
Flask React SPA

Status

build unknown

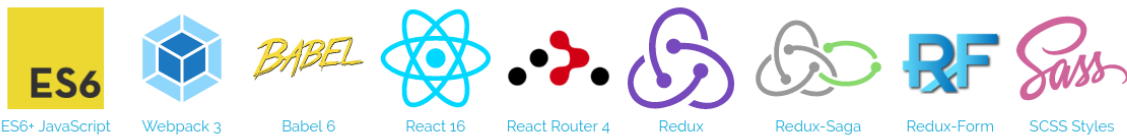
FlaskReact.SPA ARTICLES SERIES STYLES CONTACT

SIGN UP LOGIN

Welcome to Flask React SPA!

A production-ready boilerplate built with Python 3, Flask and ES6 React/Redux

Frontend Stack



Backend Stack



Production Deployment



React v16 Frontend

The frontend is heavily inspired by react boilerplate, and indeed borrows a good chunk of boilerplate from it.

- React Router v4
- Redux, Redux-Saga and Redux-Form for handling state and side effects
- Webpack 3 and Babel 6
 - Hot Module Reloading
 - Tree Shaking
 - Code Splitting (asynchronous components via react-loadable)

Entry point is at `frontend/app/index.js`.

Flask Backend

- SQLAlchemy ORM with Flask-SQLAlchemy and migrations provided by Flask-Alembic
- RESTful APIs provided by a customized integration between Flask-RESTful and Flask-Marshmallow
- Flask-Security provides authentication, authorization, registration and change/forgot password functionality
 - User session management via Flask-Login
 - User permissions and roles via Flask-Principal
 - Secrets encryption via passlib and itsdangerous
 - CSRF protection via Flask-WTF
- Flask-Admin integrated for painless model CRUD administration
- Flask-Session for server-side sessions
- Celery for asynchronous tasks, such as sending emails via Flask-Mail

The backend is structured using the Application Factory Pattern, in conjunction with a little bit of declarative configuration in `backend/config.py` (for ordered registration of extensions, and auto-detection of views, models, serializers, model admins and cli commands). The entry point is the `create_app()` method in `backend/app.py` (`wsgi.py` in production).

Ansible Production Deployment

- CentOS/RHEL 7.x
- Python 3.6 (provided by the IUS Project)
- PostgreSQL 9.6
- Redis 3.2
- NGINX + uWSGI + supervisord
- Lets Encrypt HTTPS
- Email sending via Postfix with SSL and OpenDKIM

Local Development QuickStart:

Using docker-compose

Dependencies:

- `docker` and `docker-compose` (at least docker engine v1.13)

```
1 # install
2 $ git clone git@github.com:briancappello/flask-react-spa.git
3 $ cd flask-react-spa
4
5 # configure (the defaults are fine for development)
6 $ edit `backend/config.example.py` and save as `backend/config.py`
7 $ edit `frontend/app/config.example.js` and save as `frontend/app/
  config.js`
8
9 # run it
10 $ docker-compose up --build # grab a coffee; bootstrapping takes a
    while the first time
```

Once it's done building and everything has booted up:

- Access the app at: <http://localhost:8888>
- Access MailHog at: <http://localhost:8025>
- Access the docs at: <http://localhost:5500>
- Webpack Bundle Analyzer: <http://localhost:5555>
- The API (eg for testing with CURL): <http://localhost:5000>
- And last but not least, the database is exposed on port 5442

Running locally

This assumes you're on a reasonably standard *nix system. Windows *might* work if you know what you're doing, but you're on your own there.

Dependencies:

- Python 3.6+
- Your virtualenv tool of choice (strongly recommended)
- PostgreSQL or MariaDB (MySQL)
- Redis (used for sessions persistence and the Celery tasks queue)
- node.js & npm (v6 or later recommended, only required for development)
- MailHog (not required, but it's awesome for testing email related tasks)

```
1 # install
2 $ git clone git@github.com:briancappello/flask-react-spa.git
3 $ cd flask-react-spa
4 $ mkvirtualenv -p /path/to/python3 flask_react_spa
5 $ pip install -r requirements.txt
6 $ pip install -r requirements-dev.txt # for tests and sphinx docs
7
8 # configure
```

```
9 $ edit `backend/config.example.py` and save as `backend/config.py`
10 $ edit `frontend/app/config.example.js` and save as `frontend/app/
    config.js`
11
12 # set up database
13 $ sudo -u postgres -i psql
14 postgres=# CREATE USER flask_api WITH PASSWORD 'flask_api';
15 postgres=# CREATE DATABASE flask_api;
16 postgres=# GRANT ALL PRIVILEGES ON DATABASE flask_api TO flask_api;
17 postgres=# \q # (quit)
18
19 # run db migrations
20 $ python manage.py db upgrade
21
22 # load db fixtures (optional)
23 $ python manage.py db fixtures fixtures.json
24
25 # frontend dev server:
26 $ npm install
27 $ npm run start
28
29 # backend dev server:
30 $ python manage.py run
31
32 # backend celery workers:
33 $ python manage.py celery worker
34 $ python manage.py celery beat
```

Full Documentation

Run `make docs` and browse to <http://localhost:5500>

Sources are in the `/docs` folder.

FIXME: publish to GitHub Pages.

License

MIT