



It's Your Life

with





# Todo List

## REST API

### 만들기 실습

과제를 그지같이 했지만  
어쨌든 제출한 게 자랑스러울 때



## Todo

새로운 할 일을 입력하세요

추가

JPA 로 API 만들기

완료

수정

삭제

Vue 랑 통신 해보기

완료

수정

삭제

~~열공 하카~~ ✓

미완료

수정

삭제

Vue 코드는 여기서 받으시면 됩니다!

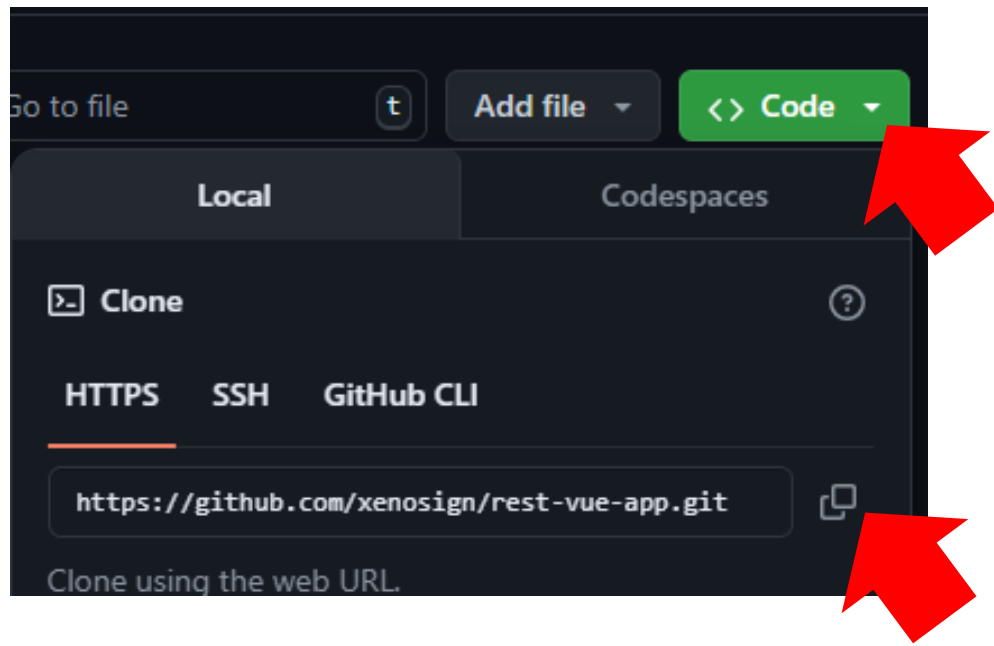


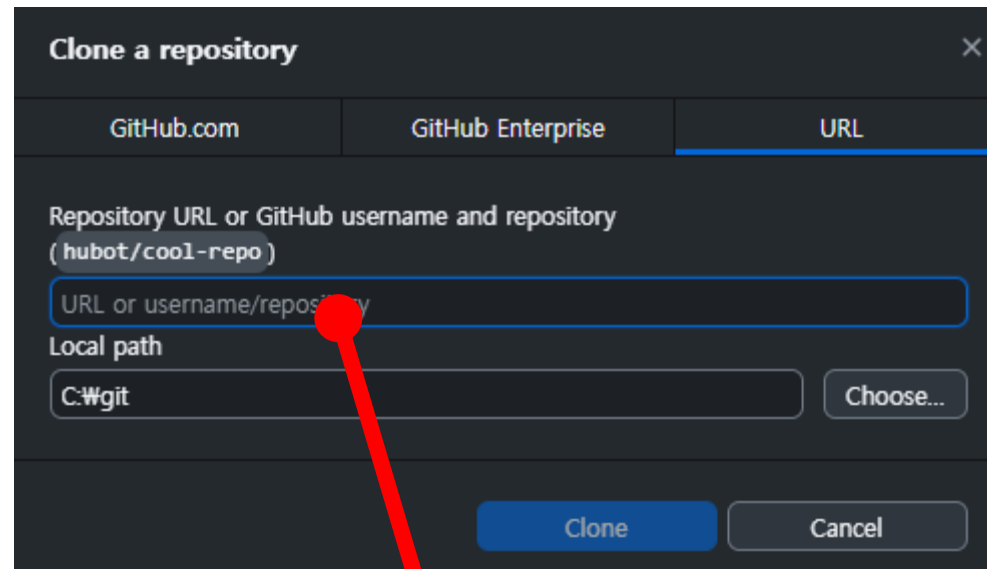
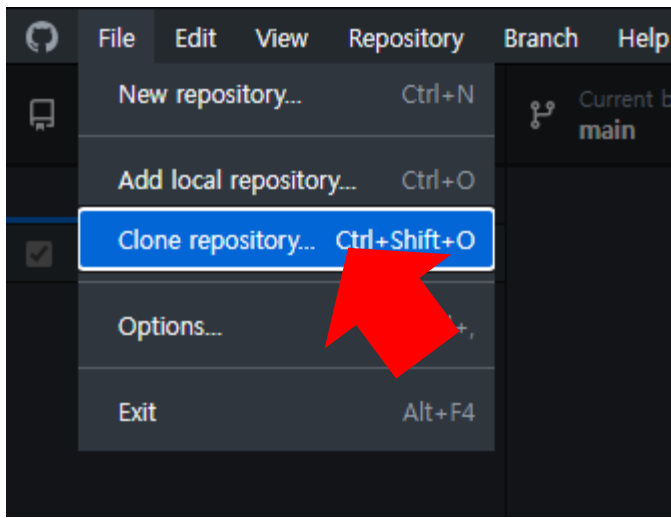
<https://github.com/xenosign/rest-vue-app>



# Vue 코드 클론

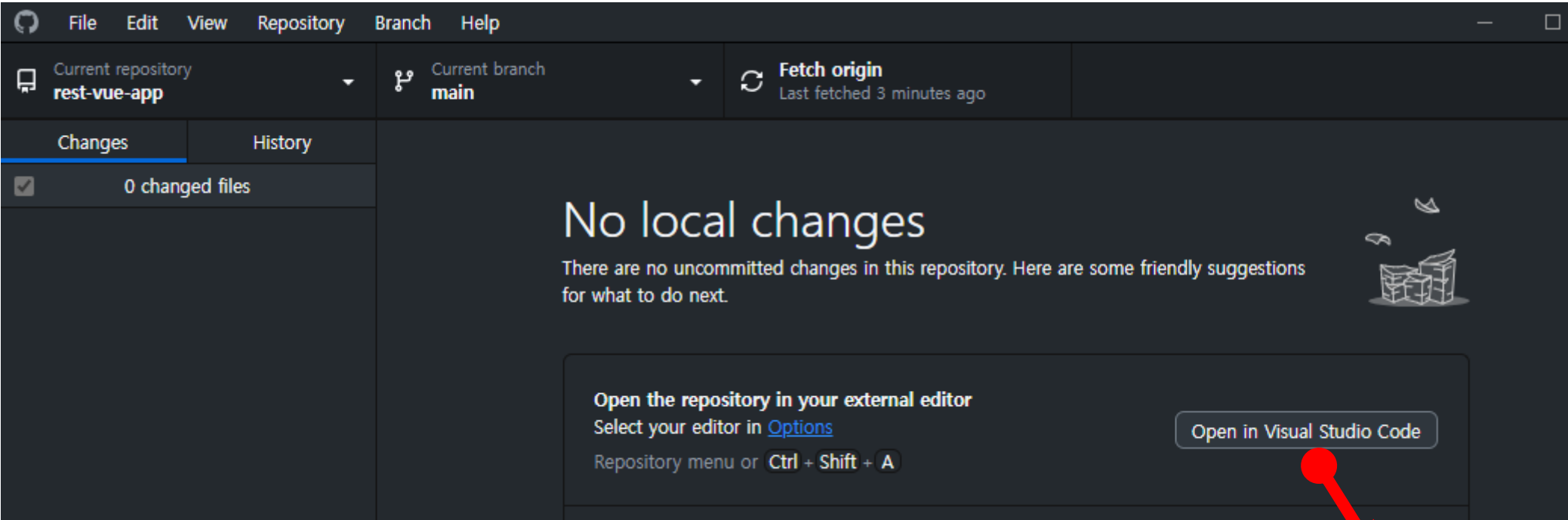
## & 실행





복사한 주소를 복붙 → Clone





VSCoDe 로 열어 주세요





터미널로 열어 주세요!

파일(F) 편집(E) 선택 영역(S) 보기(V) 이동(G) 실행(R) 터미널(T) 도움말(H)

탐색기

- REST-VUE-APP
  - .vscode
  - node\_modules
  - public
  - src
    - components
      - OnClickBoard.vue
      - OnMountedBoard.vue
      - App.vue
      - main.js
  - .gitattributes
  - .gitignore
  - index.html
  - jsconfig.json
  - package-lock.json
  - package.json
  - README.md
  - vite.config.js

src > JS main.js

```
1 import { createApp } from 'vue';
2 import App from './App.vue';
3
4 createApp(App).mount('#app');
5
```

문제 출력 디버그 콘솔 터미널 포트

```
student@M503 MINGW64 /c/git/rest-vue-app (main)
$
```



```
student@M503 MINGW64 /c/git/rest-vue-app (main)  
$ npm i
```

npm i 로 의존성이 있는 라이브러리를 설치

→ 이 때 Vue 프레임 워크와 Axios 등이  
설치 됩니다!

```
student@M503 MINGW64 /c/git/rest-vue-app (main)  
$ npm run dev
```

설치가 완료 되면  
npm run dev 로 Vue App 구동!



```
student@M503 MINGW64 /c/git/rest-vue-app (main)
$ npm run dev

> rest-vue-app@0.0.0 dev
> vite

VITE v5.4.0 ready in 282 ms

→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```

## Todo

추가

Todo 리스트 받기 실패



# DB 만들기!

<https://github.com/xenosign/spring-code-repo/blob/main/sql/todo.sql>



```
USE mybatis;
```

```
DROP TABLE IF EXISTS todo;
```

```
CREATE TABLE todo (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    todo VARCHAR(50) NOT NULL,  
    done BOOLEAN  
);
```

```
INSERT INTO todo (todo, done)  
VALUES ("JPA 로 API 만들기", false),  
("Vue 랑 통신 해보기", false),  
("열공 하기", true);
```

```
select * from todo;
```

Result Grid     Filter Rows: <input type="text"/>			
	id	todo	done
▶	1	JPA 로 API 만들기	0
	2	Vue 랑 통신 해보기	0
	3	열공 하기	1
✱	NULL	NULL	NULL





# Todo List

간단 설명



# Todo

새로운 할 일을 입력하세요

추가

JPA 로 API 만들기

완료

수정

삭제

Vue 랑 통신 해보기

완료

수정

삭제

~~열공~~ 하가 ✓

미완료

수정

삭제



# Todo List

# REST API 만들기





다음의 문서를 보시고 스펙에 맞는 백엔드 서버를 만들어 주시면 됩니다!


단, CORS 이슈가 발생하므로 TodoController 는 CORS 를 "\*" 로 설정 필수!

마이바티스 또는 JPA 를 사용해서 구현하시면 됩니다!

둘 다 하면 BEST!!!!!!

# Todo List 받아오기

- 메서드 : GET 방식
- 요청 uri
  - <http://localhost:8080/todo>
- 응답 형식 : RESTful
- 데이터 : Todo 가 들어있는 배열
- 데이터 형식 : JSON



```
GET http://localhost:8080/todo

Params
Authorization
Headers (7)
Body
Scripts
Test

Query Params

Key
Key

Body
Cookies (1)
Headers (8)
Test Results

Pretty
Raw
Preview
Visualize
JSON
[
  {
    "id": 1,
    "todo": "JPA 로 API 만들기",
    "done": false
  },
  {
    "id": 2,
    "todo": "Vue 란 통신 해보기",
    "done": false
  },
  {
    "id": 3,
    "todo": "열공 하기",
    "done": true
  }
]
```

# id 로 Todo 찾기



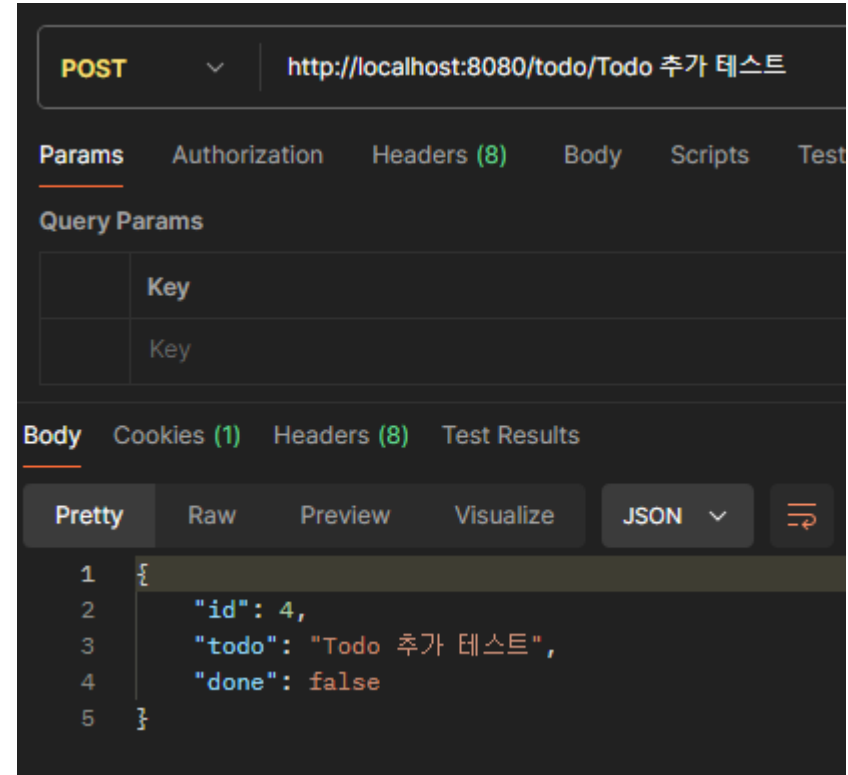
- 메서드 : GET 방식
- 요청 uri
  - <http://localhost:8080/todo/:id>
- 응답 형식 : RESTful
- 데이터 : id 값으로 찾은 Todo 객체
- 데이터 형식 : JSON
- `/:id` 는 파라미터를 PathVariable 형태로 데이터를 전달 합니다

A screenshot of a REST client interface. At the top, a dropdown menu shows 'GET' and the URL 'http://localhost:8080/todo/1'. Below this, there are tabs for 'Params', 'Authorization', 'Headers (7)', 'Body', and 'Scripts'. The 'Params' tab is selected, showing a table with two rows, each with a 'Key' column. Below the 'Params' tab, there are tabs for 'Body', 'Cookies (1)', 'Headers (8)', and 'Test Results'. The 'Body' tab is selected, showing a JSON response in 'Pretty' format. The JSON response is: 

```
{
  "id": 1,
  "todo": "JPA 로 API 만들기",
  "done": false
}
```

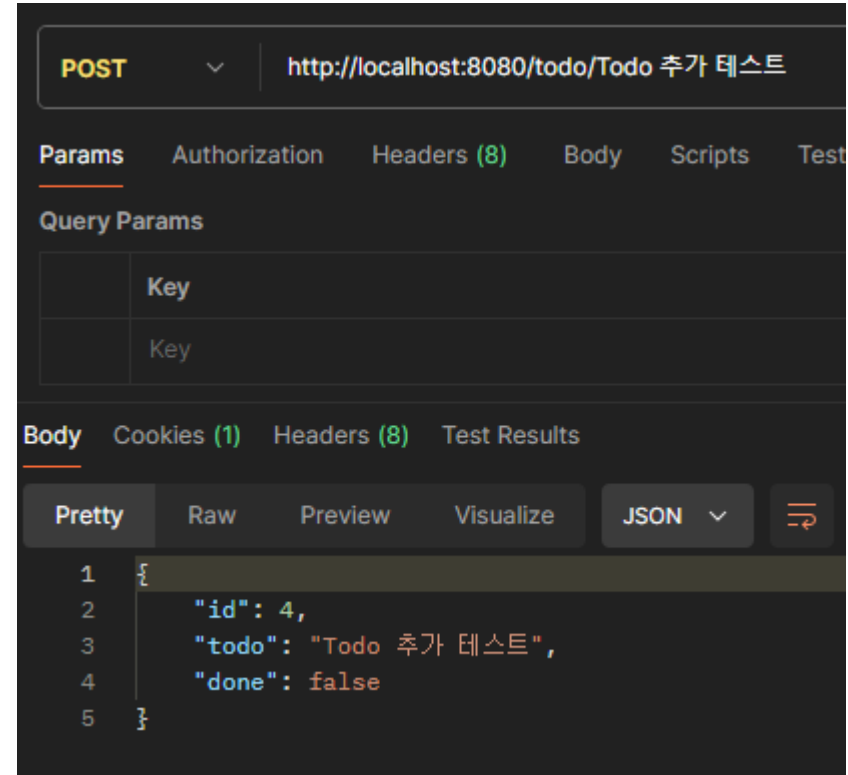
# Todo 추가하기

- 메서드 : POST 방식
- 요청 uri
  - <http://localhost:8080/todo/:todo>
- 응답 형식 : RESTful
- 데이터 : 추가 된 Todo 객체
- 데이터 형태 : JSON



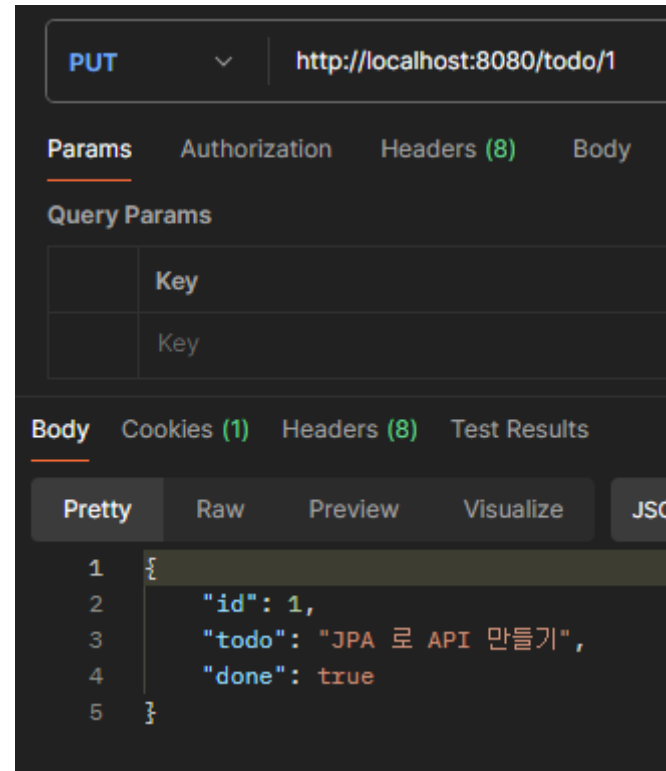
# Todo 추가하기

- 메서드 : POST 방식
- 요청 uri
  - <http://localhost:8080/todo/:todo>
- 응답 형식 : RESTful
- 데이터 : 추가 된 Todo 객체
- 데이터 형태 : JSON



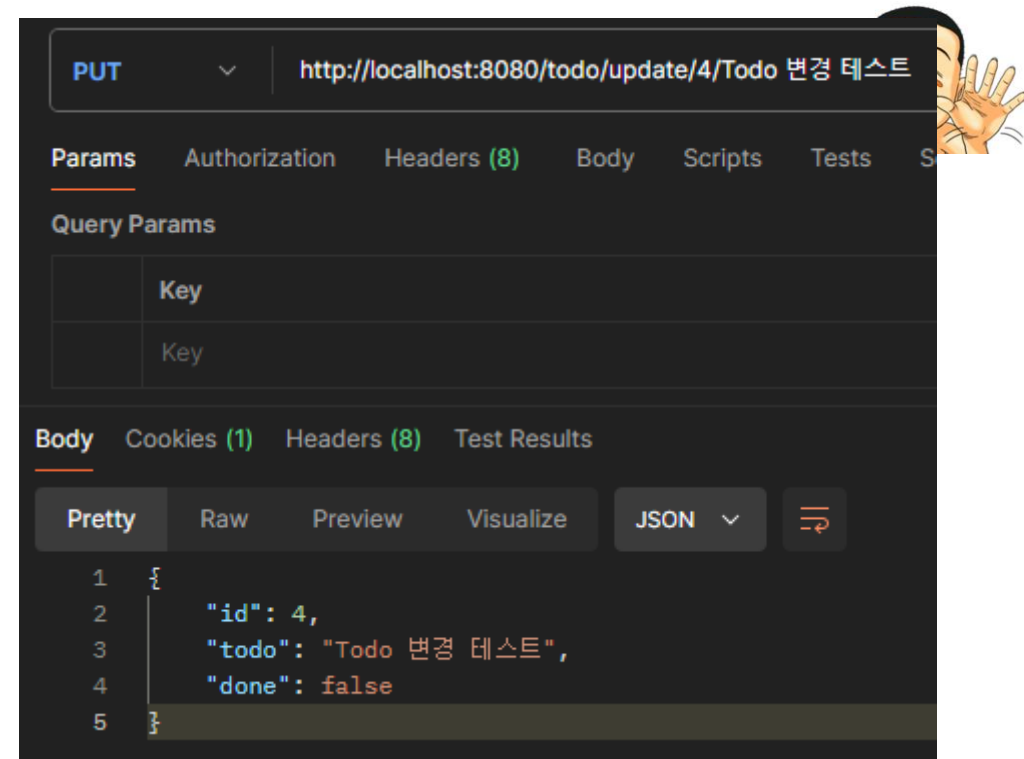
# Todo 완료 여부 변경

- 메서드 : PUT 방식
- 요청 uri
  - <http://localhost:8080/todo/:id>
- 응답 형식 : RESTful
- 데이터 : 완료 여부가 변경 된 Todo 객체
- 데이터 형태 : JSON



# Todo 내용 변경

- 메서드 : PUT 방식
- 요청 uri
  - <http://localhost:8080/todo/update/:id/:todo>
- 응답 형식 : RESTful
- 데이터 : 내용이 변경 된 Todo 객체
- 데이터 형태 : JSON



# Todo 삭제

- 메서드 : DELETE 방식
- 요청 uri
  - <http://localhost:8080/todo/:id>
- 응답 형식 : RESTful
- 데이터 : 삭제 된 Todo 객체
- 데이터 형태 : JSON

