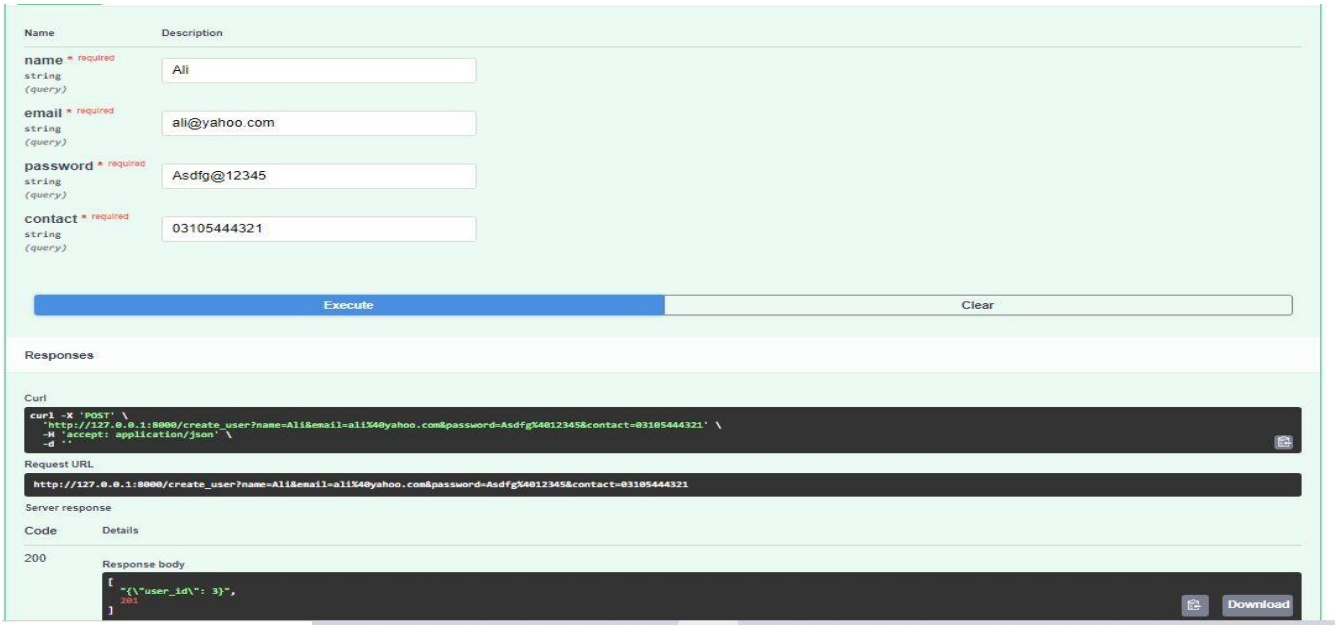

Name
Description
name * required
string
(query)
Ali
email * required
string
(query)
ali@yahoo.com
password * required
string
(query)
Asdfg@12345
contact * required
string
(query)
03105444321
Execute
Clear
Responses
Curl
curl -X 'POST' \
'http://127.0.0.1:8000/create_user?name=Ali&email=ali%40yahoo.com&password=Asdfg%4012345&contact=03105444321' \
-H 'accept: application/json' \
-d ''
Request URL
http://127.0.0.1:8000/create_user?name=Ali&email=ali%40yahoo.com&password=Asdfg%4012345&contact=03105444321
Server response
Code
Details
200
Response body
[
"{\"user_id\": 3}",
]
Download

Fig : 2

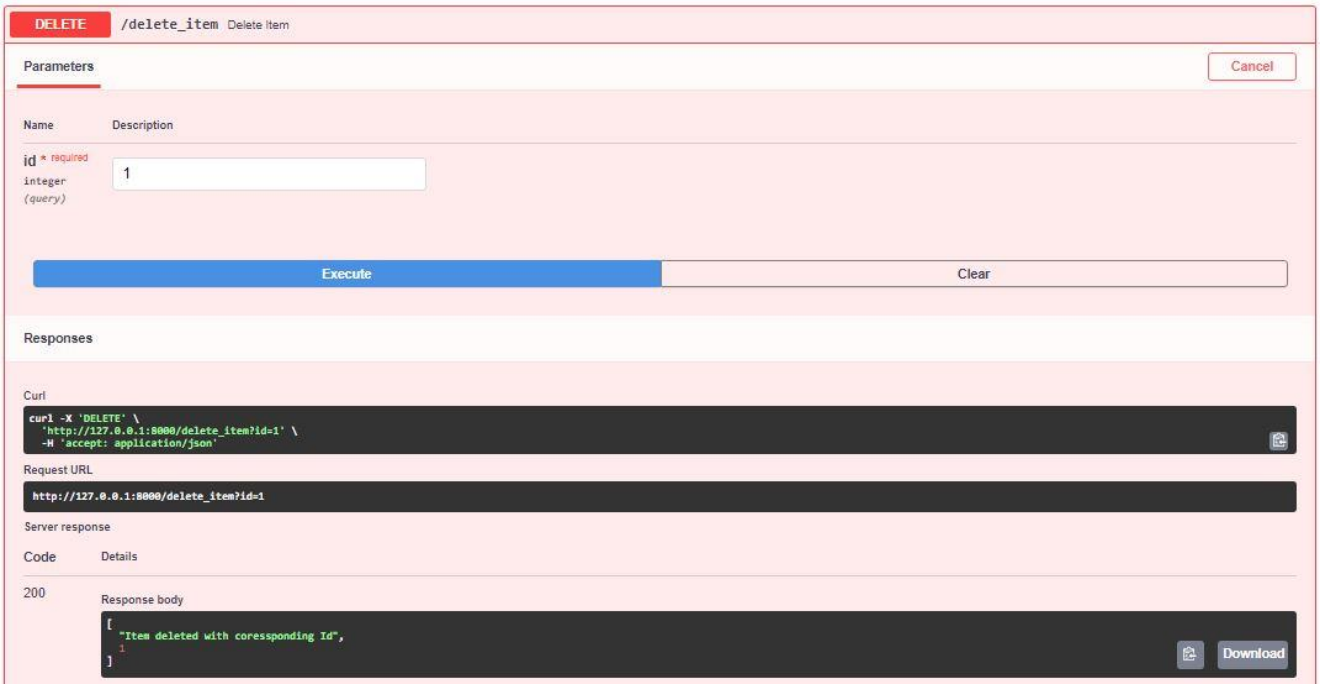

DELETE
/delete_item
Delete Item
Parameters
Cancel
Name
Description
id * required
integer
(query)
1
Execute
Clear
Responses
Curl
curl -X 'DELETE' \
'http://127.0.0.1:8000/delete_item?id=1' \
-H 'accept: application/json'
Request URL
http://127.0.0.1:8000/delete_item?id=1
Server response
Code
Details
200
Response body
[
"Item deleted with coressponding Id",
]
Download

Fig : 3

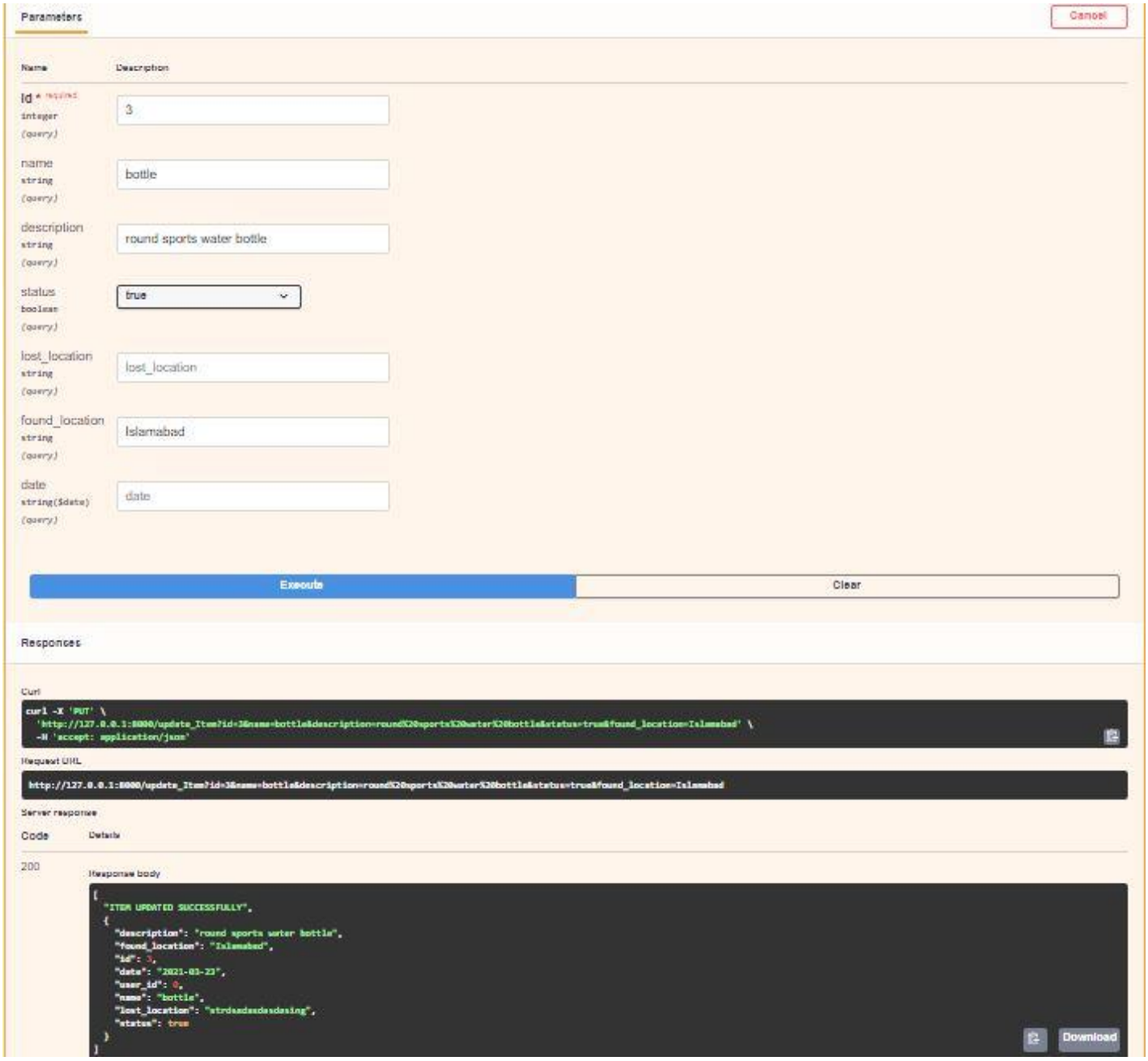

Parameters
Cancel
Name
Description
Id * required
integer
(query)
3
name
string
(query)
bottle
description
string
(query)
round sports water bottle
status
boolean
(query)
true
lost_location
string
(query)
lost_location
found_location
string
(query)
Islamabad
date
string($date)
(query)
date
Execute
Clear
Responses
Curl
curl -X 'PUT' \
'http://127.0.0.1:8000/update_Item?id=3&name=bottle&description=round%20sports%20water%20bottle&status=true&found_location=Islamabad' \
-H 'accept: application/json'
Request URL
http://127.0.0.1:8000/update_Item?id=3&name=bottle&description=round%20sports%20water%20bottle&status=true&found_location=Islamabad
Server response
Code
Details
200
Response body
{
"ITEM UPDATED SUCCESSFULLY",
{
"description": "round sports water bottle",
"found_location": "Islamabad",
"id": 3,
"date": "2021-03-23",
"user_id": 0,
"name": "bottle",
"lost_location": "strdsadasdasdasing",
"status": true
}
]
Download

Fig : 4

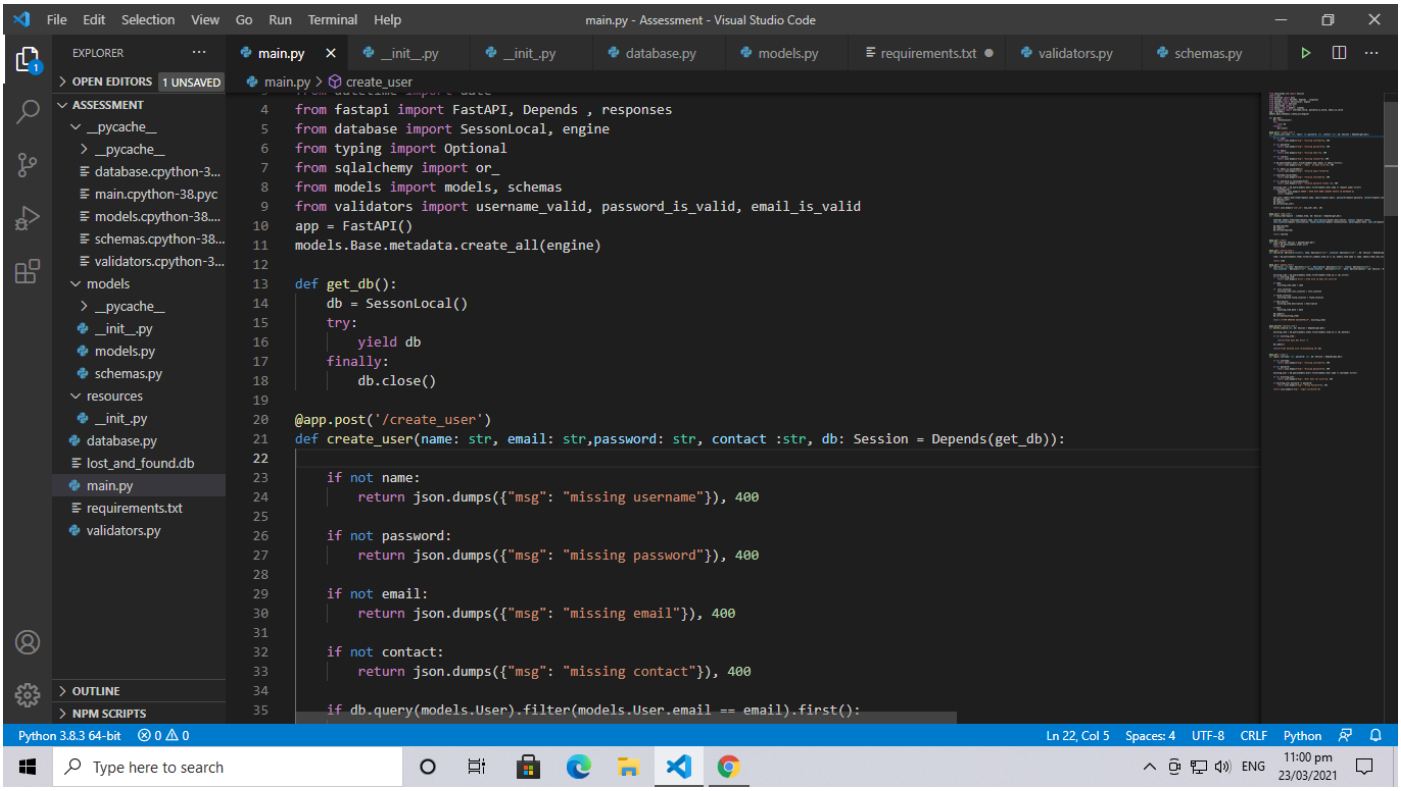
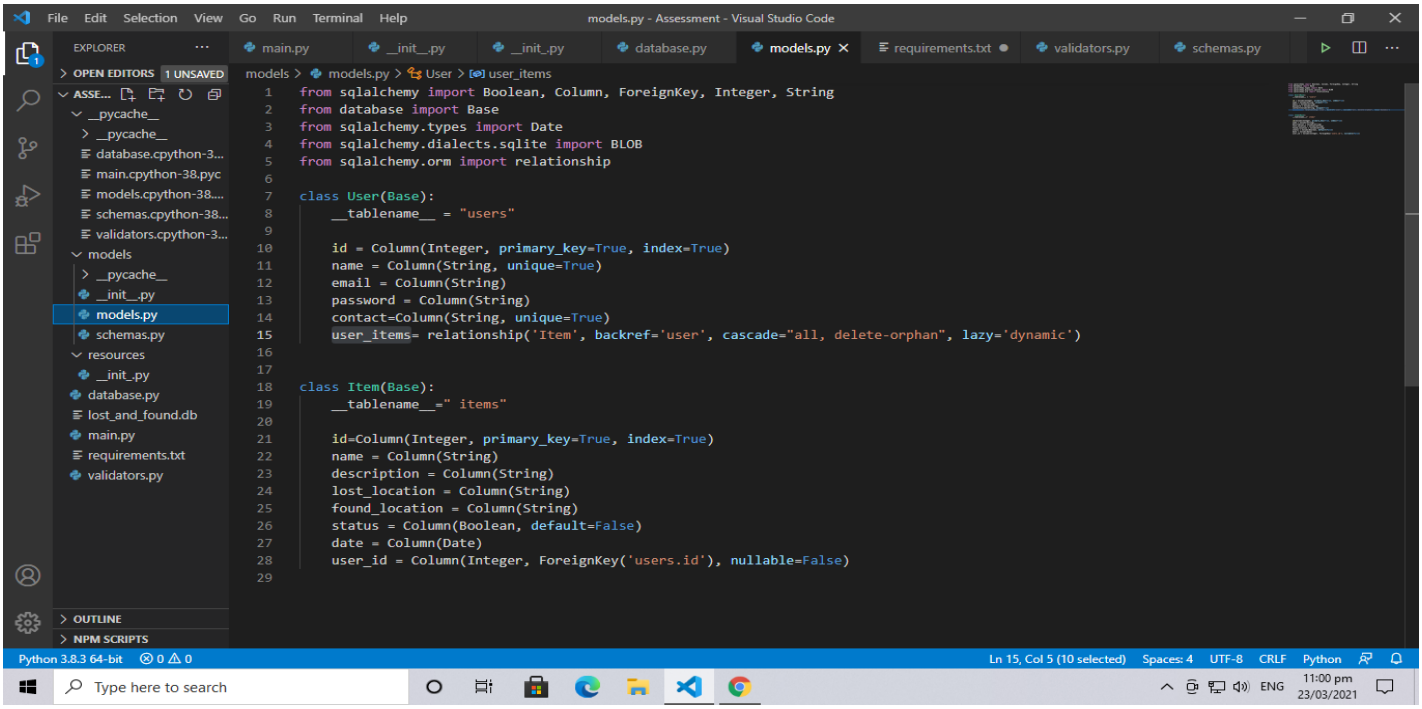

File Edit Selection View Go Run Terminal Help
main.py - Assessment - Visual Studio Code
EXPLORER
OPEN EDITORS 1 UNSAVED
ASSESSMENT
main.py
__init__.py
__init_.py
database.py
models.py
requirements.txt
validators.py
schemas.py
main.py > create_user
from fastapi import FastAPI, Depends , responses
from database import SessonLocal, engine
from typing import Optional
from sqlalchemy import or_
from models import models, schemas
from validators import username_valid, password_is_valid, email_is_valid
app = FastAPI()
models.Base.metadata.create_all(engine)
def get_db():
db = SessonLocal()
try:
yield db
finally:
db.close()
@app.post('/create_user')
def create_user(name: str, email: str,password: str, contact :str, db: Session = Depends(get_db)):
if not name:
return json.dumps({"msg": "missing username"}), 400
if not password:
return json.dumps({"msg": "missing password"}), 400
if not email:
return json.dumps({"msg": "missing email"}), 400
if not contact:
return json.dumps({"msg": "missing contact"}), 400
if db.query(models.User).filter(models.User.email == email).first():
__pycache__
database.cpython-3...
main.cpython-38.pyc
models.cpython-38....
schemas.cpython-38...
validators.cpython-3...
models
resources
lost_and_found.db
requirements.txt
OUTLINE
NPM SCRIPTS
Python 3.8.3 64-bit
Ln 22, Col 5 Spaces: 4 UTF-8 CRLF Python
Type here to search
ENG
11:00 pm
23/03/2021

Fig : 5


File Edit Selection View Go Run Terminal Help
models.py - Assessment - Visual Studio Code
EXPLORER
OPEN EDITORS 1 UNSAVED
main.py
__init__.py
__init_.py
database.py
models.py
requirements.txt
validators.py
schemas.py
models > models.py > User > user_items
from sqlalchemy import Boolean, Column, ForeignKey, Integer, String
from database import Base
from sqlalchemy.types import Date
from sqlalchemy.dialects.sqlite import BLOB
from sqlalchemy.orm import relationship
class User(Base):
__tablename__ = "users"
id = Column(Integer, primary_key=True, index=True)
name = Column(String, unique=True)
email = Column(String)
password = Column(String)
contact=Column(String, unique=True)
user_items= relationship('Item', backref='user', cascade="all, delete-orphan", lazy='dynamic')
class Item(Base):
__tablename__=" items"
id=Column(Integer, primary_key=True, index=True)
name = Column(String)
description = Column(String)
lost_location = Column(String)
found_location = Column(String)
status = Column(Boolean, default=False)
date = Column(Date)
user_id = Column(Integer, ForeignKey('users.id'), nullable=False)
__pycache__
database.cpython-3...
main.cpython-38.pyc
models.cpython-38....
schemas.cpython-38...
validators.cpython-3...
models
resources
lost_and_found.db
requirements.txt
OUTLINE
NPM SCRIPTS
Python 3.8.3 64-bit
Ln 15, Col 5 (10 selected) Spaces: 4 UTF-8 CRLF Python
Type here to search
ENG
11:00 pm
23/03/2021