React

Contents

Web Experiment and mySQL Database Setup

Creating React App

- Install NPM (Node package manager) and Node.js.
- Install Babel (JS compiler): `` `npm install babel-core babel-loader babel-preset-env babel-preset-react babel-webpack-plugin --save-dev` ``
- Install React JS: `` `npm install create-react-app` ``
- Create your project with React: `` `npx create-react-app [project name]` ``
- To start React app, cd to project folder and run: `` `npm start` ``

For new projects, only 4. onward is required.

Upload React App to GitHub Repo

- On Github website, create new repository, but DO NOT initialise repo with a README.
- Open Git bash cmd.
- cd to project folder, run the following:

```
git init
git add README.md
git add .
git commit -m "first commit"
git remote add origin https://github.com/[username]/[project name].git
git push -u origin master
```

Link GitHub Repo to Scalingo App

- Ensure that in package.json:

```
"scripts": { "start": "node server.js" }
```

- Add a `server.js` file to the project folder that includes:

```
  const express = require("express");
  const path = require("path");
  const app = express();
  const port = process.env.PORT;
  app.use(express.static(path.join(__dirname, "build")));
  -app.get("/", function (req, res) {
    +app.get("/*", function (req, res) {
      res.sendFile(path.join(__dirname, "build", "index.html"));
   });
  });
  app.listen(port, () => console.log("Listening on Port", port));
```

- On Scalingo, create new app, enter app name, ignore deployment page.
- In the app folder, go to Code, scroll down to Deploy with GitHub.
- Choose repo. Note: project needs to be on Github!
- Choose Manual deploy.

Install Flask & SQL dependencies (for mySQL database python backend)

- Install python3 and pip. To check, run: ``` `python --version` ``` and ``` `pip --version` ```
- If python is ver 3.3 and above, there is no need to install `virtualenv`. Just use `venv` to create a virtual environment in a project folder that already exists:

```
  $ cd [project folder]
  $ python -m venv [virtual env folder name]
```

- To activate virtual env (in Windows):

```
  $ [virtual env folder name]\Scripts\activate
```

- Install dependencies (look at requirements.txt) e.g. Flask:

```
  $ pip install Flask
```

- To deactivate virtual env:

```
  $ dectivate
```

- If you are using deploying the database on localhost, remember to always activate the virtual environment before deploying `app.py`

Deploying Python Backend

On Localhost

- In SQL Workbench, ensure that you have set up a schema (localhost db) labelled with database name
- Add `config.js` file to Components folder of ReactJS task code containing:

```
export const DATABASE_URL = "http://localhost:5000";
```

- To post data from task to localhost database, in ReactJS task code, add:

```
fetch(`${DATABASE_URL}/[task_data]/` + [participant ID], {
  method: "POST",
  headers: {
    Accept: "application/json",
    "Content-Type": "application/json",
  },
  body: JSON.stringify([data string]),
});
```

- Remember to import the `config.js` file into task code:

```
import { DATABASE_URL } from "./config";
```

- In `app.py` of python backend folder, ensure:

```
app.config['SQLALCHEMY_DATABASE_URI'] = config.get('Database Parameters','database_url')
```

and

```
@app.route('/testmethod', methods=['GET', 'POST'])
def mytest():
result = dict()
result['test'] = 'ok'
return jsonify(result), 200
```

- In `config.text`, ensure:

```
[Database Parameters]
database_url: mysql+pymysql://[user/root]:[password]@[localhost]/[database name]
```

- Then run:

```
$ cd [python backend folder]
$ [virtual env folder name]\Scripts\activate
$ python app.py
```

- Go to `` `http://localhost:5000/testmethod ` `` to test if connection works. It should show `{"test":"ok"}`.

On Scalingo

- In `config.js` file of Components folder of ReactJS task code, configure:

```
export const DATABASE_URL = "https://[database name].osc-fr1.scalingo.io";
```

- In `app.py` of python code, configure:

```
app.config['SQLALCHEMY_DATABASE_URI'] = os.environ.get('DATABASE_URL')
```

- Upload python backend to a Github repo and upload as Scalingo app (see above)
- Ensure that you have included the SQL database add-on (important!!). Go to 'Addons' in your new api application, choose 'MySQL' and then a plan depending on your requirements (e.g., Starter 512M) .

Connecting to Scalingo mySQL database

- Add your SSH public key to Scalingo. This has to be added with every new device (thus new public key).
- Edit 28/03/23: Scalingo has deprecated the use of ssh-rsa keys, instead ensure that key pairs are either rsa-sha2 (512/256) or ssh-ed25519
- If using ssh-ed25519, note that MYSQL Workbench needs to be version 8.0.32 and no password is set for the key, otherwise the connection will not be made

- Get environment variables of Scalingo python backend app, e.g.:

```
SCALINGO_MYSQL_URL=mysql://<username>:<password>@<hostname>:<port>/<user>
```

- Edit out `useSSL=true&verifyServerCertificate=false` if necessary.

- To connect database to mySQL Workbench GUI, click setup new connection and enter the following details based on Scalingo's environment details:

```
Connection Method: Standard TCP/IP over SSH
SSH Hostname: ssh.osc-fr1.scalingo.com
SSH Username: git
SSH Password: (empty)
SSH Key File: /Users/.ssh/rd_rsa
MySQL Hostname: <hostname>
MySQL Server Port: <port>
Username: <username>
Password: <password>
```

Scalingo deployment from Github

- Currently Scalingo only allows selection from 20 repositories from your Github account, the others will not be displayed due to pagination issue on Scalingo's side. In case where the repo you want is not available for selection, use CLI to create the app for deployment
- Do do this, install CLI tool and add it to git-bash path https://doc.scalingo.com/platform/cli/start
- Use SSH key or generate an api-token for login https://doc.scalingo.com/platform/cli/introduction
- Use gitbash, login via `scalingo login --api-token <token>`
- Follow https://doc.scalingo.com/platform/deployment/deploy-with-github for creating app
- Note if OS is windows, `--region osc-fr1` needs to be added with every `scalingo` command. See: https://github.com/Scalingo/cli/issues/809

- If app is already created and originally pushed via git, cd to project folder and run `git remote add scalingo git@ssh.osc-fr1.scalingo.com:[my-app].git` replacing `[my-app]` as stated in the app's deployment config
- then run `git push scalingo main` to re-deploy
- Note that 'main' can also be 'master' depending on the branch to be specified

Retrieved from "https://devcompsy.org/wiki/index.php?title=React&oldid=494"

This page was last edited on 6 June 2023, at 13:51.