ESD (Employability Skills Development) FA-2

Title:

Create the API in python for complete CRUD and test it with postman – Expense Tracker App

Tools: Microsoft Word, PhpMyAdmin

Screenshot:

Database Design

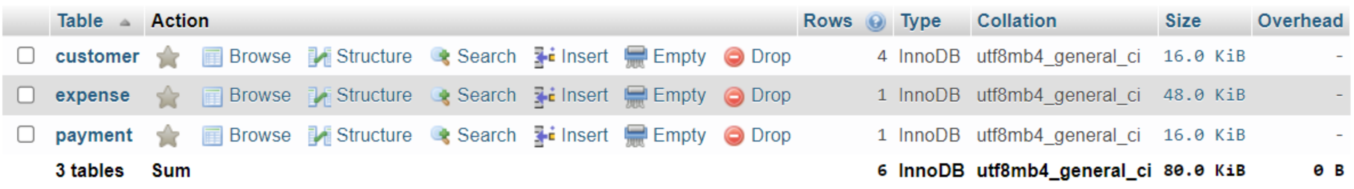
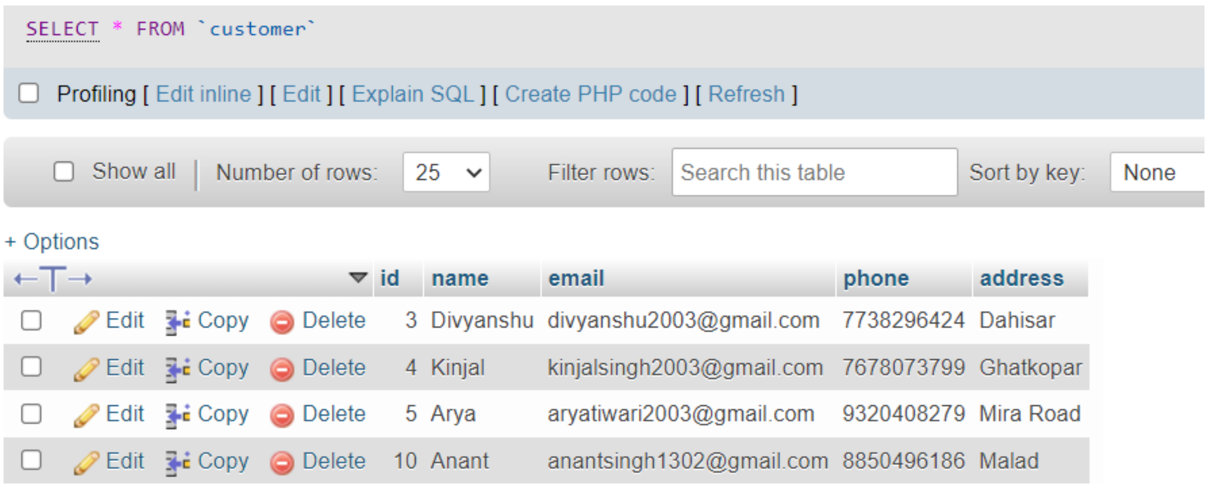


Table	Action	Rows	Type	Collation	Size	Overhead
customer	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB	-
expense	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	48.0 KiB	-
payment	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-
3 tables	Sum	6	InnoDB	utf8mb4_general_ci	80.0 KiB	0 B

Table: Customer



```
SELECT * FROM `customer`
```

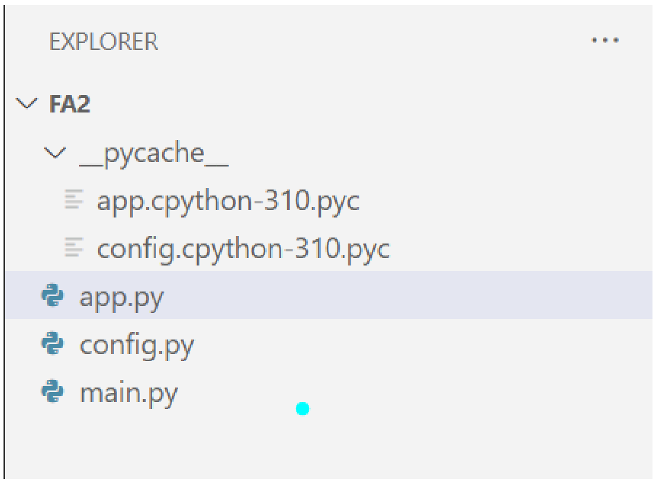
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	id	name	email	phone	address
Edit Copy Delete	3	Divyanshu	divyanshu2003@gmail.com	7738296424	Dahisar
Edit Copy Delete	4	Kinjal	kinjalsingh2003@gmail.com	7678073799	Ghatkopar
Edit Copy Delete	5	Arya	aryatiwari2003@gmail.com	9320408279	Mira Road
Edit Copy Delete	10	Anant	anantsingh1302@gmail.com	8850496186	Malad

Project Directory



```
EXPLORER
FA2
  __pycache__
    app.cpython-310.pyc
    config.cpython-310.pyc
  app.py
  config.py
  main.py
```

app.py

```
from flask import Flask
from flask_cors import CORS, cross_origin

app = Flask(__name__)
CORS(app)
```

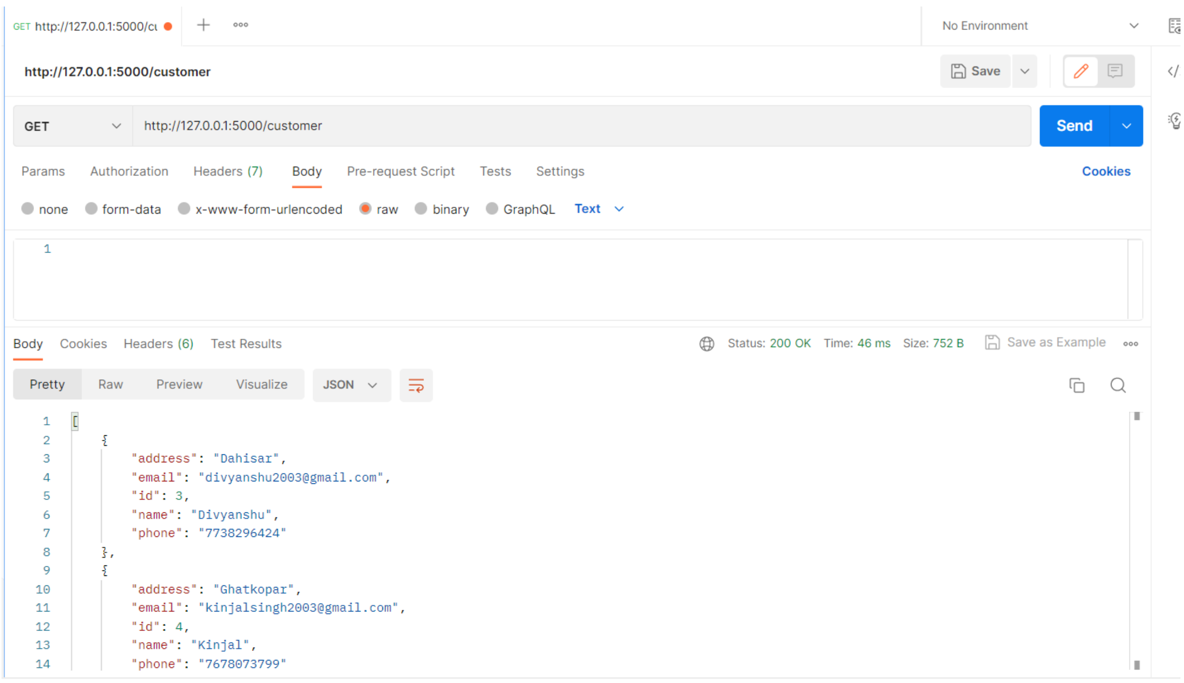
config.py

```
from app import app
from flaskext.mysql import MySQL

mysql = MySQL()
app.config['MYSQL_DATABASE_USER'] = 'root'
app.config['MYSQL_DATABASE_PASSWORD'] = ''
app.config['MYSQL_DATABASE_DB'] = 'esd_grp. 13'
app.config['MYSQL_DATABASE_HOST'] = 'localhost'
mysql.init_app(app)
```

main.py

```
import pymysql
from app import app
from config import mysql
from flask import jsonify
from flask import flash, request
from contextlib import closing

@app.route('/create', methods=['POST'])
def create_customer():
    try:
        _json = request.json
        _name = _json['name']
        _email = _json['email']
        _phone = _json['phone']
        _address = _json['address']
        if _name and _email and _phone and
_address and request.method == 'POST':
            conn = mysql.connect()
            cursor =
conn.cursor(pymysql.cursors.DictCursor
)
            sqlQuery = "INSERT INTO
customer(name, email, phone, address)
VALUES(%s, %s, %s, %s)"
            bindData = (_name, _email,
_phone, _address)
            cursor.execute(sqlQuery,
bindData)
            conn.commit()
            respone = jsonify('User added
successfully!')
            respone.status_code = 200
            return respone
        else:
            return showMessage("User not
added successfully")
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()


@app.route('/customer')
def customer():
    try:
        conn = mysql.connect()
        cursor =
conn.cursor(pymysql.cursors.DictCursor
)
        cursor.execute("SELECT id, name,
email, phone, address FROM customer")
        customerRows = cursor.fetchall()
        respone = jsonify(customerRows)
        respone.status_code = 200
        return respone
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()


@app.route('/customer/<int:customer_id
>')
def customer_details(customer_id):
    try:
        conn = mysql.connect()
        cursor =
conn.cursor(pymysql.cursors.DictCursor
)
        cursor.execute("SELECT id, name,
email, phone, address FROM customer
WHERE id =%s", customer_id)
        customerRow = cursor.fetchone()
        respone = jsonify(customerRow)
        respone.status_code = 200
        return respone
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()


@app.route('/update', methods=['PUT'])
def update_customer():
    try:
        _json = request.json
        _id = _json['id']
        _name = _json['name']
        _email = _json['email']
        _phone = _json['phone']
        _address = _json['address']
        if _name and _email and _phone and
_address and _id and request.method ==
'PUT':
            sqlQuery = "UPDATE customer
SET name=%s, email=%s, phone=%s,
address=%s WHERE id=%s"
            bindData = (_name, _email,
_phone, _address, _id,)
            conn = mysql.connect()
            cursor = conn.cursor()
            cursor.execute(sqlQuery,
bindData)
            conn.commit()
            respone = jsonify('User updated
successfully!')
            respone.status_code = 200
            return respone
        else:
            return showMessage('"User not
updated successfully"')
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()


@app.route('/delete/<int:id>',
methods=['DELETE'])
def delete_customer(id):
    try:
        conn = mysql.connect()
        cursor = conn.cursor()
        cursor.execute("DELETE FROM
customer WHERE id =%s", (id,))
        conn.commit()
        respone = jsonify('User deleted
successfully!')
        respone.status_code = 200
        return respone
    except Exception as e:
        print(e)
    finally:
        cursor.close()
        conn.close()


@app.errorhandler(404)
def showMessage(error=None):
    message = {
        'status': 404,
        'message': 'Record not found: ' +
request.url,
    }
    respone = jsonify(message)
    respone.status_code = 404
    return respone


if __name__ == "__main__":
    app.run(debug=True)
```

API_GET (READ)



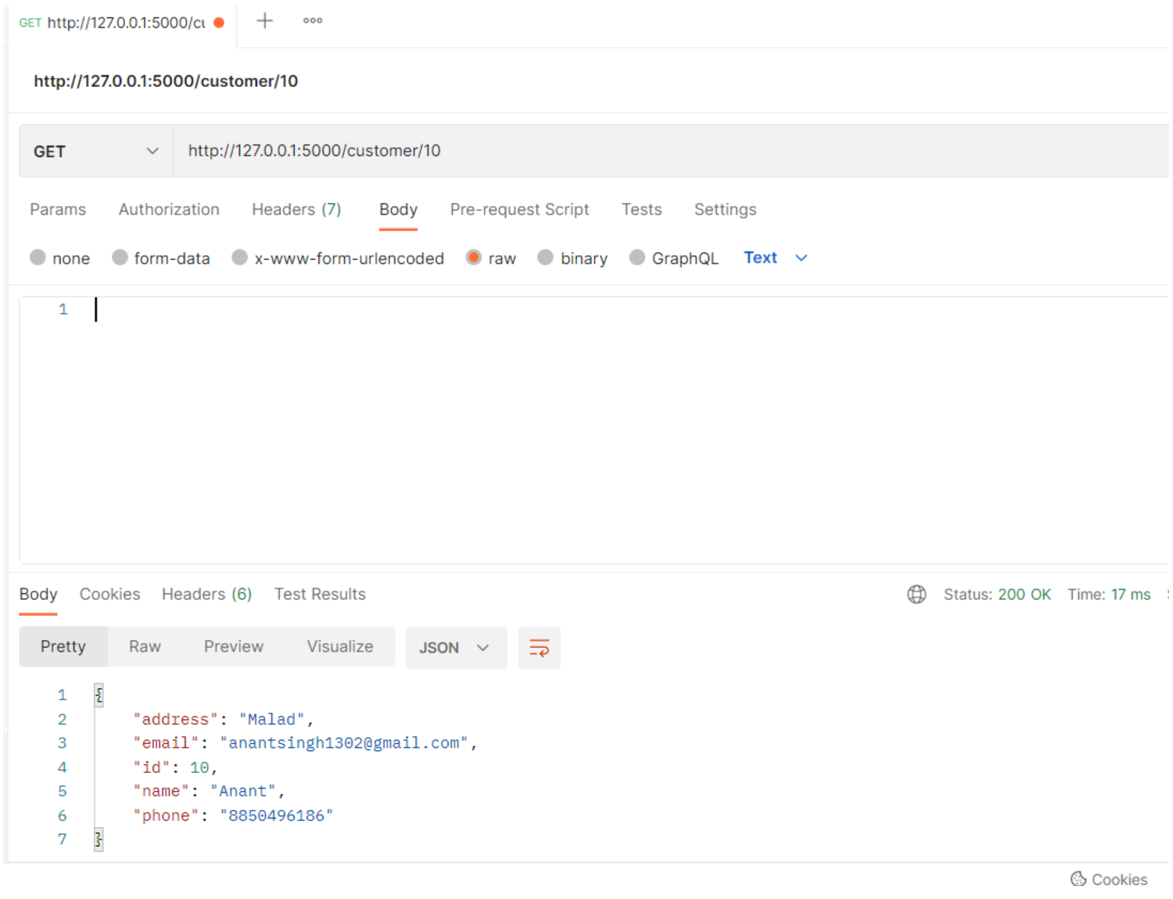
GET http://127.0.0.1:5000/customer

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Body: raw, Text

Status: 200 OK Time: 46 ms Size: 752 B

```
[
    {
        "address": "Dahisar",
        "email": "divyanshu2003@gmail.com",
        "id": 3,
        "name": "Divyanshu",
        "phone": "7738296424"
    },
    {
        "address": "Ghatkopar",
        "email": "kinjalsingh2003@gmail.com",
        "id": 4,
        "name": "Kinjal",
        "phone": "7678073799"
```



GET http://127.0.0.1:5000/customer/10

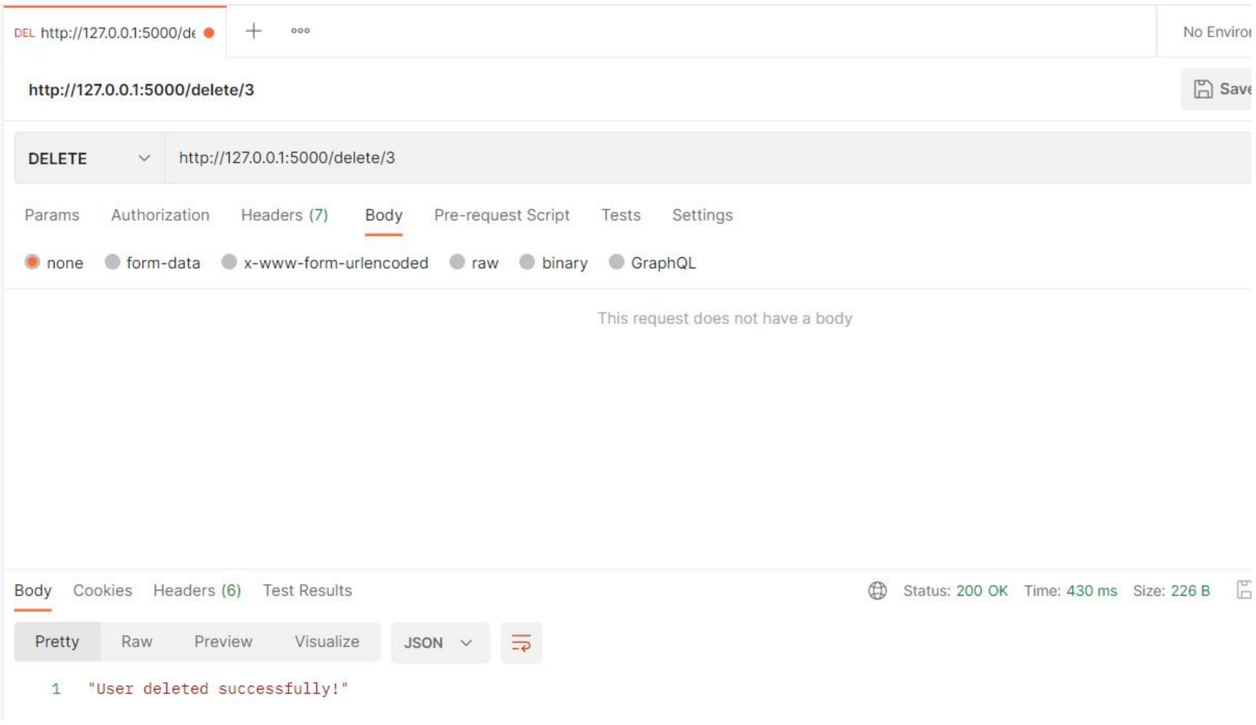
Params Authorization Headers (7) Body Pre-request Script Tests Settings

Body: raw, Text

Status: 200 OK Time: 17 ms

```
{
    "address": "Malad",
    "email": "anantsingh1302@gmail.com",
    "id": 10,
    "name": "Anant",
    "phone": "8850496186"
}
```

API_DELETE (DELETE)



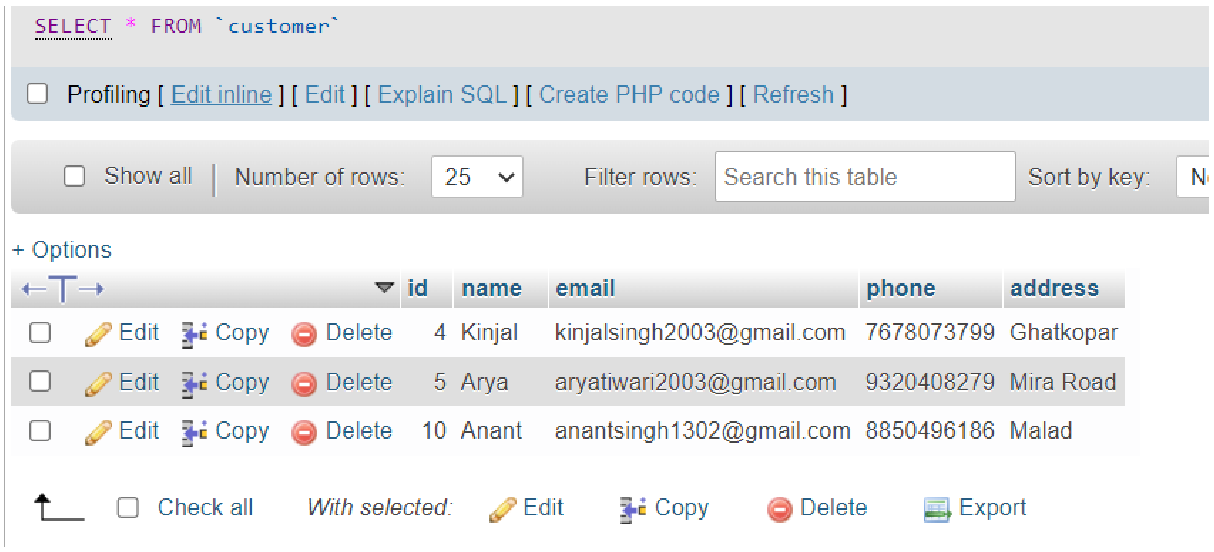
http://127.0.0.1:5000/delete/3

DELETE http://127.0.0.1:5000/delete/3

This request does not have a body

Status: 200 OK Time: 430 ms Size: 226 B

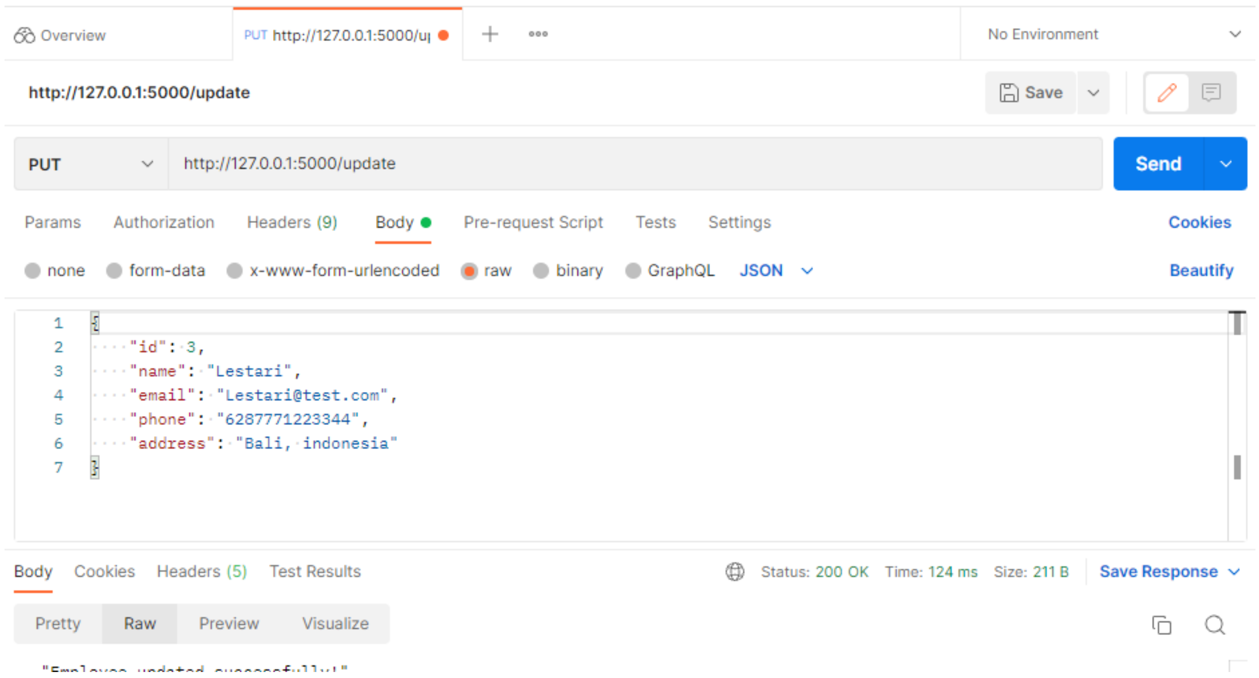
```
"User deleted successfully!"
```


Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key:
+ Options
Check all
With selected:
Edit
Copy
Delete
Export

```
SELECT * FROM `customer`
```

	id	name	email	phone	address
Edit Copy Delete	4	Kinjal	kinjalsingh2003@gmail.com	7678073799	Ghatkopar
Edit Copy Delete	5	Arya	aryatiwari2003@gmail.com	9320408279	Mira Road
Edit Copy Delete	10	Anant	anantsingh1302@gmail.com	8850496186	Malad

API_UPDATE (UPDATE)



http://127.0.0.1:5000/update

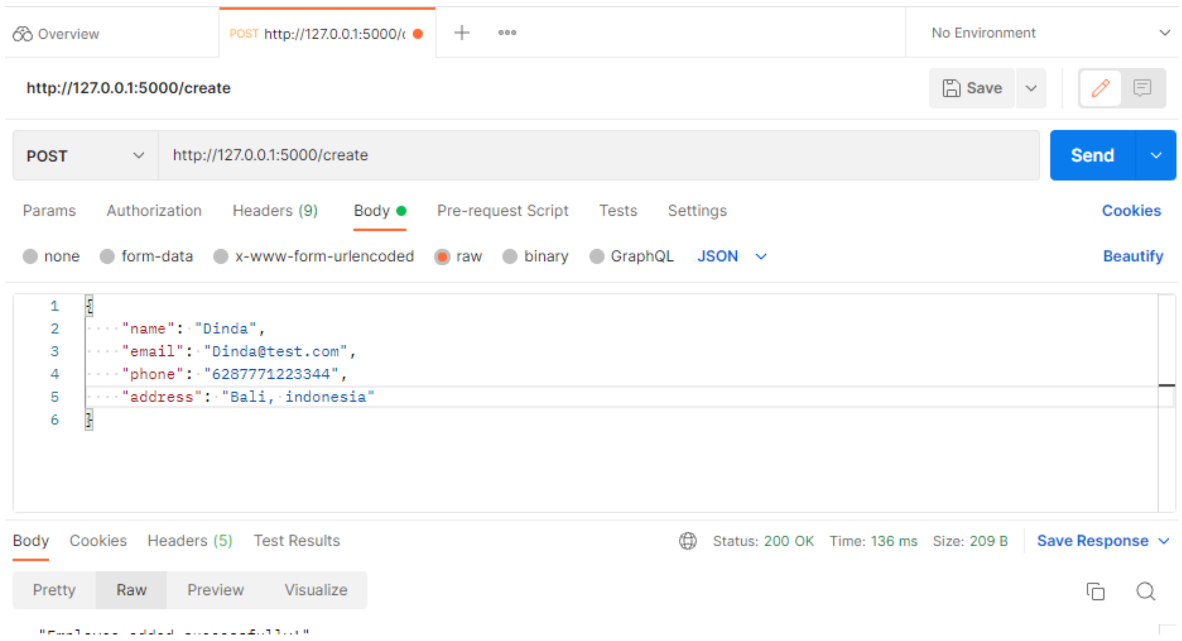
PUT http://127.0.0.1:5000/update

```
{
    "id": 3,
    "name": "Lestari",
    "email": "Lestari@test.com",
    "phone": "6287771223344",
    "address": "Bali, indonesia"
}
```

Status: 200 OK Time: 124 ms Size: 211 B

```
"Employee updated successfully!"
```

API_PUT (CREATE)



http://127.0.0.1:5000/create

POST http://127.0.0.1:5000/create

```
{
    "name": "Dinda",
    "email": "Dinda@test.com",
    "phone": "6287771223344",
    "address": "Bali, indonesia"
}
```

Status: 200 OK Time: 136 ms Size: 209 B

```
"Employee added successfully!"
```