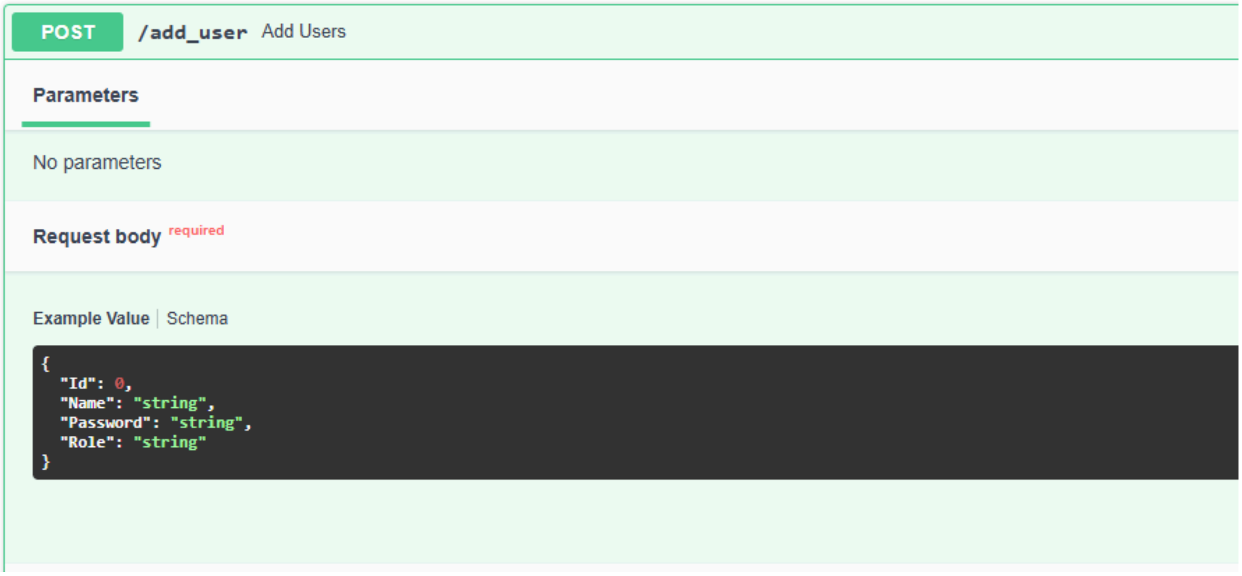
Miniproject 2
Dharunya S

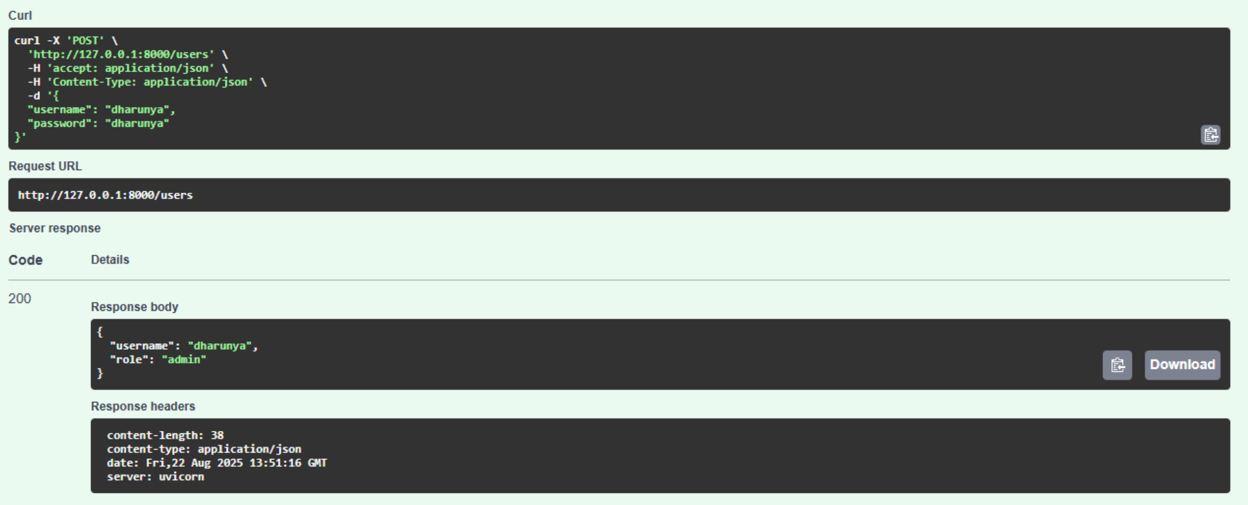
FastApi.py

```
import os
from typing import List
from dotenv import load_dotenv
from fastapi import Depends, FastAPI, Request
from pydantic import BaseModel
from sqlalchemy import Column, Integer, String, create_engine
from sqlalchemy.orm import sessionmaker,declarative_base,Session
from passlib.context import CryptContext
import uvicorn


app=FastAPI()

load_dotenv()
SQLALCHEMY_DATABASE_URL=os.getenv("mini_db")

engine=create_engine(SQLALCHEMY_DATABASE_URL,connect_args={"check_same_thread":False})
session=sessionmaker(autocommit=False,autoflush=False,bind=engine)

pwd_context = CryptContext(schemes=["bcrypt"], deprecated="auto")


base=declarative_base()
class User(base):
   __tablename__='usermin'

   Id=Column(Integer,primary_key=True,index=True )
   Name=Column(String(50),nullable=False)
   Password=Column(String(256),nullable=False)
   Role=Column(String(50),nullable=False)
   base.metadata.create_all(bind=engine)

def get_db():
   db=session()
   try:
      yield db
   finally:
      db.close()
```

```
class users(BaseModel):
    Id:int
    Name:str
    Password:str
    Role:str
    class Config:
        orm_mode=True
def hash_password(password: str) -> str:
    return pwd_context.hash(password)

@app.post('/add_user')
def add_users(us:users,db:Session=Depends(get_db)):
    hashed_pw = hash_password(us.Password)
    us= User(Name=us.Name,Password=hashed_pw,Role=us.Role)
    db.add(us)
    db.commit()
    db.refresh(us)
    return "data inserted"

class UserRequest(BaseModel):
    username: str
    password: str

@app.post("/users")
def get_user(request: UserRequest, db: Session = Depends(get_db)):
    usr = db.query(User).filter(User.Name ==
request.username,User.Password==request.password).first()


    return {
        "name": usr.Name,
        "role": usr.Role
    }

if __name__=="__main__":
    uvicorn.run("mini2:app",host="127.0.0.1",port=8000,reload=True)
```



```
{
  "Id": 0,
  "Name": "string",
  "Password": "string",
  "Role": "string"
}
```

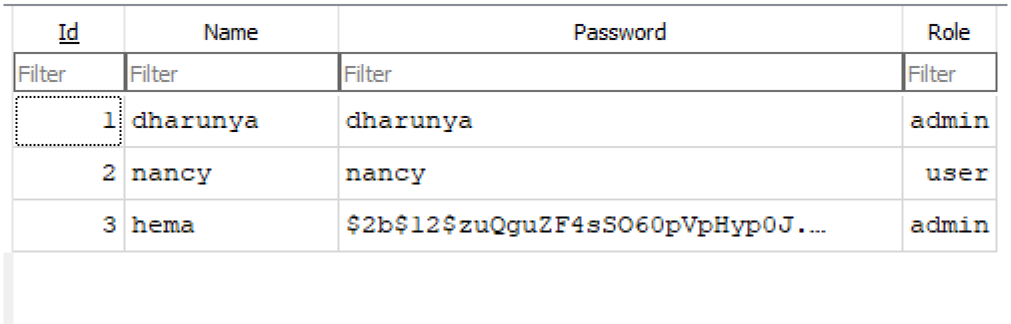


```
curl -X 'POST' \
  'http://127.0.0.1:8000/users' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "username": "dharunya",
  "password": "dharunya"
}'
```

```
http://127.0.0.1:8000/users
```

```
{
  "username": "dharunya",
  "role": "admin"
}
```

```
content-length: 38
content-type: application/json
date: Fri,22 Aug 2025 13:51:16 GMT
server: uvicorn
```



Id	Name	Password	Role
Filter	Filter	Filter	Filter
1	dharunya	dharunya	admin
2	nancy	nancy	user
3	hema	$2b$12$zuQguZF4sSO60pVpHyp0J.…	admin

BankController.cs

using Day12.Security;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Options;

```
using Mini2.Context;
using Mini2.Models;
using Mini2.Security;
using System.Security.Claims;
using System.Text.Json;

namespace Mini2.Controllers
{
    public record LoginRequest(string Username, string Password);
    [ApiController]
    [Route("[controller]")]
    public class BankController : Controller
    {
        MyAppDbContext appDbContext;
        Microsoft.Extensions.Options.IOptions<JWToptions> jwtOptions;

        public BankController(MyAppDbContext ctx,
Microsoft.Extensions.Options.IOptions<JWToptions> _jwtOptions)
        {
            appDbContext = ctx;
            jwtOptions = _jwtOptions;
        }

        [Authorize(Roles ="Admin")]
        [HttpPost("AddCustomer")]
        public IActionResult AddCustomer(CustomerDet cust)
        {
            appDbContext.CustomerDets.Add(cust);
            appDbContext.SaveChanges();
            return Ok("Customer Added");
        }
        [Authorize(Roles = "Admin")]
        [HttpPost("AddAccount")]
        public IActionResult AddAccount(CustomerAndAccountEntry caentry)
        {
            var customer = appDbContext.CustomerDets.Where(x => x.Name ==
caentry.customerName).FirstOrDefault();
            if (customer == null)
            {
                return NotFound("Customer Not Found");
            }
            AccountDet acc = new AccountDet
            {
                accountNumber = caentry.accno,
                amount_in_account = caentry.amount_in_account,
                acc_created_date = caentry.acc_created_date,
                CustomerId = customer.Id,
                Type = caentry.Type

            };
            appDbContext.AccountDets.Add(acc);
            appDbContext.SaveChanges();
            return Ok("Account Added");

        }
        const string ADMIN = "admin";

        [HttpPost("getLoginusingapi")]

        public async Task<IActionResult> GetLogin(LoginRequest login)
        {

            var httpClient = new HttpClient();
            var response = await httpClient.PostAsJsonAsync("http://127.0.0.1:8000/users",
login);
            var user = await response.Content.ReadFromJsonAsync<User>();
            if (user == null)
            {
                return Unauthorized("sorry not authorized");
            }

            else
            {
                var claims = new List<Claim>
                {
                    new Claim(ClaimTypes.Name,login.Username),

                };
                if (user.Role.Trim().ToLower().Contains(ADMIN))
                {
                    claims.Add(new Claim(ClaimTypes.Role, "Admin"));
                }
                var token = jwtService.CreateJwtToken(jwtOptions.Value, claims);
                return Ok(token);
            }


        }
        [HttpGet("GetTotalBalance")]
        public IActionResult GetTotalBalancebycustomer(string name)
        {
            var customer =
appDbContext.AccountDets.Where(x=>x.CustomerDet.Name==name).Sum(y=>y.amount_
in_account);

            return Ok(customer);
        }

    }

}
```

MyAppDbContext.cs

```
using Microsoft.EntityFrameworkCore;
using Mini2.Models;

namespace Mini2.Context
{
    public class MyAppDbContext: DbContext
    {
        public MyAppDbContext(DbContextOptions<MyAppDbContext> options) :
base(options) { }

        public DbSet<CustomerDet> CustomerDets { get; set; }
        public DbSet<AccountDet> AccountDets { get; set; }
    }
}
```

AccountDet.cs

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace Mini2.Models
{
    public class AccountDet
    {
        [Key]
        public int accountNumber { get; set; }
        public int amount_in_account { get; set; }
        public DateOnly acc_created_date { get; set; }
        public string Type { get; set; }

        public int CustomerId { get; set; }
        [ForeignKey("CustomerId")]

        public CustomerDet CustomerDet { get; set; }
    }
}
```

CustomerDet.cs

```
using System.ComponentModel.DataAnnotations;

namespace Mini2.Models
{
    public class CustomerDet
    {
        [Key]
        public int Id { get; set; }
        public string Name { get; set; }
        public int Age { get; set; }
    }
}
```

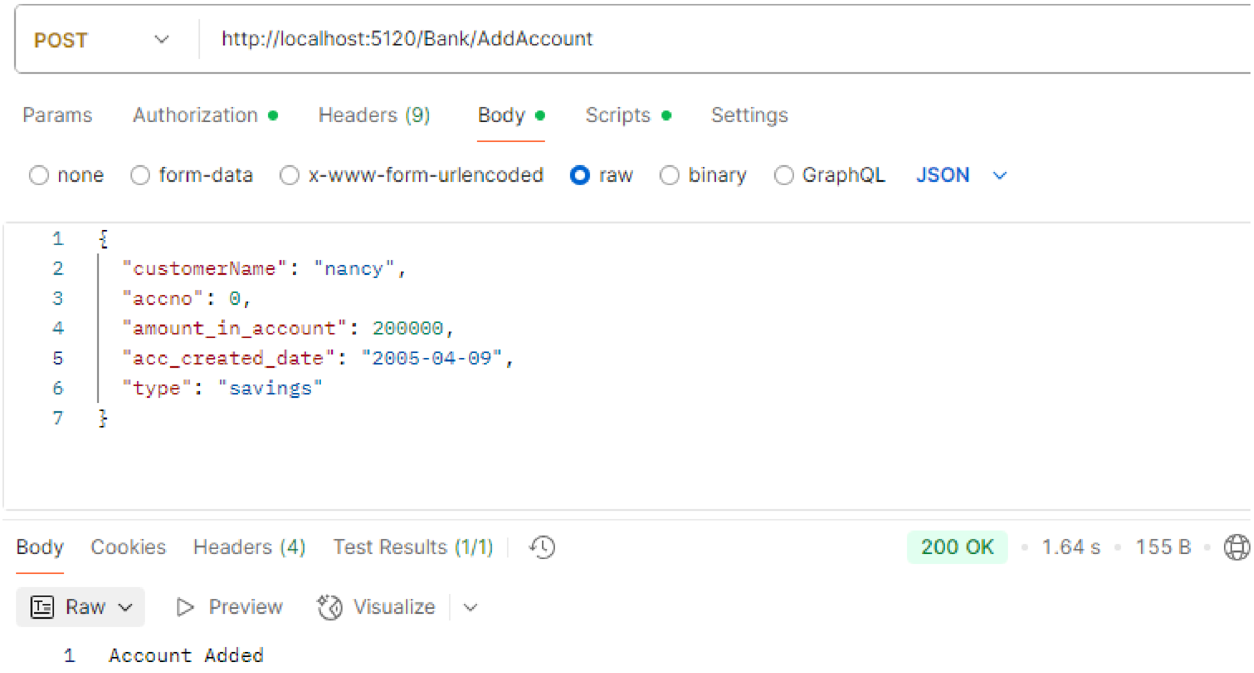
CustomerAndAccountEntry.cs

```
namespace Mini2.Models
{
    public class CustomerAndAccountEntry
    {
        public string customerName { get; set; }
        public int accno { get; set; }

        public int amount_in_account { get; set; }
        public DateOnly acc_created_date { get; set; }
        public string Type { get; set; }


    }
}
```

User.cs

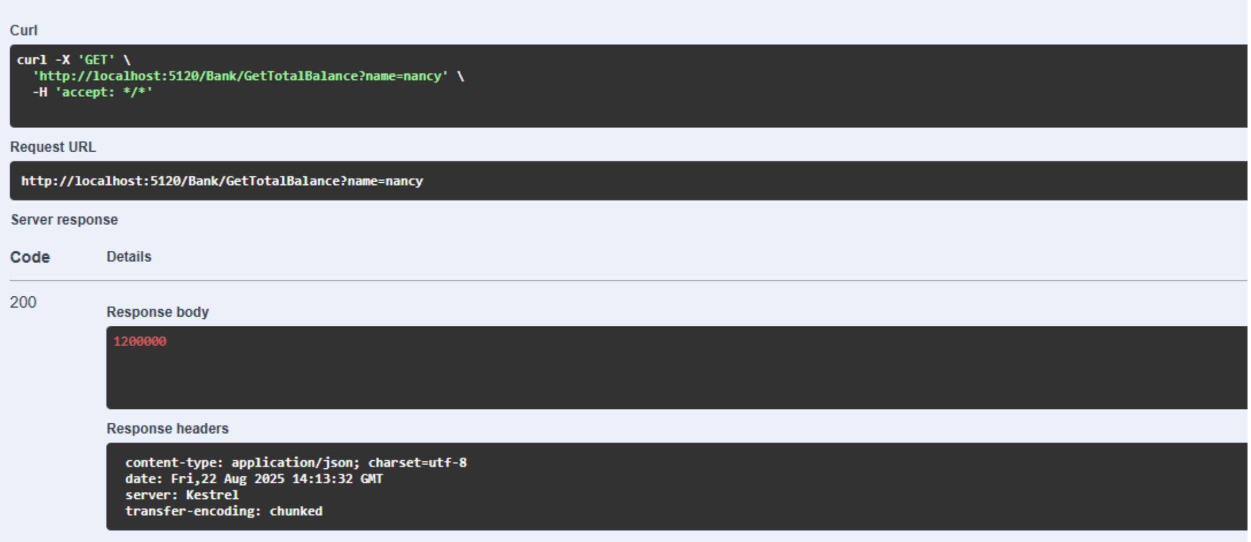
```
namespace Mini2.Models
{
    public class User
```

```
  {
     public string Name { get; set; }

     public string Role { get; set; }

  }
}
```



```
{
  "customerName": "nancy",
  "accno": 0,
  "amount_in_account": 200000,
  "acc_created_date": "2005-04-09",
  "type": "savings"
}
```

```
Account Added
```



Curl

```
curl -X 'GET' \
  'http://localhost:5120/Bank/GetTotalBalance?name=nancy' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:5120/Bank/GetTotalBalance?name=nancy
```

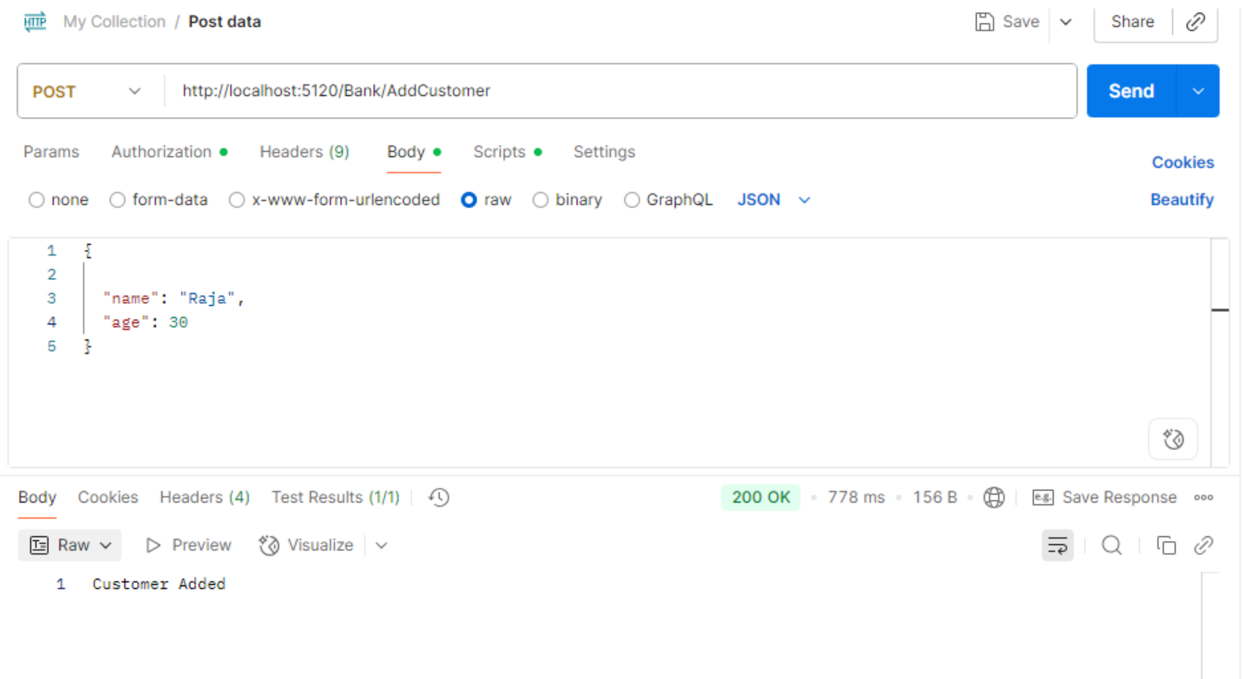
Server response

Code	Details
200	Response body

```
1200000
```

Response headers

```
content-type: application/json; charset=utf-8
date: Fri,22 Aug 2025 14:13:32 GMT
server: Kestrel
transfer-encoding: chunked
```



My Collection / Post data

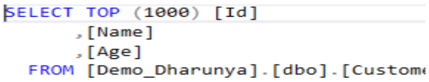
POST http://localhost:5120/Bank/AddCustomer

Body: ○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

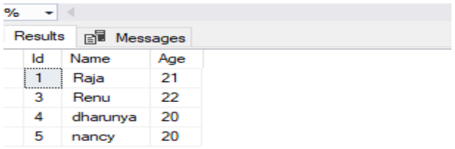
```
{

  "name": "Raja",
  "age": 30
}
```

200 OK • 778 ms • 156 B

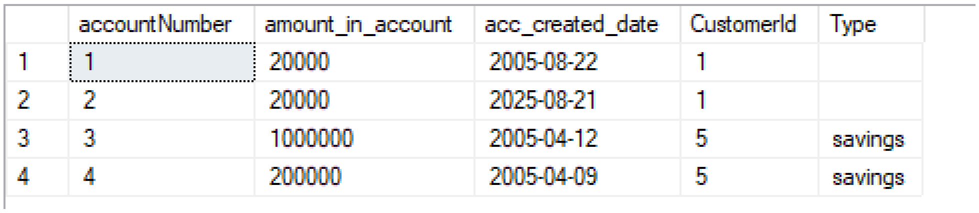
```
Customer Added
```



```
SELECT TOP (1000) [Id]
      ,[Name]
      ,[Age]
  FROM [Demo_Dharunya].[dbo].[Custome
```



	Id	Name	Age
	1	Raja	21
	3	Renu	22
	4	dharunya	20
	5	nancy	20



	accountNumber	amount_in_account	acc_created_date	CustomerId	Type
1	1	20000	2005-08-22	1	
2	2	20000	2025-08-21	1	
3	3	1000000	2005-04-12	5	savings
4	4	200000	2005-04-09	5	savings