5. Axios를 사용하여 API를 가져오는 페이지 만들기

- API 에서 가져온 데이터를 Vue를 사용해서 HTML에 삽입하는 페이지

*templates/main/main_page.vue

```
<div id="app">
    ---- {{ '{{ message }}' }}
    <div id="rng">
        ---- {{ '{{ rng }}' }}
    </div>
</div>

<script src="https://unpkg.com/vue@3/dist/vue.global.js"></script>
<script src="https://unpkg.com/axios@1.1.2/dist/axios.min.js">      // Axios 가져오기
</script>

<script>

const app = Vue.createApp({
    data() {
        return {
            message: '안녕하세요 Vue!',
            rng: ""
        }
    },
    methods : {
        get_rng() {
            let path = "/api/rng/"

            axios.get(path).then((res) => {
                console.log(res)                    // 개발자 도구 (F12)에서 자료구조 확인
용
                console.log(res.data.rng)
                this.rng = res.data.rng;
            }).catch((error) => {
                console.error(error);
            });
        }
    },
    created() {
        this.get_rng()
    }
})

app.mount('#app')       // '#app'을 '#rng'로 적용하면 rng만 vue가 적용됨
```

```
</script>
```

```
!flask run
```

*127.0.0.1:5000
F12 눌러서 API 데이터 확인
새로고침 계속 누르면 바뀌는 거 확인

4. API 서버 만들기

- 블루프린트 작성

*views/api_view.py

```
from flask import Blueprint, jsonify
import random

bp = Blueprint('api', __name__, url_prefix='/api')

@bp.route('/rng/', methods= ["GET"])
def random_number_generator():
    output = {
        "rng" : random.randrange(1, 10)
    }
    return jsonify(output)
```

get 방식으로 통신할 거니까 methods 리스트에 "GET"을 추가한다

- 블루프린트 제출

*app.py

```
from flask import Flask
def create_app():
    app = Flask(__name__)

    from views import main_view, api_view

    app.register_blueprint(main_view.bp)
    app.register_blueprint(api_view.bp)

    return app
```

- 웹페이지에서 작동확인

```
!flask run
```

*127.0.0.1:5000/api/rng/

```
{"random":3}
```

새로고침할 때마다 바뀜

3. HTML（Template），Vue 만들기

- render_template() 함수

*views/main_views.py

```
from flask import Blueprint, render_template

bp = Blueprint('main', __name__, url_prefix='/')

@bp.route('/')
def main_page():
    return render_template("main/main_page.html")
```

- HTML 만들기

```
!mkdir templates
!mkdir templates/main
```

*templates/main/main_page.html

```
<h1>hello</h1>
```

```
flask run
```

- Vue 로 만들기 (.vue 확장자는 vue.js 가 들어갈 수 있는 html 파일이라 생각하면 됨)

```
from flask import Blueprint, render_template

bp = Blueprint('main', __name__, url_prefix='/')

@bp.route('/')
def main_page():
    return render_template("main/main_page.vue")
```

*이름 바꾸기 : main_page.html -> main_page.vue

*main_page.vue

```
<div id="app">
    ---- {{ '{{ message }}' }}
</div>

<script src="https://unpkg.com/vue@3/dist/vue.global.js"></script>
<script>

const app = Vue.createApp({
    data() {
        return {
            message: '안녕하세요 Vue!'
        }
    }
})

app.mount('#app')

</script>
```

2. 라우터 (View) 만들기

*View를 담을 폴더를 만듬

```
!mkdir views
```

- 블루프린트 만들기

*views/main_view.py

```
from flask import Blueprint

bp = Blueprint('main', __name__, url_prefix='/')

@bp.route('/')
def main_page():
    return 'Hello, Pybo!'
```

- 블루프린트 제출

*app.py

```
from flask import Flask
def create_app():
    app = Flask(__name__)

    from views import main_view

    app.register_blueprint(main_view.bp)

    return app
```

```
!flask run
```

1. Flask 시작하기

- 가상환경 설정

```
!virtualenv -p python3 {가상환경 이름}

!source {가상환경 폴더}/bin/activate
```

- 시작 코드

*console

```
!pip3 install flask
```

*app.py

```
from flask import Flask
def create_app():        # 약속된 함수
    app = Flask(__name__)

    @app.route('/')
    def main_page():
        return 'Hello, World!'

    return app
```

*console

```
!export FLASK_APP=app.py
!export FLASK_DEBUG=true        # 디버그 모드 설정
flask run
```