KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

vue-router와 axios를 사용한 예제

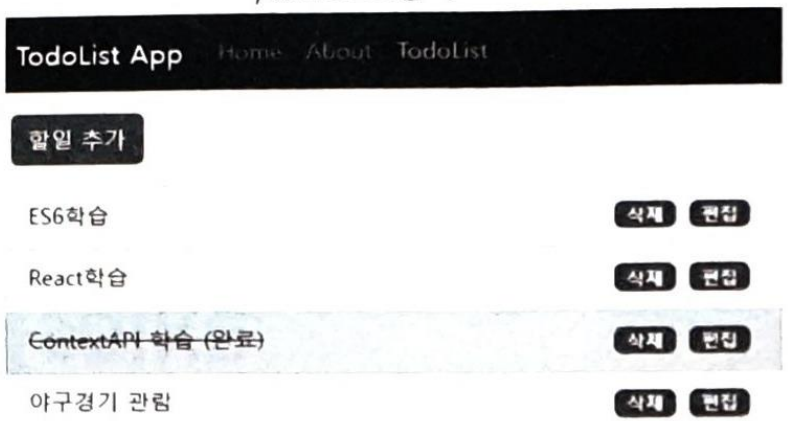
[KB] IT's Your Life

KB 국민은행

작성할 화면

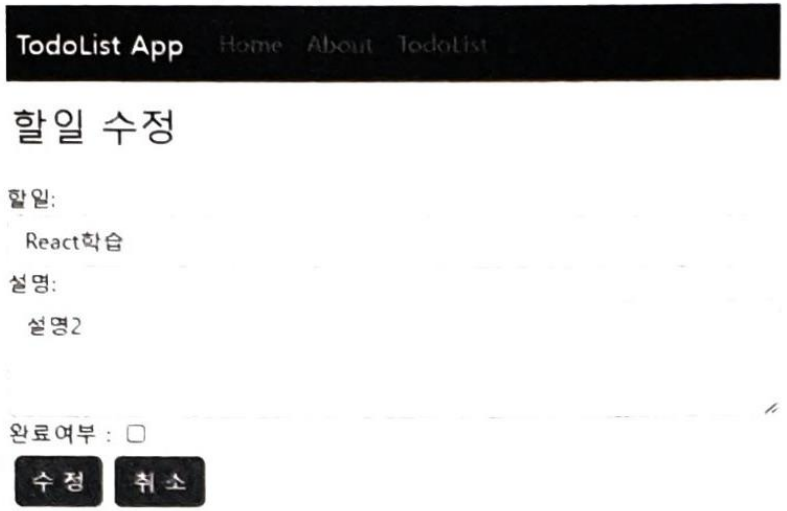
/home, /about 요청시

/todos 요청시

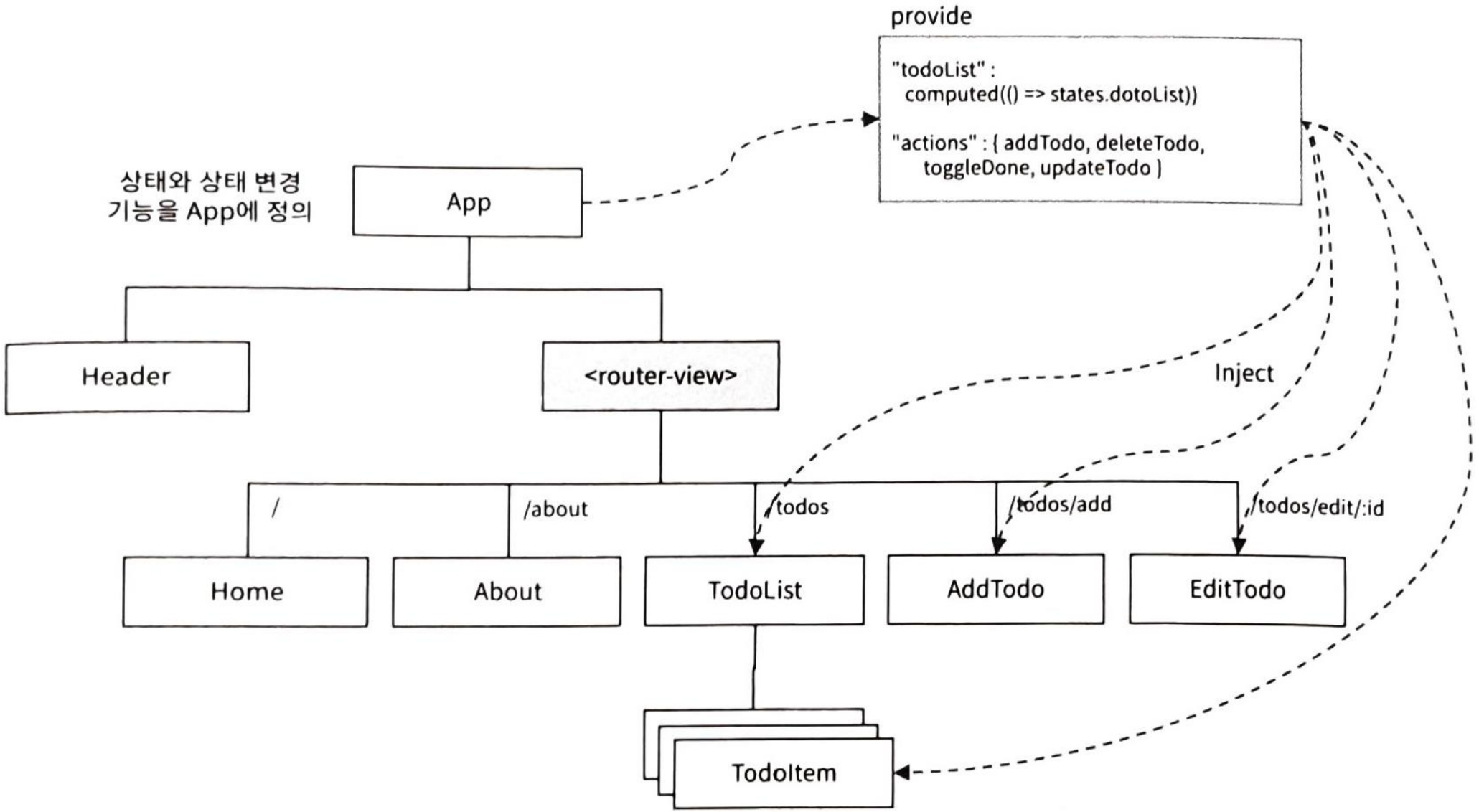


/todos/add 요청시

/todos/edit/:id 요청시



컴포넌트 계층 구조


provide
"todoList" :
computed(() => states.dotoList))
"actions" : { addTodo, deleteTodo,
toggleDone, updateTodo }
상태와 상태 변경
기능을 App에 정의
App
Header
<router-view>
Inject
/
/about
/todos
/todos/add
/todos/edit/:id
Home
About
TodoList
AddTodo
EditTodo
TodoItem

상태 데이터

[상태 데이터]

```
{
  todoList : [
    { id: 1, todo: "ES6학습", desc: "설명1", done: false },
    ......
  ]
}
```

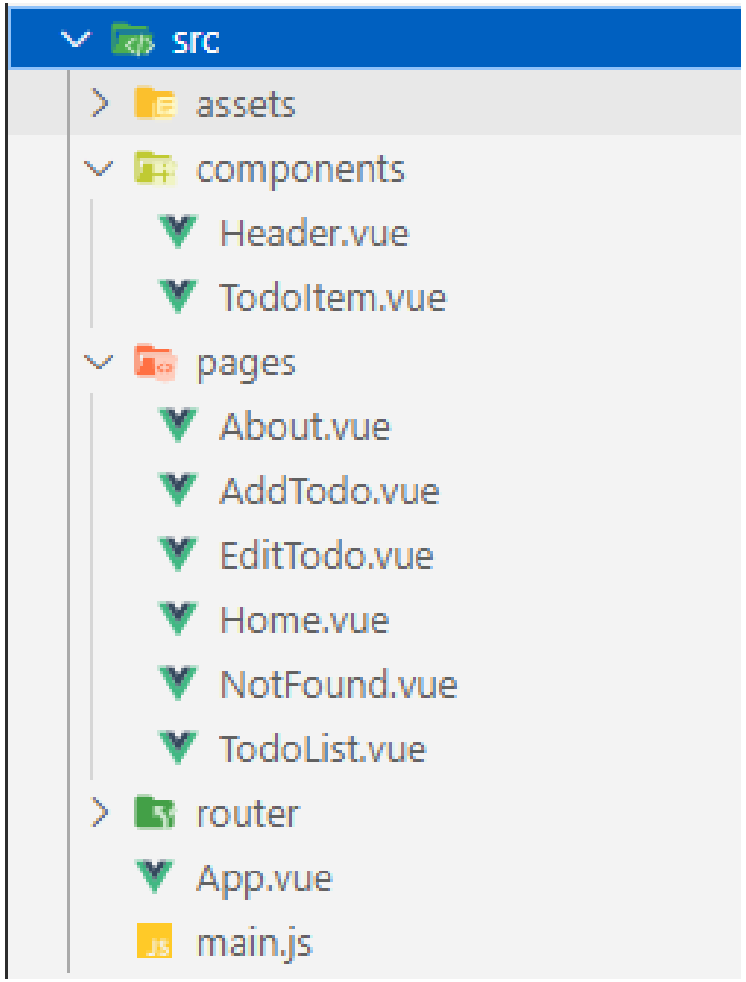
[상태 변경 기능]

```
addTodo : ({ todo, desc }) => { }
updateTodo : ({ id, todo, desc, done }) => { }
deleteTodo : (id) => { }
toggleDone : (id) => { }
```

프로젝트 생성

npm init vue todolist-app-router
→ 라우트 기능 설정
cd todolist-app-router
npm install
npm install bootstrap@5 axios

디렉토리및 파일 구조


src
assets
components
Header.vue
TodoItem.vue
pages
About.vue
AddTodo.vue
EditTodo.vue
Home.vue
NotFound.vue
TodoList.vue
router
App.vue
main.js

src/pages/Home.vue

```
<template>
    <div class="card card-body">
        <h2>Home</h2>
    </div>
</template>
```

○ 나머지 페이지 컴포넌트도 같은 방식으로 작성

src/pages/NotFound.vue

```
<template>
    <div class="m-3">
        <h3>존재하지 않는 경로</h3>
        <p>요청 경로 : {{currentRoute.path}}</p>
    </div>
</template>

<script setup>
import { useRoute } from 'vue-router'
const currentRoute = useRoute();
</script>
```

src/main.js

```
import './assets/main.css'

import { createApp } from 'vue'
import App from './App.vue'
import router from './router'
import 'bootstrap/dist/css/bootstrap.css'

const app = createApp(App)

app.use(router)

app.mount('#app')
```

src/assets/main.css

```
body {  margin: 0; padding: 0; font-family: sans-serif;  }
.title {  text-align: center; font-weight: bold; font-size: 20pt;  }
.todo-done {  text-decoration: line-through;  }
.container {  padding: 10px 10px 10px 10px;  }
.panel-borderless {  border: 0; box-shadow: none;  }
.pointer {  cursor: pointer;  }
```

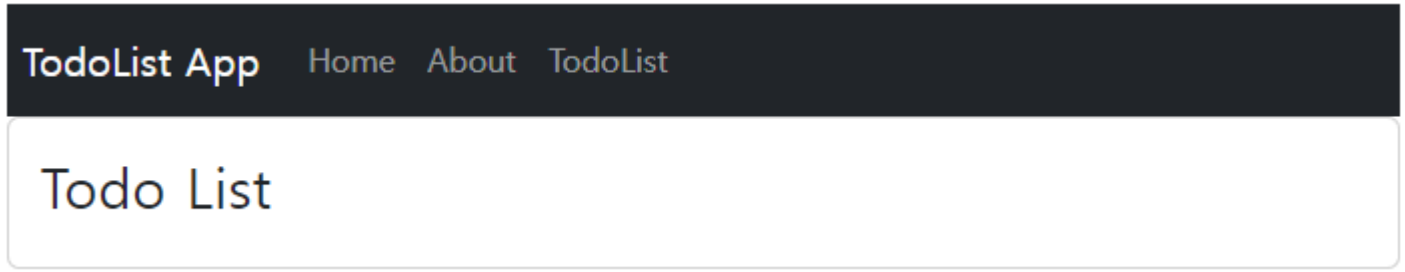
src/components/Header.vue

```
<template>
    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">
      <span class="navbar-brand ps-2">TodoList App</span>
      <button class="navbar-toggler" type="button" @click="isNavShow = !isNavShow">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div :class="isNavShow ? 'collapse navbar-collapse show' : 'collapse navbar-collapse'">
        <ul class="navbar-nav">
          <li class="nav-item">
            <router-link class="nav-link" to="/">Home</router-link>
          </li>
          <li class="nav-item">
            <router-link class="nav-link" to="/about">About</router-link>
          </li>
          <li class="nav-item">
            <router-link class="nav-link" to="/todos">TodoList</router-link>
          </li>
        </ul>
      </div>
    </nav>
</template>
```

src/components/Header.vue

```
<script setup>
import { ref } from 'vue';

const isNavShow = ref(false);
</script>
```



src/App.vue

```
<template>
    <div class="container">
        <Header />
        <router-view />
    </div>
</template>

<script setup>
import { reactive, computed, provide } from 'vue'
import Header from '@/components/Header.vue'

const states = reactive({
  todoList : [
    { id: 1, todo: "ES6학습", desc: "설명1", done: false },
    { id: 2, todo: "React학습", desc: "설명2", done: false },
    { id: 3, todo: "ContextAPI 학습", desc: "설명3", done: true },
    { id: 4, todo: "야구경기 관람", desc: "설명4", done: false },
  ]
})
```

src/App.vue

```
const addTodo = ({ todo, desc }) => {
  states.todoList.push({ id: new Date().getTime(), todo, desc, done: false })
};

const updateTodo = ({ id, todo, desc, done }) => {
  let index = states.todoList.findIndex((todo) => todo.id === id);
  states.todoList[index] = { ...states.todoList[index], todo, desc, done };
};

const deleteTodo = (id) => {
  let index = states.todoList.findIndex((todo) => todo.id === id);
  states.todoList.splice(index, 1);
}

const toggleDone = (id) => {
  let index = states.todoList.findIndex((todo) => todo.id === id);
  states.todoList[index].done = !states.todoList[index].done;
}

provide('todoList', computed(()=>states.todoList))
provide('actions', { addTodo, deleteTodo, toggleDone, updateTodo })
</script>
```

src/components/TodoItem.vue

```
<template>
    <li :class="todoItem.done ? 'list-group-item list-group-item-success' : 'list-group-item'">
      <span :class="todoItem.done ? 'todo-done pointer' : 'pointer'"
        @click="toggleDone(todoItem.id)">
        {{todoItem.todo}}
        {{todoItem.done ? '(완료)' : '' }}
      </span>
      <span class="float-end badge bg-secondary pointer m-1"
        @click="router.push(`/todos/edit/${todoItem.id}`)">
        편집</span>
      <span class="float-end badge bg-secondary pointer m-1"
        @click="deleteTodo(todoItem.id)">
        삭제</span>
    </li>
</template>
```

src/components/TodoItem.vue

```
<script setup>
import { useRouter } from 'vue-router';
import { inject } from 'vue';

defineProps({
  todoItem: { Type: Object, required:true }
})

const router = useRouter();
const { deleteTodo, toggleDone } = inject('actions');
</script>
```

src/pages/TodoList.vue

```
<template>
    <div class="row">
        <div class="col p-3">
            <router-link class="btn btn-primary" to="/todos/add">
            할일 추가
            </router-link>
        </div>
    </div>
    <div class="row">
        <div class="col">
            <ul class="list-group">
                <TodoItem v-for="todoItem in todoList" :key="todoItem.id" :todoItem="todoItem" />
            </ul>
        </div>
    </div>
</template>

<script setup>
import {inject} from 'vue';
import TodoItem from '@/components/TodoItem.vue'

const todoList = inject('todoList');
</script>
```

src/pages/AddTodo.vue

```
<template>
    <div class="row">
        <div class="col p-3">
            <h2>할일 추가</h2>
        </div>
    </div>
    <div class="row">
        <div class="col">
            <div class="form-group">
                <label htmlFor="todo">할일 :</label>
                <input type="text" class="form-control" id="todo" v-model="todoItem.todo" />
            </div>
            <div class="form-group">
                <label htmlFor="desc">설명 :</label>
                <textarea class="form-control" rows="3" id="desc" v-model="todoItem.desc"></textarea>
            </div>
            <div class="form-group">
                <button type="button" class="btn btn-primary m-1" @click="addTodoHandler"> 추 가
            </button>
                <button type="button" class="btn btn-primary m-1" @click="router.push('/todos')"> 취 소
            </button>
            </div>
        </div>
    </div>
</template>
```

src/pages/AddTodo.vue

```
<script setup>
import { inject, reactive } from 'vue';
import { useRouter } from 'vue-router';

const router = useRouter();
const { addTodo } = inject('actions');
const todoItem =  reactive({ todo:"", desc:"" })

const addTodoHandler = () => {
    let { todo } = todoItem;
    if (!todo || todo.trim()==="") {
        alert('할일은 반드시 입력해야 합니다');
        return;
    }
    addTodo({ ...todoItem });
    router.push('/todos')
}
</script>
```

src/pages/EditTodo.vue

```
<template>
    <div class="row">
        <div class="col p-3">
            <h2>할일 수정</h2>
        </div>
    </div>
    <div class="row">
        <div class="col">
            <div class="form-group">
                <label htmlFor="todo">할일:</label>
                <input type="text" class="form-control" id="todo" v-model="todoItem.todo" />
            </div>
            <div class="form-group">
                <label htmlFor="desc">설명:</label>
                <textarea class="form-control" rows="3" id="desc" v-model="todoItem.desc"></textarea>
            </div>
            <div class="form-group">
                <label htmlFor="done">완료여부 : </label> 
                <input type="checkbox" v-model="todoItem.done" />
            </div>
```

src/pages/EditTodo.vue

```
            <div class="form-group">
                <button type="button" class="btn btn-primary m-1" @click="updateTodoHandler">
                  수 정
                </button>
                <button type="button" class="btn btn-primary m-1" @click="router.push('/todos')">
                  취 소
                </button>
            </div>
        </div>
    </div>
</template>
```

src/pages/EditTodo.vue

```
<script setup>
import { inject, reactive } from 'vue';
import { useRouter, useRoute } from 'vue-router';

const todoList = inject('todoList');
const { updateTodo } = inject('actions');
const router = useRouter();
const currentRoute = useRoute();

const matchedTodoItem = todoList.value.find((item)=> item.id === parseInt(currentRoute.params.id))
if (!matchedTodoItem)  {
    router.push('/todos');
}
const todoItem =  reactive({ ...matchedTodoItem })

const updateTodoHandler = () => {
    let { todo } = todoItem;
    if (!todo || todo.trim()==="") {
        alert('할일은 반드시 입력해야 합니다');
        return;
    }
    updateTodo({ ...todoItem });
    router.push('/todos');
}
</script>
```

백엔드 API 실행과 프록시 설정

axios 설치

- npm install axios

vite.config.js 변경

```
…
  server: {
    proxy: {
      '/api': {
        target: 'http://localhost:3000',
        changeOrigin: true,
        rewrite: (path) => path.replace(/^\/api/, ''),
      },
    },
  },
})
```

db.json

```
{
  "todos": [
    { "id": "1", "todo": "ES6학습", "desc": "설명1", "done": false },
    { "id": "2", "todo": "React학습", "desc": "설명2", "done": false },
    { "id": "3", "todo": "ContextAPI 학습", "desc": "설명3", "done": true },
    { "id": "4", "todo": "야구경기 관람", "desc": "설명4", "done": false }
  ]
}
```

데이터 서버 기동

- 새 터미널에서

  npx json-server db.json

src/App.vue

```
<template>
  <div class="container">
      <Header />
      <router-view />
  </div>
</template>

<script setup>
import { reactive, provide, computed } from 'vue'
import Header from '@/components/Header.vue'
import axios from 'axios';

const BASEURI = "/api/todos";
const states = reactive({ todoList:[]  })
```

src/App.vue

```
//TodoList 목록을 조회합니다.
const fetchTodoList = async () => {
  try {
    const response = await axios.get(BASEURI);
    if (response.status === 200) {
        states.todoList = response.data;
    } else {
        alert('데이터 조회 실패');
    }
  } catch(error) {
    alert('에러발생 :' + error);
  }
}
```

src/App.vue

```
// 새로운 TodoItem을 추가합니다.
const addTodo = async ({ todo, desc }, successCallback) => {
  try {
    const payload = { todo, desc };
    const response = await axios.post(BASEURI, payload);
    if (response.status === 201) {
      states.todoList.push({ ...response.data, done: false });
      successCallback();
    } else {
      alert('Todo 추가 실패');
    }
  } catch (error) {
    alert('에러발생 :' + error);
  }
};
```

src/App.vue

```
// 기존 TodoItem을 변경합니다.
const updateTodo = async ({ id, todo, desc, done }, successCallback) => {
  try {
    const payload = { id, todo, desc, done };
    const response = await axios.put(BASEURI + `/${id}`, payload);
    if (response.status === 200) {
      let index = states.todoList.findIndex((todo) => todo.id === id);
      states.todoList[index] = payload;
      successCallback();
    } else {
      alert('Todo 변경 실패');
    }
  } catch (error) {
    alert('에러발생 :' + error);
  }
};
```

src/App.vue

```
//기존 TodoItem을 삭제합니다.
const deleteTodo = async (id) => {
  try {
    const response = await axios.delete(BASEURI + `/${id}`);
    console.log(response.status, response.data);
    if (response.status === 200) {
      let index = states.todoList.findIndex((todo) => todo.id === id);
      states.todoList.splice(index, 1);
    } else {
      alert('Todo 삭제 실패');
    }
  } catch (error) {
    alert('에러발생 :' + error);
  }
};
```

src/App.vue

```
//기존 TodoItem의 완료여부(done) 값을 토글합니다.
const toggleDone = async (id) => {
  try {
    let todo = states.todoList.find((todo) => todo.id === id);
    let payload = { ...todo, done: !todo.done };
    const response = await axios.put(BASEURI + `/${id}`, payload);
    if (response.status === 200) {
      todo.done = payload.done;
    } else {
      alert('Todo 완료 변경 실패');
    }
  } catch (error) {
    alert('에러발생 :' + error);
  }
};

provide('todoList', computed(()=>states.todoList));
provide('actions', { addTodo, deleteTodo, toggleDone, updateTodo, fetchTodoList })

fetchTodoList();
</script>
```

src/pages/AddTodo.vue 변경

```
…

<script setup>
import { inject, reactive } from 'vue';
import { useRouter } from 'vue-router';

const router = useRouter();
const { addTodo } = inject('actions');
const todoItem =  reactive({ todo:"", desc:"" })

const addTodoHandler = () => {
    let { todo } = todoItem;
    if (!todo || todo.trim()==="") {
        alert('할일은 반드시 입력해야 합니다');
        return;
    }
    addTodo({ ...todoItem }, ()=>{
        router.push('/todos')
    });
}
</script>
```

src/pages/EditTodo.vue 변경

```
…
<script setup>
import { inject, reactive } from 'vue';
import { useRouter, useRoute } from 'vue-router';

const todoList = inject('todoList');
const { updateTodo } = inject('actions');
const router = useRouter();
const currentRoute = useRoute();

const matchedTodoItem = todoList.value.find((item)=> item.id === currentRoute.params.id)
if (!matchedTodoItem)  {
    router.push('/todos');
}
const todoItem =  reactive({ ...matchedTodoItem })
```

src/pages/EditTodo.vue 변경

```
const updateTodoHandler = () => {
    let { todo } = todoItem;
    if (!todo || todo.trim()==="") {
        alert('할일은 반드시 입력해야 합니다');
        return;
    }
    updateTodo({ ...todoItem }, ()=>{
        router.push('/todos');
    });
}
</script>
```

src/pages/TodoList.vue 변경

```
<template>
    <div class="row">
        <div class="col p-3">
            <router-link class="btn btn-primary" to="/todos/add">
            할일 추가
            </router-link>
            <button class="btn btn-primary ms-1" @click="fetchTodoList">
            새로 고침
            </button>
        </div>
    </div>
    <div class="row">
        <div class="col p-3">
            <ul class="list-group">
                <TodoItem v-for="todoItem in todoList" :key="todoItem.id" :todoItem="todoItem" />
            </ul>
        </div>
    </div>
</template>
```

src/pages/TodoList.vue 변경

```
<script setup>
import {inject} from 'vue';
import TodoItem from '@/components/TodoItem.vue'

const todoList = inject('todoList');
const { fetchTodoList } = inject('actions');
</script>
```

로딩 아이콘

- npm install vue-csspin

src/components/Loading.vue

```
<template>
    <VueCsspin message="Loading" spin-style="cp-flip" />
</template>

<script setup>
import { VueCsspin } from 'vue-csspin'
import 'vue-csspin/dist/vue-csspin.css'
</script>
```

src/App.vue 변경

```
<template>
  <div class="container">
      <Header />
      <router-view />
      <Loading v-if="states.isLoading" />
  </div>
</template>

<script setup>
import { reactive, provide, computed } from 'vue'
import Header from '@/components/Header.vue'
import Loading from '@/components/Loading.vue'
import axios from 'axios';
…
```

src/App.vue 변경

```
//TodoList 목록을 조회합니다.
const fetchTodoList = async () => {
  states.isLoading = true;
  …
  states.isLoading = false;
}

// 새로운 TodoItem을 추가합니다.
const addTodo = async ({ todo, desc }, successCallback) => {
  states.isLoading = true;
  …
  states.isLoading = false;
}

// 기존 TodoItem을 변경합니다.
const updateTodo = async ({ id, todo, desc, done }, successCallback) => {
  states.isLoading = true;
  …
  states.isLoading = false;
}
```

src/App.vue 변경

```
//기존 TodoItem을 삭제합니다.
const deleteTodo = async (id) => {
  states.isLoading = true;
  …
  states.isLoading = false;
}

//기존 TodoItem의 완료여부(done) 값을 토글합니다.
const toggleDone = async (id) => {
  states.isLoading = true;
  …
  states.isLoading = false;
}

provide('todoList', computed(()=>states.todoList));
provide('actions', { addTodo, deleteTodo, toggleDone, updateTodo, fetchTodoList })

fetchTodoList();
</script>
```