Redis, FastAPI and Vue.js in a WebApp

We'll create a Web Application with a FastAPI backend working with a Redis database and a Vue.js frontend. The application will be hosted on render.com

---

Redis Database

- Log in to *https://dashboard.render.com/*
- Select `+ New` > `Redis`
- Name: **human-record**
- Region: **Singapore**
- Maxmemory Policy: **noeviction**
- Instance Type: **Free**
- Click `Create Redis`

Note the value of "Internal Redis URL" *redis://red-cri65j3qf0us739pke10:6379*

FastAPI Backend Code

- Create a project directory on the local computer `~/projects/python/webapp-01/fastapi-backend`
- Save the following code with the filename `fastapi-backend.py`

```
from fastapi import FastAPI
import redis

app = FastAPI()

r = redis.Redis(host='redis://red-cri65j3qf0us739pke10', port=6379, decode_responses=True)

@app.get("/")
def read_root():
    return {"message": "Server says Hello!"}
```

```
@app.post("/add")
def add_record(id: int, name: str, age: int, weight: float):
    r.hset(f"user:{id}", mapping={"name": name, "age": age, "weight": weight})
    return {"message": "Record added"}

@app.get("/get/{id}")
def get_record(id: int):
    record = r.hgetall(f"user:{id}")
    return record

@app.put("/update/{id}")
def update_record(id: int, name: str = None, age: int = None, weight: float = None):
    if name:
        r.hset(f"user:{id}", "name", name)
    if age:
        r.hset(f"user:{id}", "age", age)
    if weight:
        r.hset(f"user:{id}", "weight", weight)
    return {"message": "Record updated"}

@app.delete("/delete/{id}")
def delete_record(id: int):
    r.delete(f"user:{id}")
    return {"message": "Record deleted"}
```

- Save the following content with a filename `requirements.txt`

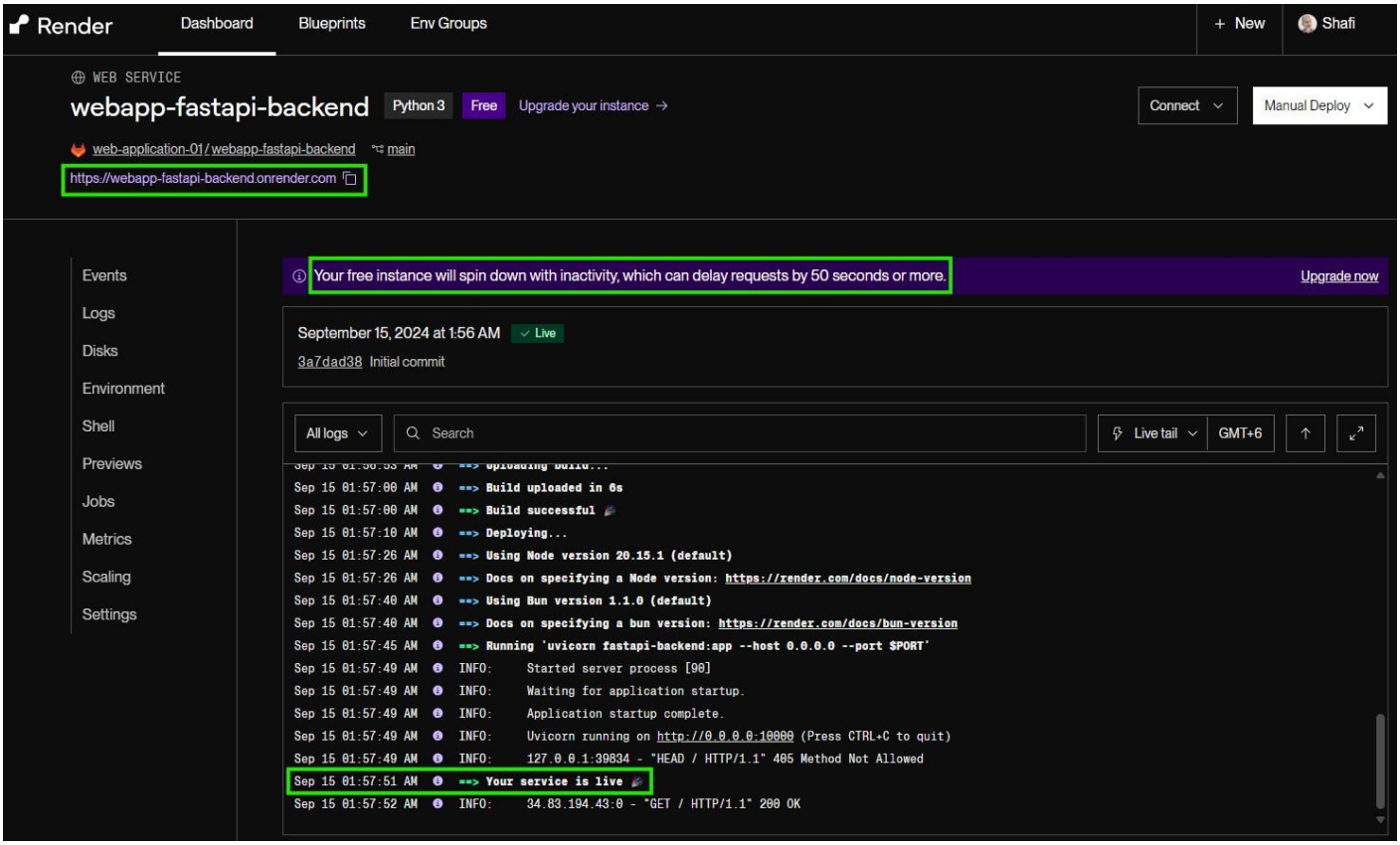
```
fastapi
uvicorn[standard]
redis
```

- Create a public GitLab project *https://gitlab.com/web-application-01/webapp-fastapt-backend*
- Run the following commands from the directory where the fastapi project file was created

```
cd ~/projects/python/webapp-01/fastapi-backend
git init
git add .
git commit -m "Initial commit"
git remote add origin git@gitlab.com:web-application-01/webapp-fastapi-backend.git
git push -u origin main
```

FastAPI Backend Service

- Go back to *https://dashboard.render.com/*
- Select `+ New` > `Web Service`
- Select `GitLab` from `Source Code` > `Git Provider` section
- Click `Authorize Render for GitLab`
- Select the `web-application-01/webapp-fastapt-backend` project and click `Connect`
- Name: **webapp-fastapi-backend**
- Language: **Python3**
- Branch: **main**
- Region: **Singapore**
- Build command: `pip install -r requirements.txt`
- Start command: `uvicorn fastapi-backend:app --host 0.0.0.0 --port $PORT`
- Instance Type: **Free**
- Click `Deploy Web Service`
- The following image shows the service is live message and an associated URL



- Note the free instance warning
- Accessing the URL using the `curl` command or a browser we get the proper response (*https://webapp-fastapi-backend.onrender.com*)

```
{
  "message": "Server says Hello!"
```

```
}
```

Vue.js Frontend Code

- Create a Vue.js project

```
cd ~/projects/python/webapp-01
pip install nodeenv
nodeenv --node=20.6.0 --force .
source bin/activate
npm install -g axios @vue/cli vue-router
vue create vuejs-frontend
cd vuejs-frontend
```

- Remote the `README.md` and `src/components/HelloWorld.vue` files
- Modify the `src/main.js` file

```
import { createApp } from 'vue';
import App from './App.vue';
import router from './router';

createApp(App).use(router).mount('#app');
```

- Modify the `src/App.vue`

```
<template>
  <div id="app">
    <router-view></router-view>
  </div>
</template>

<script>
export default {
  name: 'App',
};
</script>

<style>
/* Add your global styles here */
```

```
</style>
```

- Create a directory `src/router` and create a router configuration file `src/router/index.js`

```
import { createRouter, createWebHistory } from 'vue-router';
import HomePage from '../views/HomePage.vue';
import CreateRecord from '../components/CreateRecord.vue';
import RetrieveRecords from '../components/RetrieveRecords.vue';
import UpdateRecord from '../components/UpdateRecord.vue';
import DeleteRecord from '../components/DeleteRecord.vue';

const routes = [
  { path: '/', component: HomePage },
  { path: '/create', component: CreateRecord },
  { path: '/retrieve', component: RetrieveRecords },
  { path: '/update', component: UpdateRecord },
  { path: '/delete', component: DeleteRecord },
];

const router = createRouter({
  history: createWebHistory(),
  routes,
});

export default router;
```

- Create a directory `src/views` and create `src/views/HomePage.vue` file there

```
<template>
  <div>
    <h1>Server says Hello!</h1>
    <p>Create, Retrieve, Update and Delete records - Redis Database</p>
    <nav>
      <router-link to="/create">Create</router-link>
      <router-link to="/retrieve">Retrieve</router-link>
      <router-link to="/update">Update</router-link>
      <router-link to="/delete">Delete</router-link>
    </nav>
    <router-view></router-view>
  </div>
```

```
</template>

<script>
export default {
  name: 'HomePage',
};
</script>

<style>
nav {
  display: flex;
  gap: 10px;
}
</style>
```

- Create `src/components/CreateRecord.vue` (*backend-url will be placed in the axios-method*)

```
<template>
  <div>
    <h2>Create Record</h2>
    <form @submit.prevent="addRecord">
      <div>
        <label>ID:</label>
        <input v-model="id" required />
      </div>
      <div>
        <label>Name:</label>
        <input v-model="name" required />
      </div>
      <div>
        <label>Age:</label>
        <input v-model="age" type="number" />
      </div>
      <div>
        <label>Weight:</label>
        <input v-model="weight" type="number" step="0.1" />
      </div>
      <button type="submit">Add</button>
      <button @click="cancel">Cancel</button>
    </form>
```

```
  </div>
</template>

<script>
import axios from 'axios';

export default {
  data() {
    return {
      id: '',
      name: '',
      age: '',
      weight: ''
    };
  },
  methods: {
    addRecord() {
      axios.post('https://webapp-fastapi-backend.onrender.com/add', {
        id: this.id,
        name: this.name,
        age: this.age,
        weight: this.weight
      })
      .then(response => {
        alert(response.data.message);
        this.resetForm();
      })
      .catch(error => {
        console.error(error);
      });
    },
    resetForm() {
      this.id = '';
      this.name = '';
      this.age = '';
      this.weight = '';
    },
    cancel() {
      this.$router.push('/');
    }
```

```
  }
};
</script>
```

- Create `src/components/RetrieveRecord.vue` (*backend-url will be placed in the axios-method*)

```
<template>
  <div>
    <h2>Retrieve Records</h2>
    <div v-if="records.length === 0">
      <p>There are no records in the Database</p>
      <button @click="goBack">Go Back</button>
    </div>
    <div v-else>
      <table>
        <thead>
          <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Age</th>
            <th>Weight</th>
          </tr>
        </thead>
        <tbody>
          <tr v-for="record in records" :key="record.id">
            <td>{{ record.id }}</td>
            <td>{{ record.name }}</td>
            <td>{{ record.age }}</td>
            <td>{{ record.weight }}</td>
          </tr>
        </tbody>
      </table>
      <button @click="goBack">Go Back</button>
    </div>
  </div>
</template>

<script>
import axios from 'axios';
```

```
export default {
  data() {
    return {
      records: []
    };
  },
  created() {
    this.fetchRecords();
  },
  methods: {
    fetchRecords() {
      axios.get('https://webapp-fastapi-backend.onrender.com/get-all')
      .then(response => {
        this.records = response.data;
      })
      .catch(error => {
        console.error(error);
      });
    },
    goBack() {
      this.$router.push('/');
    }
  }
};
</script>
```

- Create `src/components/UpdateRecord.vue` (*backend-url will be placed in the axios-method*)

```
<template>
  <div>
    <h2>Update Record</h2>
    <div v-if="records.length === 0">
      <p>There are no records in the Database</p>
      <button @click="goBack">Go Back</button>
    </div>
    <div v-else>
      <select v-model="selectedRecord">
        <option v-for="record in records" :key="record.id" :value="record">
          {{ record.name }}
        </option>
```

```
      </select>
      <button @click="modifyRecord">Modify</button>
    </div>
    <div v-if="selectedRecord">
      <form @submit.prevent="updateRecord">
        <div>
          <label>ID:</label>
          <input v-model="selectedRecord.id" disabled />
        </div>
        <div>
          <label>Name:</label>
          <input v-model="selectedRecord.name" required />
        </div>
        <div>
          <label>Age:</label>
          <input v-model="selectedRecord.age" type="number" />
        </div>
        <div>
          <label>Weight:</label>
          <input v-model="selectedRecord.weight" type="number" step="0.1" />
        </div>
        <button type="submit">Update</button>
        <button @click="cancel">Cancel</button>
      </form>
    </div>
  </div>
</template>

<script>
import axios from 'axios';

export default {
  data() {
    return {
      records: [],
      selectedRecord: null
    };
  },
  created() {
    this.fetchRecords();
```

```
  },
  methods: {
    fetchRecords() {
      axios.get('https://webapp-fastapi-backend.onrender.com/get-all')
      .then(response => {
        this.records = response.data;
      })
      .catch(error => {
        console.error(error);
      });
    },
    modifyRecord() {
      // Logic to modify the selected record
    },
    updateRecord() {
      axios.put(`https://webapp-fastapi-backend.onrender.com/update/${this.selectedRecord.id}`, {
        name: this.selectedRecord.name,
        age: this.selectedRecord.age,
        weight: this.selectedRecord.weight
      })
      .then(response => {
        alert(response.data.message);
        this.resetForm();
      })
      .catch(error => {
        console.error(error);
      });
    },
    resetForm() {
      this.selectedRecord = null;
      this.fetchRecords();
    },
    goBack() {
      this.$router.push('/');
    },
    cancel() {
      this.$router.push('/');
    }
  }
};
```

```
</script>
```

- Create `src/components/DeleteRecord.vue` (*backend-url will be placed in the axios-method*)

```
<template>
  <div>
    <h2>Delete Records</h2>
    <div v-if="records.length === 0">
      <p>There are no records in the Database</p>
      <button @click="goBack">Go Back</button>
    </div>
    <div v-else>
      <table>
        <thead>
          <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Age</th>
            <th>Weight</th>
            <th>Actions</th>
          </tr>
        </thead>
        <tbody>
          <tr v-for="record in records" :key="record.id">
            <td>{{ record.id }}</td>
            <td>{{ record.name }}</td>
            <td>{{ record.age }}</td>
            <td>{{ record.weight }}</td>
            <td><button @click="deleteRecord(record.id)">Delete</button></td>
          </tr>
        </tbody>
      </table>
      <button @click="goBack">Go Back</button>
    </div>
  </div>
</template>

<script>
import axios from 'axios';
```

```
export default {
  data() {
    return {
      records: []
    };
  },
  created() {
    this.fetchRecords();
  },
  methods: {
    fetchRecords() {
      axios.get('https://webapp-fastapi-backend.onrender.com/get-all')
      .then(response => {
        this.records = response.data;
      })
      .catch(error => {
        console.error(error);
      });
    },
    deleteRecord(id) {
      axios.delete(`https://webapp-fastapi-backend.onrender.com/delete/${id}`)
      .then(response => {
        alert(response.data.message);
        this.fetchRecords(); // Refresh the records list
      })
      .catch(error => {
        console.error(error);
      });
    },
    goBack() {
      this.$router.push('/');
    }
  }
};
</script>
```

- Modify the `package.json` file to include the `axios` and `vue-router` packages. Make sure the dependencies section has the following entry

```
{
 "dependencies": {
   // other dependencies
   "axios": "^0.21.1",
   "vue": "^3.0.0",
   "vue-router": "^4.0.0"
 }
}
```

- Create a public GitLab project *https://gitlab.com/web-application-01/webapp-vuejs-frontend*
- Run the following commands from the project-root directory of the Vue.js project

```
cd ~/projects/python/webapp-01/vuejs-frontend
git add .
git commit -m "Initial commit"
git remote add origin git@gitlab.com:web-application-01/webapp-vuejs-frontend.git
git push -u origin main
```

- Finally, close the `nodeenv`

```
cd ~/projects/python/webapp-01
deactivate_node
```

Vue.js Frontend Service

- Go back to *https://dashboard.render.com/*
- Select `+ New` > `Static Site`
- Select the `web-application-01/webapp-vuejs-frontend` project and click `Connect`
- Name: **webapp-vuejs-frontend**
- Branch: **main**
- Build command: `npm install && npm run build`
- Publish directory: **dist**
- Click `Deploy Static Site`
- The following image shows the service is live message and an associated URL

- Accessing the URL using a browser we get the application page (*https://webapp-vue-js-frontend.onrender.com*)

Revision #2
Created 13 September 2024 15:51:15 by Shafi
Updated 15 September 2024 16:22:02 by Shafi