Project Generation - Template

You can use a project generator to get started, as it includes a lot of the initial set up, security, database and first API endpoints already done for you.

A project generator will always have a very opinionated setup that you should update and adapt for your own needs, but it might be a good starting point for your project.

Full Stack FastAPI PostgreSQL

GitHub: https://github.com/tiangolo/full-stack-fastapi-postgresql

Full Stack FastAPI PostgreSQL - Features

- Full **Docker** integration (Docker based).
- Docker Swarm Mode deployment.
- **Docker Compose** integration and optimization for local development.
- **Production ready** Python web server using Uvicorn and Gunicorn.
- Python **FastAPI** backend:
  - **Fast**: Very high performance, on par with **NodeJS** and **Go** (thanks to Starlette and Pydantic).
  - **Intuitive**: Great editor support. Completion everywhere. Less time debugging.
  - **Easy**: Designed to be easy to use and learn. Less time reading docs.
  - **Short**: Minimize code duplication. Multiple features from each parameter declaration.
  - **Robust**: Get production-ready code. With automatic interactive documentation.
  - **Standards-based**: Based on (and fully compatible with) the open standards for APIs: OpenAPI and JSON Schema.
  - **Many other features** including automatic validation, serialization, interactive documentation, authentication with OAuth2 JWT tokens, etc.
- **Secure password** hashing by default.
- **JWT token** authentication.
- **SQLAlchemy** models (independent of Flask extensions, so they can be used with Celery workers directly).
- Basic starting models for users (modify and remove as you need).
- **Alembic** migrations.
- **CORS** (Cross Origin Resource Sharing).
- **Celery** worker that can import and use models and code from the rest of the backend selectively.
- REST backend tests based on **Pytest**, integrated with Docker, so you can test the full API interaction, independent on the database. As it runs in Docker, it can build a new data store from scratch each time (so you can use ElasticSearch, MongoDB, CouchDB, or whatever you want, and just test that the API works).
- Easy Python integration with **Jupyter Kernels** for remote or in-Docker development with extensions like Atom Hydrogen or Visual Studio Code Jupyter.
- **Vue** frontend:
  - Generated with Vue CLI.
  - **JWT Authentication** handling.
  - Login view.
  - After login, main dashboard view.
  - Main dashboard with user creation and edition.
  - Self user edition.
  - **Vuex**.
  - **Vue-router**.
  - **Vuetify** for beautiful material design components.
  - **TypeScript**.

    - Docker server based on **Nginx** (configured to play nicely with Vue-router).
    - Docker multi-stage building, so you don't need to save or commit compiled code.
    - Frontend tests ran at build time (can be disabled too).
    - Made as modular as possible, so it works out of the box, but you can re-generate with Vue CLI or create it as you need, and re-use what you want.
- **PGAdmin** for PostgreSQL database, you can modify it to use PHPMyAdmin and MySQL easily.
- **Flower** for Celery jobs monitoring.
- Load balancing between frontend and backend with **Traefik**, so you can have both under the same domain, separated by path, but served by different containers.
- Traefik integration, including Let's Encrypt **HTTPS** certificates automatic generation.
- GitLab **CI** (continuous integration), including frontend and backend testing.

Full Stack FastAPI Couchbase

GitHub: https://github.com/tiangolo/full-stack-fastapi-couchbase

⚠️ **WARNING** ⚠️

If you are starting a new project from scratch, check the alternatives here.

For example, the project generator Full Stack FastAPI PostgreSQL might be a better alternative, as it is actively maintained and used. And it includes all the new features and improvements.

You are still free to use the Couchbase-based generator if you want to, it should probably still work fine, and if you already have a project generated with it that's fine as well (and you probably already updated it to suit your needs).

You can read more about it in the docs for the repo.

Full Stack FastAPI MongoDB

...might come later, depending on my time availability and other factors. 😅 🎉

Machine Learning models with spaCy and FastAPI

GitHub: https://github.com/microsoft/cookiecutter-spacy-fastapi

Machine Learning models with spaCy and FastAPI - Features

- **spaCy** NER model integration.
- **Azure Cognitive Search** request format built in.
- **Production ready** Python web server using Uvicorn and Gunicorn.
- **Azure DevOps** Kubernetes (AKS) CI/CD deployment built in.
- **Multilingual** Easily choose one of spaCy's built in languages during project setup.
- **Easily extensible** to other model frameworks (Pytorch, Tensorflow), not just spaCy.