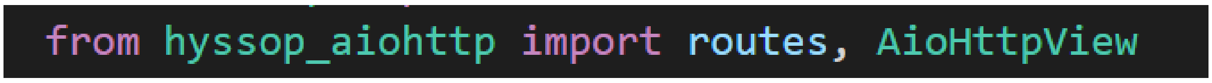
實作功能介紹
大綱

➢hyssop接收參數功能
➢project_config.yml設定
➢實作sqlalchemy_orm 操作database資料庫
➢透過hyssop.project.component實作註冊功能class
➢透過註冊功能class實作註冊API
➢透過hyssop.project.component實作登入功能class
➢透過登入功能class實作登入API
➢alembic 資料庫版本管理功能
➢app目錄結構

—-------------------------------
➢project_config.yml設定

```yaml
name: hyssop Server
port: 4444
debug: False
doc:
  description: haha api
cors:
  - origin: '*'
    allow_credentials: True
    expose_headers: '*'
    allow_headers: '*'
component:
  Register:
    p1: 'This is p1pppp'

  Login:
    p1: 'This is Loginp1pppp'
controller:
  /haha_worldxxxwww:
    enum: haha_worldxxxwww
aiohttpDDDD:
   route_decoratorsCCC:
    - 'TTT_view'
```

➢hyssop接收參數功能:
以server2/sipass_api為模板做編輯
實作功能介紹:
透過**hyssop_aiohttp**的 routes , AioHttpView 接收http client request參數



```
from hyssop_aiohttp import routes, AioHttpView
```

接收參數3種函式 get_argument，get_arguments_dict(['k1', 'k2','k3'])，match_info.get('key')

1 ➛實作AioHttpView 接收http client request 參數
1-1 AioHttpView & get_argument('kkk') /GET method
ex: ⤏phone_no = await self.get_argument('kkk',default='預設參數值')
ex:http://localhost:4444/haha_worldxxxwww?kkk=022222 #接收kkk的參數 並回應response

step1 編輯project_config.yml

```
controller:
  /haha_worldxxxwww:
    enum: haha_worldxxxwww
```

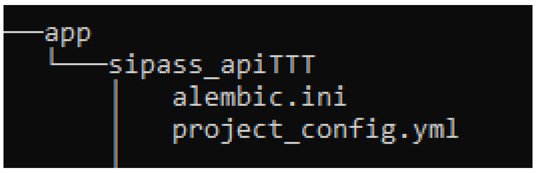
step2 編輯...\controller\__init__.py"

```
CCCViewController = ('haha_worldxxxwww','model' , 'KKView')
```

step3 編輯...\controller\model.py

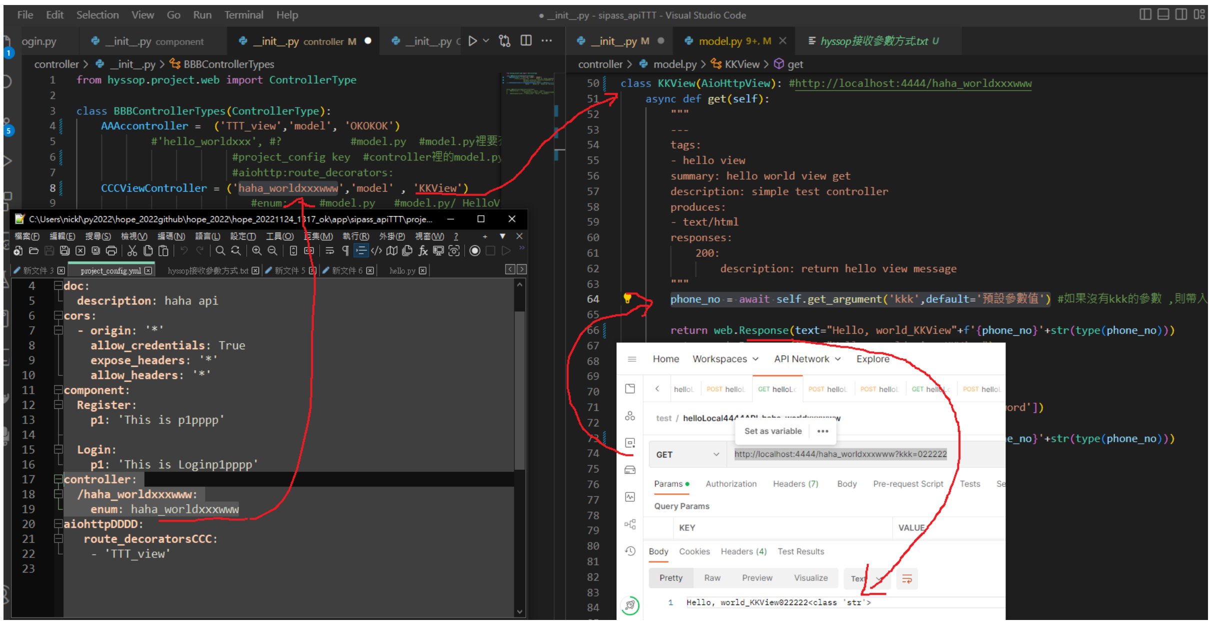
```
phone_no = await self.get_argument('kkk',default='預設參數值')
return web.Response(text="Hello, world_KKView"+f'{phone_no}'+str(type(phone_no)))
```

step4 指令 啟動server
app> python -m hyssop_aiohttp start sipass_apiTTT



```
──app
  └───sipass_apiTTT
      │    alembic.ini
      │    project_config.yml
      │
```

step5 postman 測試 http://localhost:4444/haha_worldxxxwww?kkk=022222 /GET method



```
from hyssop.project.web import ControllerType

class BBBControllerTypes(ControllerType):
    AAAcontroller =  ('TTT_view','model', 'OKOKOK')
            #'hello_worldxxx', #?            #model.py  #model.py裡要
                                  #project_config key  #controller裡的model.py
                                  #aiohttp:route_decorators:
    CCCViewController = ('haha_worldxxxwww','model' , 'KKView')
                                     #enum:        #model.py   #model.py/ HelloV
```

```
doc:
  description: haha api
cors:
  - origin: '*'
    allow_credentials: True
    expose_headers: '*'
    allow_headers: '*'
component:
  Register:
    p1: 'This is p1pppp'

  Login:
    p1: 'This is Loginp1pppp'
controller:
  /haha_worldxxxwww:
    enum: haha_worldxxxwww
aiohttpDDDD:
  route_decoratorsCCC:
   - 'TTT_view'
```

```
class KKView(AioHttpView): #http://localhost:4444/haha_worldxxxwww
    async def get(self):
        """
        ---
        tags:
        - hello view
        summary: hello world view get
        description: simple test controller
        produces:
        - text/html
        responses:
            200:
                description: return hello view message
        """
        phone_no = await self.get_argument('kkk',default='預設參數值') #如果沒有kkk的參數 ,則帶入

        return web.Response(text="Hello, world_KKView"+f'{phone_no}'+str(type(phone_no)))
```

GET http://localhost:4444/haha_worldxxxwww?kkk=022222

```
Hello, world_KKView022222<class 'str'>
```

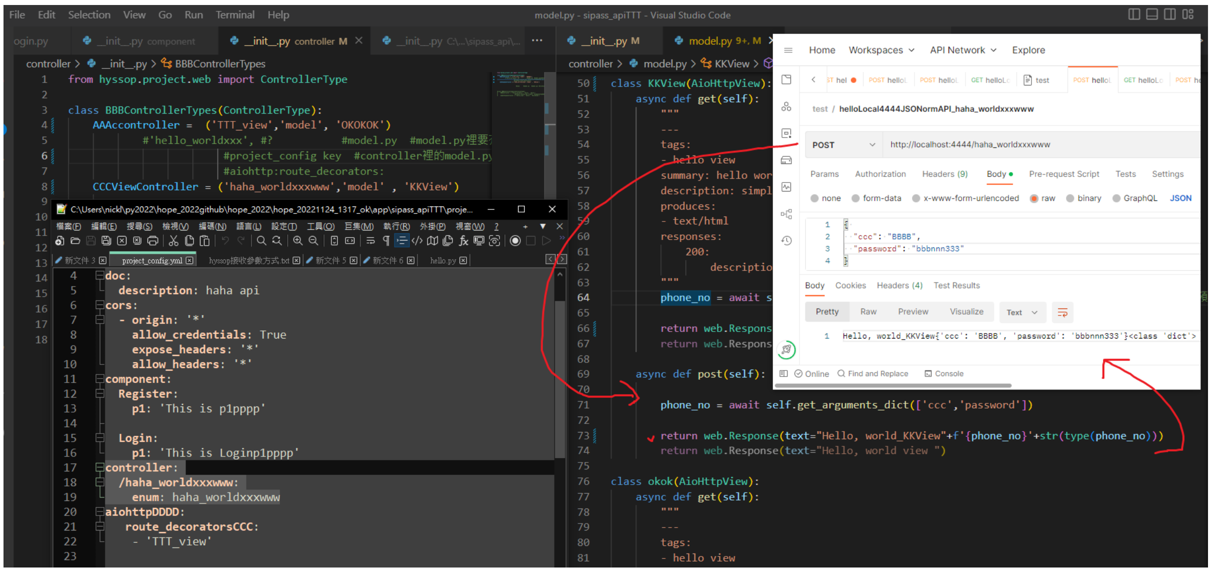
1-2 ➛AioHttpView & POST method

⤏ phone_no = await self.get_arguments_dict(['ccc','password'])

```
phone_no = await self.get_arguments_dict(['ccc','password'])
return web.Response(text="Hello, world_KKView"+f'{phone_no}'+str(type(phone_no)))
```

⤏ 啟動server指令 app> python -m hyssop_aiohttp start sipass_apiTTT

⤏ postman 測試 http://localhost:4444/haha_worldxxxwww /POST method



POST http://localhost:4444/haha_worldxxxwww

```
{
  "ccc": "BBBB",
  "password": "bbbnnn333"
}
```

```
Hello, world_KKView{'ccc': 'BBBB', 'password': 'bbbnnn333'}<class 'dict'>
```

```
    async def post(self):

        phone_no = await self.get_arguments_dict(['ccc','password'])

        return web.Response(text="Hello, world_KKView"+f'{phone_no}'+str(type(phone_no)))
        return web.Response(text="Hello, world view ")

class okok(AioHttpView):
    async def get(self):
        """
        ---
        tags:
        - hello view
```

2➛實作routes 接收http client request 參數

2-1 routes & GET method

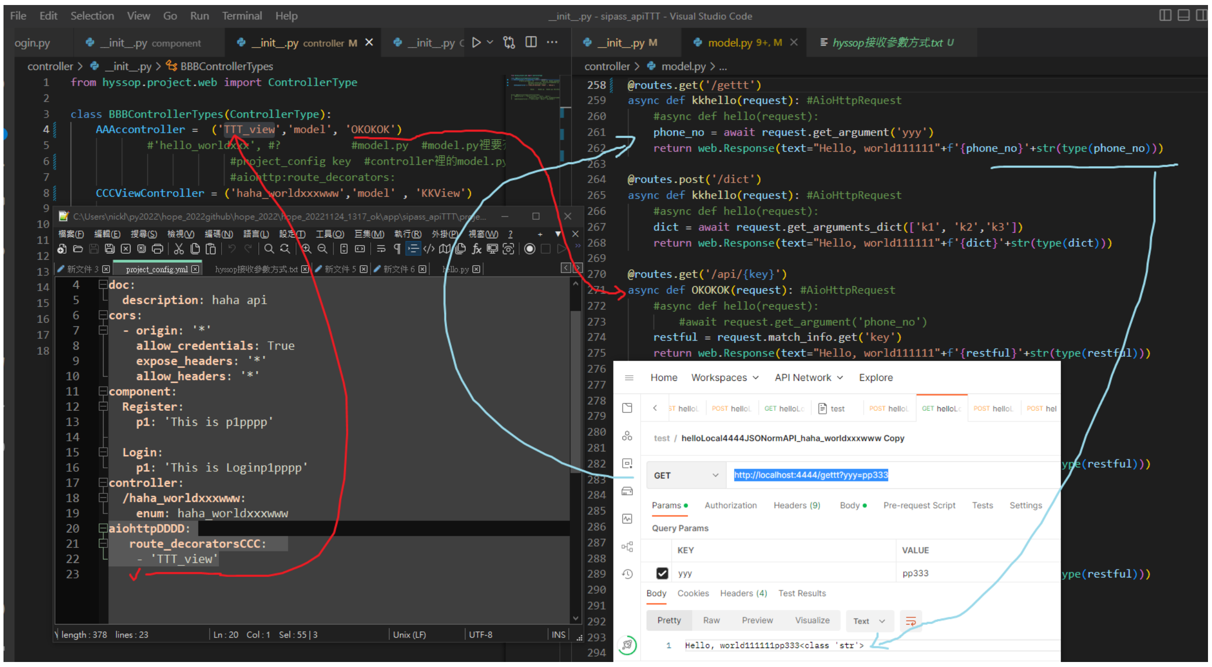
2-1-1 ⤑ @routes.get('/gettt')

⤑ phone_no = await request.get_argument('yyy')

```
phone_no = await request.get_argument('yyy')
return web.Response(text= "Hello, world111111"+f'{phone_no}'+str(type(phone_no)))
```

⤑ 啟動server指令 app> python -m hyssop_aiohttp start sipass_apiTTT

⤑ postman 測試 http://localhost:4444/haha_worldxxxwww /POST method



File Edit Selection View Go Run Terminal Help — __init__.py - sipass_apiTTT - Visual Studio Code

controller > __init__.py > BBBControllerTypes

```python
from hyssop.project.web import ControllerType

class BBBControllerTypes(ControllerType):
    AAAcontroller =  ('TTT_view','model', 'OKOKOK')
            #'hello_worldxxx', #?            #model.py  #model.py裡要
                                #project_config key  #controller裡的model.py
                                #aiohttp:route_decorators:
    CCCViewController = ('haha_worldxxxwww','model' , 'KKView')
```

C:\Users\nickl\py2022\hope_2022github\hope_2022\hope_20221124_1317_ok\app\sipass_apiTTT\proje...

project_config.yml

```yaml
doc:
  description: haha api
cors:
  - origin: '*'
    allow_credentials: True
    expose_headers: '*'
    allow_headers: '*'
component:
  Register:
    p1: 'This is p1pppp'

  Login:
    p1: 'This is Loginp1pppp'
controller:
  /haha_worldxxxwww:
    enum: haha_worldxxxwww
aiohttpDDDD:
   route_decoratorsCCC:
    - 'TTT_view'
```

length : 378   lines : 23   Ln : 20   Col : 1   Sel : 55 | 3   Unix (LF)   UTF-8   INS

controller > model.py > ...

```python
@routes.get('/gettt')
async def kkhello(request): #AioHttpRequest
    #async def hello(request):
    phone_no = await request.get_argument('yyy')
    return web.Response(text="Hello, world111111"+f'{phone_no}'+str(type(phone_no)))

@routes.post('/dict')
async def kkhello(request): #AioHttpRequest
    #async def hello(request):
    dict = await request.get_arguments_dict(['k1', 'k2','k3'])
    return web.Response(text="Hello, world111111"+f'{dict}'+str(type(dict)))

@routes.get('/api/{key}')
async def OKOKOK(request): #AioHttpRequest
    #async def hello(request):
        #await request.get_argument('phone_no')
    restful = request.match_info.get('key')
    return web.Response(text="Hello, world111111"+f'{restful}'+str(type(restful)))
```

test / helloLocal4444JSONormAPI_haha_worldxxxwww Copy

GET http://localhost:4444/gettt?yyy=pp333

Query Params

KEY	VALUE
✓ yyy	pp333

```
1  Hello, world111111pp333<class 'str'>
```

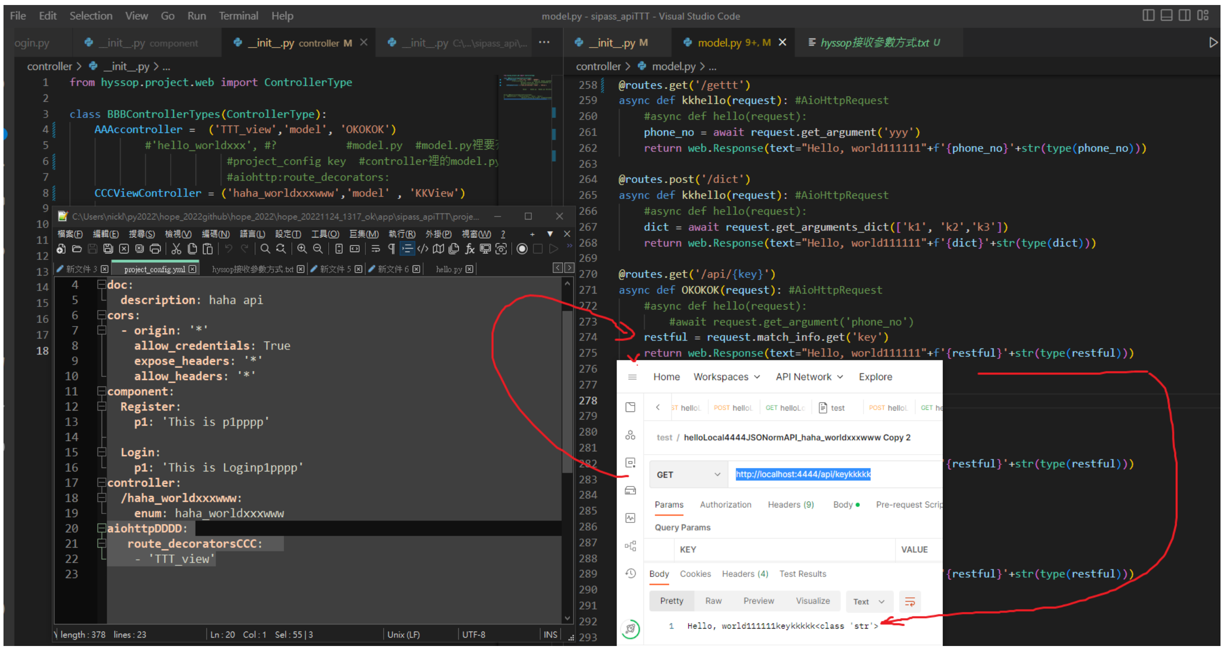
2-1-2 ⤑ @routes.get('/api/{key}')

⤑ restful = request.match_info.get('key')

```
restful = request.match_info.get('key')
return web.Response(text="Hello, world111111"+f'{restful}'+str(type(restful)))
```

⤑ 啟動server指令 app> python -m hyssop_aiohttp start sipass_apiTTT

⤑ postman 測試 http://localhost:4444/gettt?yyy=pp333 /GET method



File Edit Selection View Go Run Terminal Help — model.py - sipass_apiTTT - Visual Studio Code

```python
from hyssop.project.web import ControllerType

class BBBControllerTypes(ControllerType):
    AAAcontroller =  ('TTT_view','model', 'OKOKOK')
            #'hello_worldxxx', #?            #model.py  #model.py裡要
                                #project_config key  #controller裡的model.py
                                #aiohttp:route_decorators:
    CCCViewController = ('haha_worldxxxwww','model' , 'KKView')
```

```yaml
doc:
  description: haha api
cors:
  - origin: '*'
    allow_credentials: True
    expose_headers: '*'
    allow_headers: '*'
component:
  Register:
    p1: 'This is p1pppp'

  Login:
    p1: 'This is Loginp1pppp'
controller:
  /haha_worldxxxwww:
    enum: haha_worldxxxwww
aiohttpDDDD:
   route_decoratorsCCC:
    - 'TTT_view'
```

```python
@routes.get('/gettt')
async def kkhello(request): #AioHttpRequest
    #async def hello(request):
    phone_no = await request.get_argument('yyy')
    return web.Response(text="Hello, world111111"+f'{phone_no}'+str(type(phone_no)))

@routes.post('/dict')
async def kkhello(request): #AioHttpRequest
    #async def hello(request):
    dict = await request.get_arguments_dict(['k1', 'k2','k3'])
    return web.Response(text="Hello, world111111"+f'{dict}'+str(type(dict)))

@routes.get('/api/{key}')
async def OKOKOK(request): #AioHttpRequest
    #async def hello(request):
        #await request.get_argument('phone_no')
    restful = request.match_info.get('key')
    return web.Response(text="Hello, world111111"+f'{restful}'+str(type(restful)))
```

test / helloLocal4444JSONormAPI_haha_worldxxxwww Copy 2

GET http://localhost:4444/api/keykkkkk

```
1  Hello, world111111keykkkkk<class 'str'>
```

2-2 ➝ routes & POST method

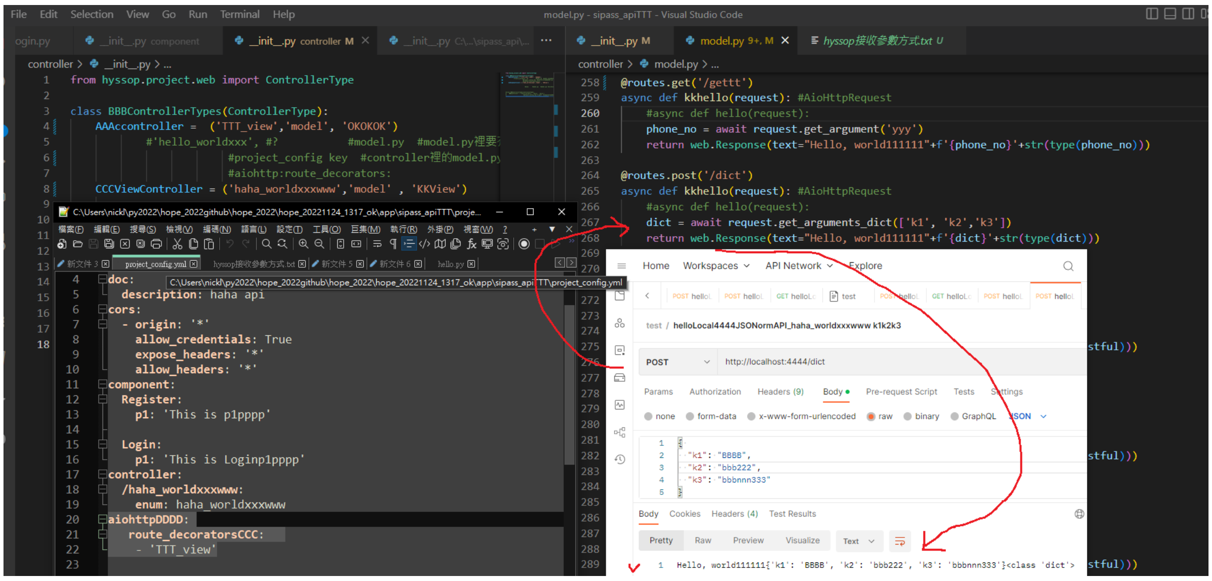
2-2-1 ⇢ @routes.post('/dict')

⇢ dict = await request.get_arguments_dict(['k1', 'k2','k3'])

```
dict = await request.get_arguments_dict(['k1', 'k2','k3'])
return web.Response(text="Hello, world111111"+f'{dict}'+str(type(dict)))
```

⇢ 啟動server指令 app> python -m hyssop_aiohttp start sipass_apiTTT

⇢ postman 測試 http://localhost:4444/dict /POST method


File Edit Selection View Go Run Terminal Help — model.py - sipass_apiTTT - Visual Studio Code

controller > _init_.py > ...

```
from hyssop.project.web import ControllerType

class BBBControllerTypes(ControllerType):
    AAAcontroller =  ('TTT_view','model', 'OKOKOK')
            #'hello_worldxxx', #?          #model.py  #model.py裡要
                      #project_config key  #controller裡的model.py
                      #aiohttp:route_decorators:
    CCCViewController = ('haha_worldxxxwww','model' , 'KKView')
```

C:\Users\nickl\py2022\hope_2022github\hope_2022\hope_20221124_1317_ok\app\sipass_apiTTT\project_config.yml

```
doc:
  description: haha api
cors:
  - origin: '*'
    allow_credentials: True
    expose_headers: '*'
    allow_headers: '*'
component:
  Register:
    p1: 'This is p1pppp'

  Login:
    p1: 'This is Loginp1pppp'
controller:
  /haha_worldxxxwww:
    enum: haha_worldxxxwww
aiohttpDDDD:
   route_decoratorsCCC:
    - 'TTT_view'
```

controller > model.py > ...

```
@routes.get('/gettt')
async def kkhello(request): #AioHttpRequest
    #async def hello(request):
    phone_no = await request.get_argument('yyy')
    return web.Response(text="Hello, world111111"+f'{phone_no}'+str(type(phone_no)))

@routes.post('/dict')
async def kkhello(request): #AioHttpRequest
    #async def hello(request):
    dict = await request.get_arguments_dict(['k1', 'k2','k3'])
    return web.Response(text="Hello, world111111"+f'{dict}'+str(type(dict)))
```

test / helloLocal4444JSONormAPI_haha_worldxxxwww k1k2k3

POST http://localhost:4444/dict

```
{
  "k1": "BBBB",
  "k2": "bbb222",
  "k3": "bbbnnn333"
}
```

```
Hello, world111111{'k1': 'BBBB', 'k2': 'bbb222', 'k3': 'bbbnnn333'}<class 'dict'>
```

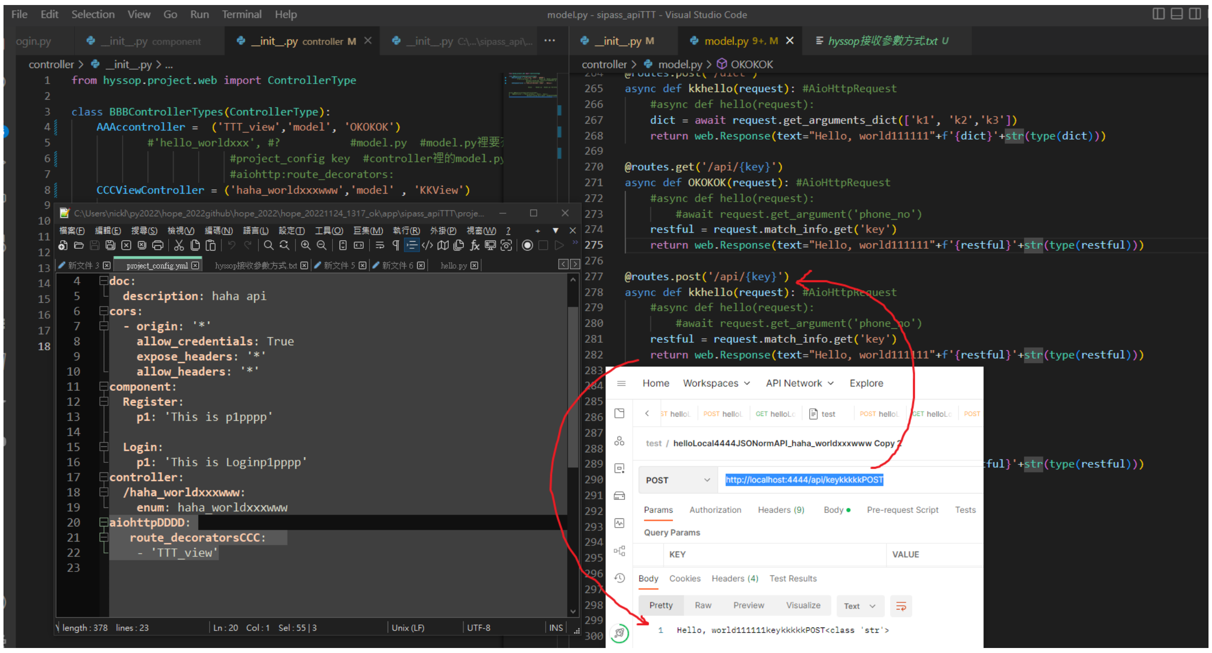
2-2-2 ⇢ @routes.post('/api/{key}')

restful = request.match_info.get('key')

```
restful = request.match_info.get('key')
return web.Response(text="Hello, world111111"+f'{restful}'+str(type(restful)))
```

⇢ 啟動server指令 app> python -m hyssop_aiohttp start sipass_apiTTT

⇢ postman 測試 http://localhost:4444/api/keykkkkkPOST /POST method


controller > model.py > OKOKOK

```
async def kkhello(request): #AioHttpRequest
    #async def hello(request):
    dict = await request.get_arguments_dict(['k1', 'k2','k3'])
    return web.Response(text="Hello, world111111"+f'{dict}'+str(type(dict)))

@routes.get('/api/{key}')
async def OKOKOK(request): #AioHttpRequest
    #async def hello(request):
        #await request.get_argument('phone_no')
    restful = request.match_info.get('key')
    return web.Response(text="Hello, world111111"+f'{restful}'+str(type(restful)))

@routes.post('/api/{key}')
async def kkhello(request): #AioHttpRequest
    #async def hello(request):
        #await request.get_argument('phone_no')
    restful = request.match_info.get('key')
    return web.Response(text="Hello, world111111"+f'{restful}'+str(type(restful)))
```

test / helloLocal4444JSONormAPI_haha_worldxxxwww Copy 2

POST http://localhost:4444/api/keykkkkkPOST

Query Params — KEY VALUE

```
Hello, world111111keykkkkkPOST<class 'str'>
```

length : 378 lines : 23 Ln : 20 Col : 1 Sel : 55 | 3 Unix (LF) UTF-8 INS

➢實作sqlalchemy_orm 操作database資料庫

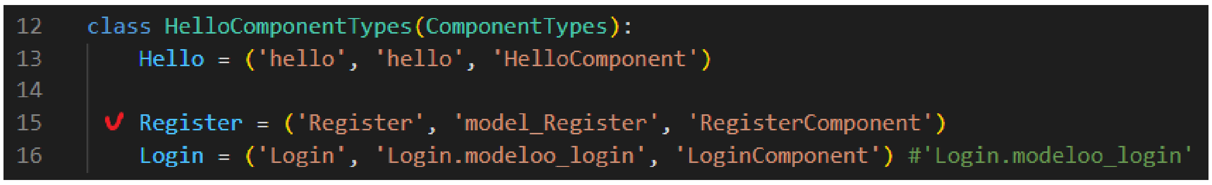
➛查詢資料　`username_exist =  session.query(Signup)`

➛增加資料　`session.add(Signup(**datas))`

➛篩選資料　`filter(Signup.username == datas['username'],)`

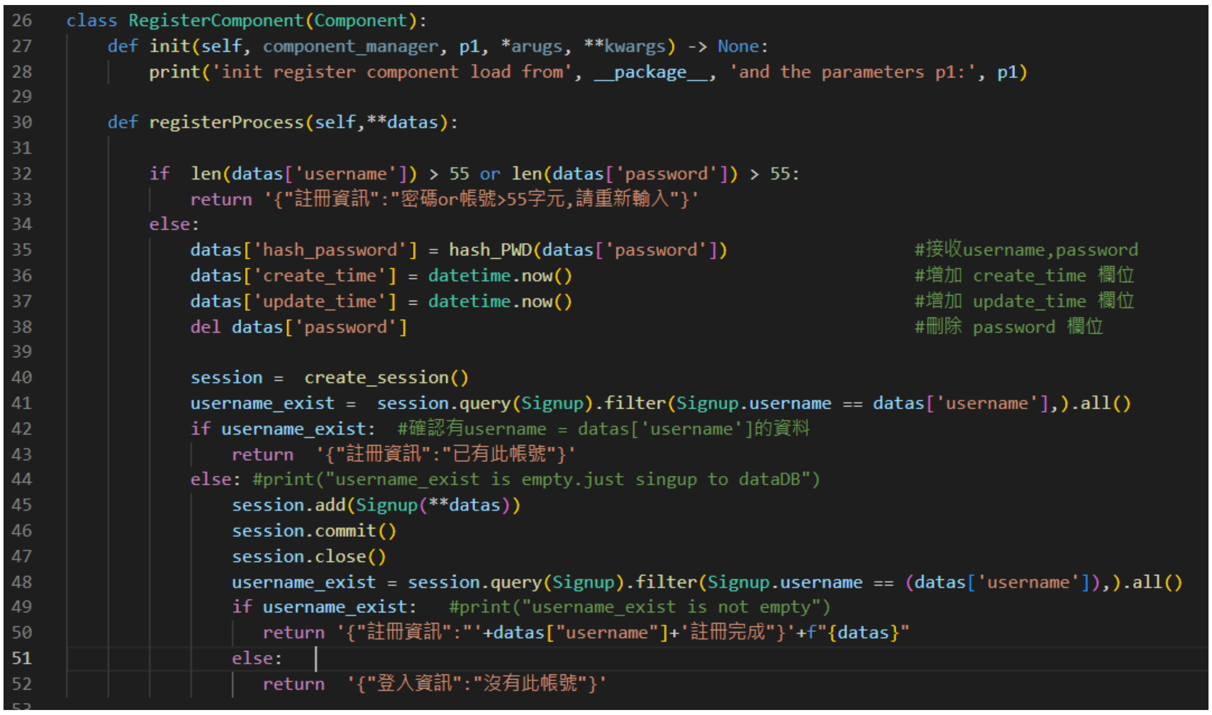
➢透過hyssop.project.component實作註冊功能class

➛component/__init__.py



```python
class HelloComponentTypes(ComponentTypes):
    Hello = ('hello', 'hello', 'HelloComponent')

    Register = ('Register', 'model_Register', 'RegisterComponent')
    Login = ('Login', 'Login.modeloo_login', 'LoginComponent') #'Login.modeloo_login'
```

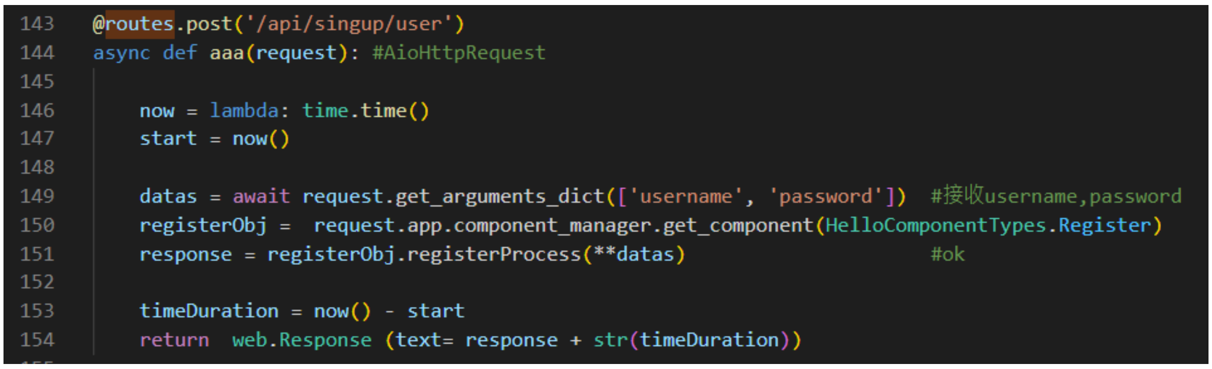
➛component/model_Register.py



```python
class RegisterComponent(Component):
    def init(self, component_manager, p1, *arugs, **kwargs) -> None:
        print('init register component load from', __package__, 'and the parameters p1:', p1)

    def registerProcess(self,**datas):

        if  len(datas['username']) > 55 or len(datas['password']) > 55:
            return '{"註冊資訊":"密碼or帳號>55字元,請重新輸入"}'
        else:
            datas['hash_password'] = hash_PWD(datas['password'])            #接收username,password
            datas['create_time'] = datetime.now()                           #增加 create_time 欄位
            datas['update_time'] = datetime.now()                           #增加 update_time 欄位
            del datas['password']                                           #刪除 password 欄位

            session =  create_session()
            username_exist =  session.query(Signup).filter(Signup.username == datas['username'],).all()
            if username_exist:  #確認有username = datas['username']的資料
                return  '{"註冊資訊":"已有此帳號"}'
            else: #print("username_exist is empty.just singup to dataDB")
                session.add(Signup(**datas))
                session.commit()
                session.close()
                username_exist = session.query(Signup).filter(Signup.username == (datas['username']),).all()
                if username_exist:   #print("username_exist is not empty")
                   return '{"註冊資訊":"'+datas["username"]+'註冊完成"}'+f"{datas}"
                else:
                   return  '{"登入資訊":"沒有此帳號"}'
```

➢透過註冊功能class實作註冊API

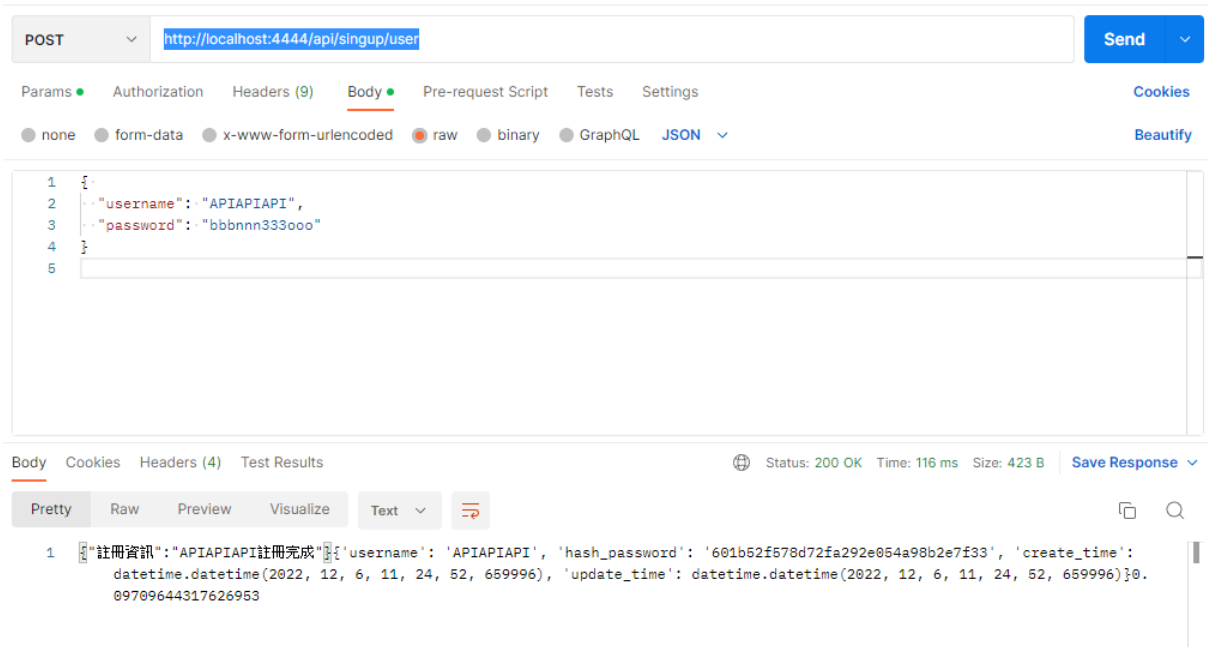
@routes.post('/api/singiup/user')



```python
@routes.post('/api/singup/user')
async def aaa(request): #AioHttpRequest

    now = lambda: time.time()
    start = now()

    datas = await request.get_arguments_dict(['username', 'password'])  #接收username,password
    registerObj =  request.app.component_manager.get_component(HelloComponentTypes.Register)
    response = registerObj.registerProcess(**datas)                        #ok

    timeDuration = now() - start
    return  web.Response (text= response + str(timeDuration))
```

➢postman 註冊API測試 http://localhost:4444/api/singup/user /POST method

第1次註冊API測試,註冊帳號



POST http://localhost:4444/api/singup/user

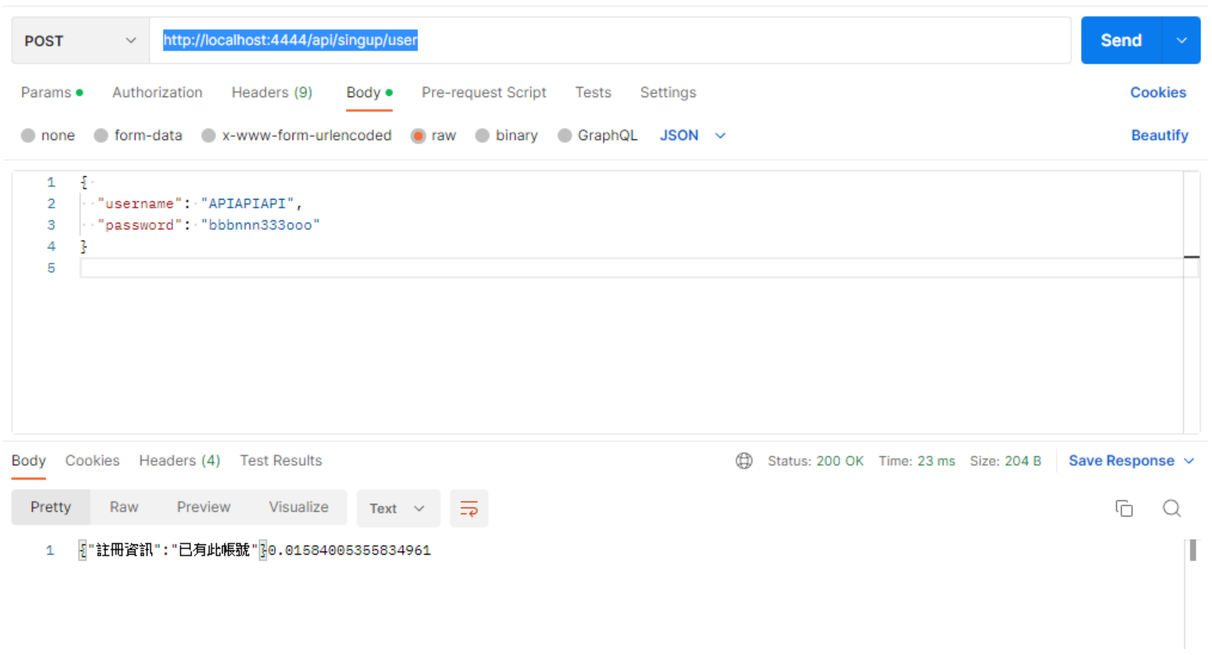
Body: raw JSON

```
{
  "username": "APIAPIAPI",
  "password": "bbbnnn333ooo"
}
```

Status: 200 OK Time: 116 ms Size: 423 B

```
{"註冊資訊":"APIAPIAPI註冊完成"}{'username': 'APIAPIAPI', 'hash_password': '601b52f578d72fa292e054a98b2e7f33', 'create_time':
    datetime.datetime(2022, 12, 6, 11, 24, 52, 659996), 'update_time': datetime.datetime(2022, 12, 6, 11, 24, 52, 659996)}0.
    09709644317626953
```

第2次註冊API測試,是否已有此username帳號



POST http://localhost:4444/api/singup/user

Body: raw JSON

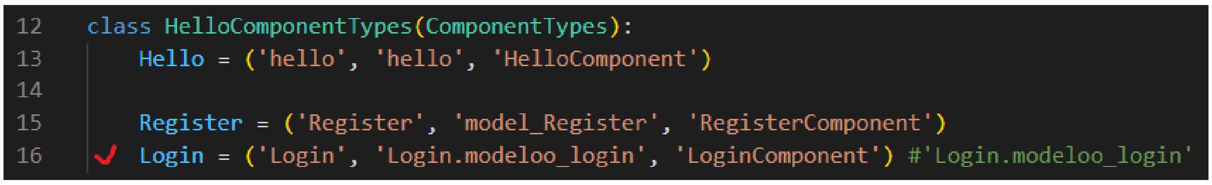
```
{
  "username": "APIAPIAPI",
  "password": "bbbnnn333ooo"
}
```

Status: 200 OK Time: 23 ms Size: 204 B

```
{"註冊資訊":"已有此帳號"}0.015840053558349961
```

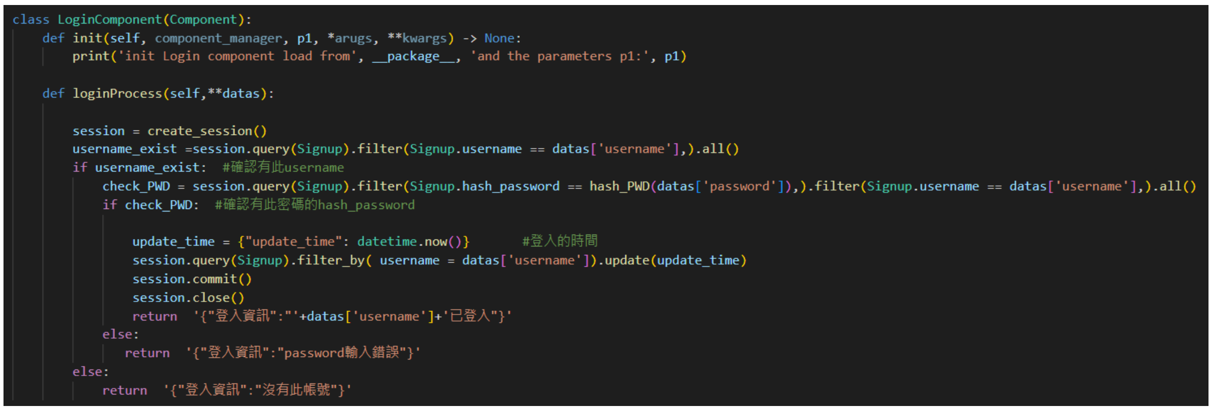
➢透過hyssop.project.component實作登入功能class

➛component/__init__.py



```python
class HelloComponentTypes(ComponentTypes):
    Hello = ('hello', 'hello', 'HelloComponent')

    Register = ('Register', 'model_Register', 'RegisterComponent')
    Login = ('Login', 'Login.modeloo_login', 'LoginComponent') #'Login.modeloo_login'
```

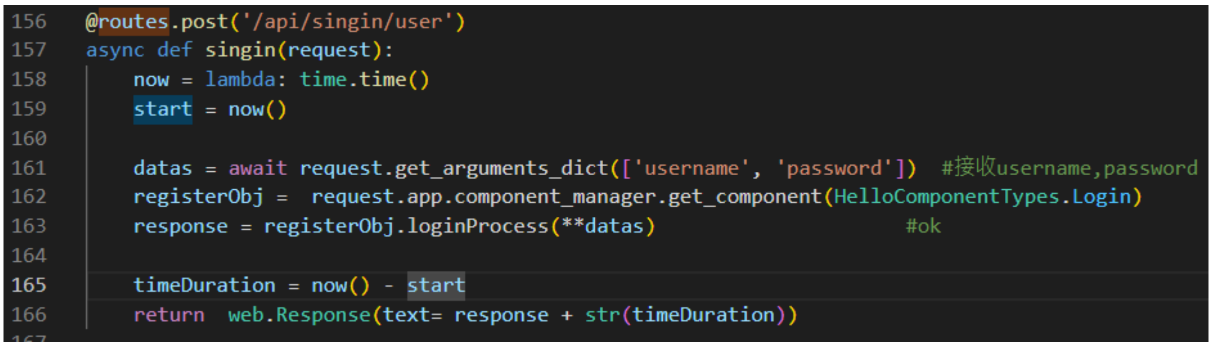
➛component\Login\modeloo_login.py



```python
class LoginComponent(Component):
    def init(self, component_manager, p1, *arugs, **kwargs) -> None:
        print('init Login component load from', __package__, 'and the parameters p1:', p1)

    def loginProcess(self,**datas):

        session = create_session()
        username_exist =session.query(Signup).filter(Signup.username == datas['username'],).all()
        if username_exist:  #確認有此username
            check_PWD = session.query(Signup).filter(Signup.hash_password == hash_PWD(datas['password']),).filter(Signup.username == datas['username'],).all()
            if check_PWD:  #確認有此密碼的hash_password

                update_time = {"update_time": datetime.now()}       #登入的時間
                session.query(Signup).filter_by( username = datas['username']).update(update_time)
                session.commit()
                session.close()
                return  '{"登入資訊":"'+datas['username']+'已登入"}'
            else:
               return  '{"登入資訊":"password輸入錯誤"}'
        else:
            return  '{"登入資訊":"沒有此帳號"}'
```

➢透過登入功能class實作登入API

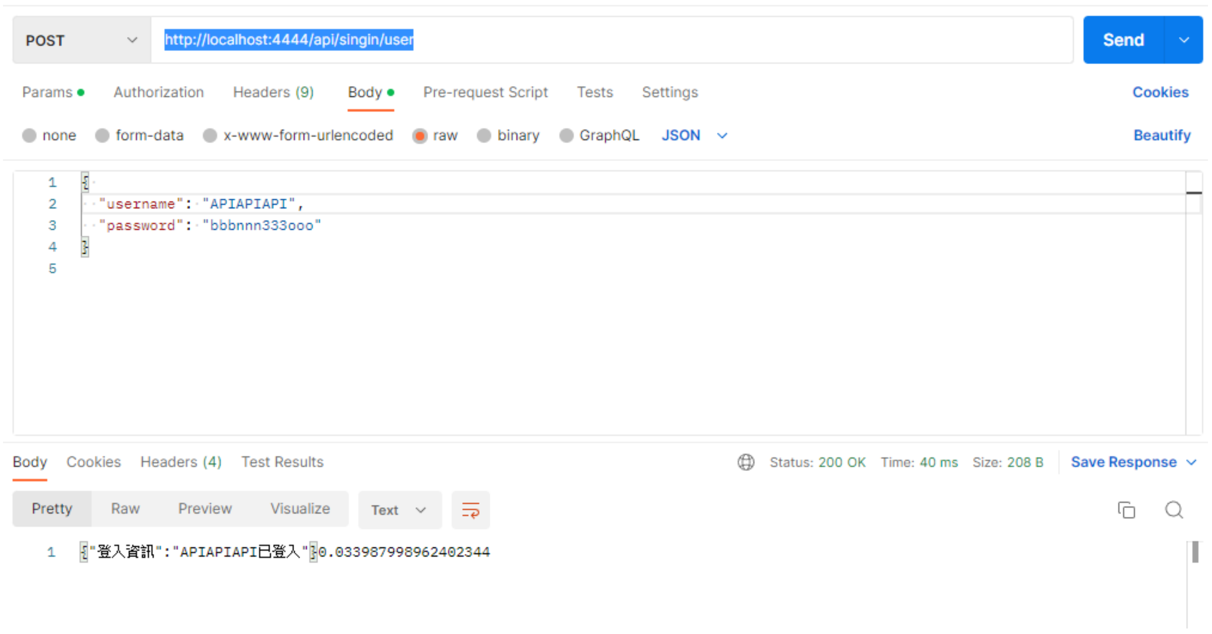
@routes.post('/api/singin/user')



```python
@routes.post('/api/singin/user')
async def singin(request):
    now = lambda: time.time()
    start = now()

    datas = await request.get_arguments_dict(['username', 'password'])  #接收username,password
    registerObj =  request.app.component_manager.get_component(HelloComponentTypes.Login)
    response = registerObj.loginProcess(**datas)                         #ok

    timeDuration = now() - start
    return  web.Response(text= response + str(timeDuration))
```

➛postman 登入API測試 http://localhost:4444/api/singin/user /POST method

第1次登入API測試,輸入正確帳密



POST http://localhost:4444/api/singin/user

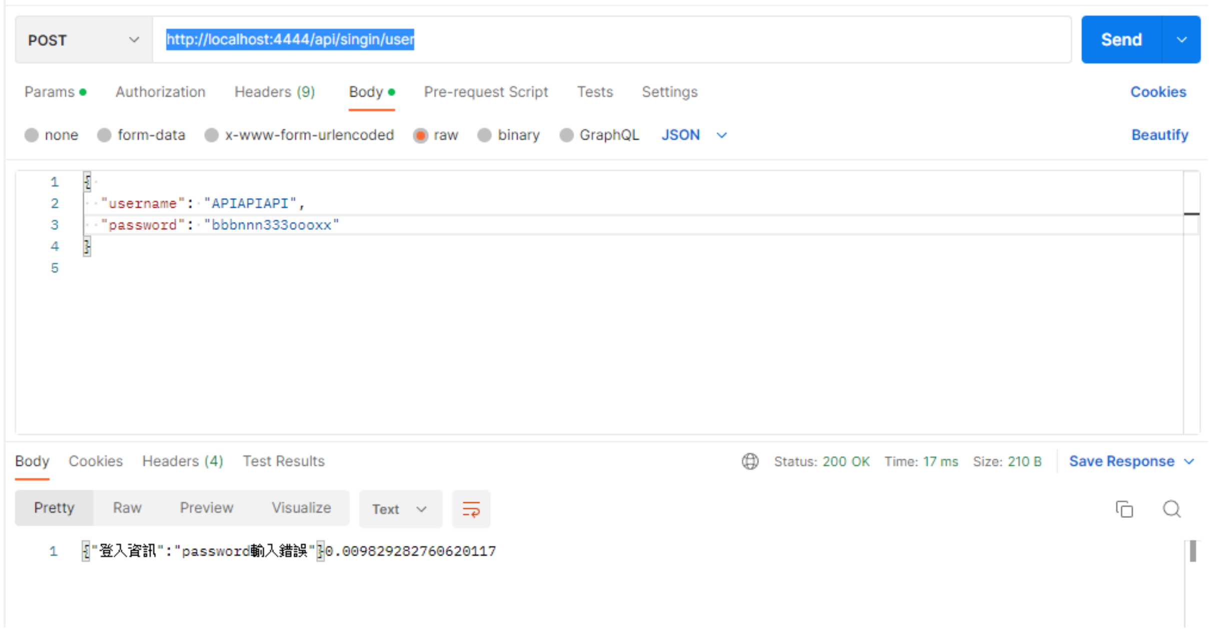
Body: raw JSON

```
{
  "username": "APIAPIAPI",
  "password": "bbbnnn333ooo"
}
```

Status: 200 OK Time: 40 ms Size: 208 B

```
{"登入資訊":"APIAPIAPI已登入"}0.033987998962402344
```

第2次登入API測試,密碼輸入錯誤



POST http://localhost:4444/api/singin/user

Body: raw JSON

```
{
  "username": "APIAPIAPI",
  "password": "bbbnnn333oooxx"
}
```

Status: 200 OK Time: 17 ms Size: 210 B

```
{"登入資訊":"password輸入錯誤"}0.009829282760620117
```

➢alembic 資料庫版本管理功能

➛製作Table_Column base.py

⤏指令app>alembic init myAlembic

⤏指令app>alembic revision -m "oooo"   #新增註解為oooo的一個版本

⤏編輯生出來的py檔裡面的 def upgrade(): , def downgrade():

#增加table 或欄位的變化操作...

⤏指令app>alembic upgrade head

則會將資料庫的schema更新成 版本號aa_oooo.py檔設定的class table&column

⤏指令app>alembic revision -m "ooooppp"   #新增註解為ooooppp的一個版本

⤏編輯生出來的py檔裡面的 def upgrade(): , def downgrade():

#增加table 或欄位的變化操作...

⤏指令app>alembic upgrade head

則會將資料庫的schema更新成 版本號bb_ooooppp.py檔設定的class table&column

⤏指令app>alembic downgrade -1 則會將資料庫的schema更新成 版本號aa_oooo.py的設定

➛如何自動產生各版本的schema py檔？ #會自動編寫def upgrade(): , def downgrade()

➛新增一個 設定Signup class table schema的signup_orm.py檔

➛編輯 alembic 專案目錄底下的 env.py 這個檔案，在最前面 import Signup,

在此目錄結構下import os,sys

```
sys.path.insert(0, os.path.dirname(os.getcwd()))
sys.path.insert(0, os.getcwd())      #路徑設定可以import上一層目錄下的.py
from component.singup_orm import Signup
```

from baseModel import User

target_metadata = Signup.metadata

指令app >alembic revision --autogenerate -m "Change column"

指令app >alembic upgrade head

 #一個--autogenerate 指令要配一個upgrade head指令將資料庫schema升級到最新版本,否則直接改BaseModel的class schema再upgrade會出現Target database is not up to date的error

ref:使用 Alembic 來進行資料庫版本管理

➢目錄結構

```
C:.
└───sipass_apiTTT
    │  alembic.ini
    │  project_config.yml
    │
    ├───component
    │  │  model_Register.py
    │  │  singup_orm.py
    │  │  __init__.py
    │  │
    │  ├───Login
    │  │  │  modeloo_login.py
    │  │  │
    │  │  └───__pycache__
    │  │          modeloo_login.cpython-38.pyc
    │  │
    │  └───__pycache__
    │
    │
    ├───controller
    │  │  model.py
    │  │  __init__.py
    │  │
    │  └───__pycache__
    │
    │
    └───myalembic
        │  env.py
        │  README
        │  script.py.mako
        │
        ├───versions
        │  │  16413da9f2f7_new_auto.py
        │  │  2272f2012ef9_new_auto.py
        │  │  506a24d3f7db_new_auto.py
        │  │  a44d3cdb2c77_signup_alembic.py
        │  │  abd461da046c_signup_alembic.py
        │  │  ba0cf2a90e3a_new_auto.py
        │  │  e5eac8144cde_new_auto.py
        │  │  fea81b6b3477_new_auto.py
        │  │
        │  └───__pycache__
        │
        └───__pycache__
              env.cpython-38.pyc
```