Build A SaaS App WITH FLASK

Build a SAAS App With Flask

Learn how to build a production ready web app with Flask and Docker. Level up and win that dream software developer job.

Nick Janetakis

Contents

Preface

Hi, I'm Nick Janetakis and I'm the creator of the Build a SAAS App with Flask course which was released in June 2016.

Feel free to contact me at nick.janetakis@gmail.com if you have any questions, or check out my website to learn more about me. http://nickjanetakis.com

Please do not distribute the source code or e-book because it's meant to be for paying students only, such as yourself. Thanks!

Chapter 1

Welcome to the Course

Chapter 2

Is Flask Right for You?

Chapter 3

Application Overview

3.1 Visualizing the Application's Architecture

gunicorn

http://docs.gunicorn.org/en/stable/

Flask

http://flask.pocoo.org/

Celery

http://docs.celeryproject.org/en/latest/

Redis

http://redis.io/

PostgreSQL

http://www.postgresql.org/

Free course on learning more about SQL

- KhanAcademy - Intro to SQL

SQLAlchemy

http://www.sqlalchemy.org/

Click

http://click.pocoo.org/6/

Stripe

https://stripe.com/

Docker

https://www.docker.com/

Chapter 4

Preparing to Follow Along

4.1 Installing a Code Editor

Free code editors

- https://www.sublimetext.com/3
- https://code.visualstudio.com
- https://atom.io
- http://www.vim.org/
- https://www.gnu.org/software/emacs/

Configure Sublime Text 3 like mine (if you want!)

http://nickjanetakis.com/blog/25-sublime-text-3-packages-for-polyglot-programmers

Additional packages that I installed which are not in the blog post

- INI
- Hyperion for gettext

Chapter 5

Getting Familiar with Docker

5.1 Why Is It worth Using Docker?

Save yourself from years of turmoil by using Docker today

http://nickjanetakis.com/blog/save-yourself-from-years-of-turmoil-by-using-docker-today

5.2 Installing Docker

Docker 1.12 is fully compatible with 1.11, so feel free to use 1.12 instead of 1.11 which is recommended below.

5.2.1 OSX

Download Docker Toolbox for OSX

https://github.com/docker/toolbox/releases/download/v1.11.0/DockerToolbox-1.11.0.pkg

Docker Toolbox installation video guide

https://www.youtube.com/watch?v=lNkVxDSRo7M

- **When he says to download the Docker Toolbox from the site, use my download link instead** because we want to use Docker 1.11 in this course. You can upgrade to a newer version after you finish the course.

Docker quickstart terminal

Throughout this course, I will say things on video such as "open up a terminal". Since you're using OSX, that will always mean to open up a Docker quickstart terminal.

5.2.2 Windows

Download Docker Toolbox for Windows

https://github.com/docker/toolbox/releases/download/v1.11.0/DockerToolbox-1.11.0.exe

Docker Toolbox installation video guide

https://www.youtube.com/watch?v=S7NVloq0EBc

- **When he says to download the Docker Toolbox from the site, use my download link instead** because we want to use Docker 1.11 in this course. You can upgrade to a newer version after you finish the course.

- Make sure at least **Docker Compose, VirtualBox and Git is checked during installation** because you will receive bash.exe lookup errors if Git is not installed and the others are mandatory.

Docker quickstart terminal

Throughout this course, I will say things on video such as "open up a terminal". Since you're using Windows, that will always mean to open up a Docker quickstart terminal.

5.2.3 Linux

Ubuntu Docker installation (xubuntu and friends also work)

```
sudo apt-get update -y \
  && sudo apt-get install -y curl apt-transport-https ca-certificates aufs-tools \
  && sudo apt-key adv \
       --keyserver hkp://p80.pool.sks-keyservers.net:80 \
       --recv-keys 58118E89F3A912897C070ADBF76221572C52609D \
  && echo "deb https://apt.dockerproject.org/repo ubuntu-$(lsb_release -cs) main" | \
       sudo tee /etc/apt/sources.list.d/docker.list \
  && sudo apt-get update -y \
  && sudo apt-get install -y docker-engine=1.11.1-0~"$(lsb_release -cs)" \
  && sudo usermod -aG docker $(whoami)
```

- Make sure to **completely logout** of your OS before moving on

Other distros Docker installation

https://docs.docker.com/engine/installation/

Docker Compose installation

```
curl -L \
  https://github.com/docker/compose/releases/download/1.7.0/docker-compose-Linux-x86_64 > \
  /tmp/docker-compose && \
  chmod +x /tmp/docker-compose && \
  sudo mv /tmp/docker-compose /usr/local/bin
```

5.3 Making Sure Docker Works on Your System

OSX and Windows users should launch the Docker quickstart terminal when running all future Docker related commands.

Check your Docker version

```
docker --version
```

Check your Docker Compose version

```
docker-compose --version
```

Chapter 6

Creating a Base Flask App

6.1 Exploring the App's Package Dependencies

Upgrading to newer versions of Flask

This course was recorded against Flask 0.10 which was the latest version for nearly 3 years.

0.11 was released literally 1 day after I finished recording this course and if you want to upgrade to that because it has features you're interested in, you can just change the 0.10 to 0.11 in `requirements.txt` and you're good to go.

When Flask 1.0 gets released which will be the next major release, I will add a new bonus lecture that goes over upgrading the code base if there are any backwards incompatible changes.

6.2 Investigating the Dockerfile

Python on the Docker Hub

https://hub.docker.com/_/python

6.3 Running the Flask Application

Build and start everything with Docker Compose

```
docker-compose up --build
```

Visit the site in your browser

http://localhost:8000/

- If you're using OSX or Windows then use your **Docker Machine IP address** instead of localhost. It is probably http://192.168.99.100:8000/.
- This IP address is at the top of your Docker quickstart terminal and can also be found by running: `docker-machine ip`

Restart Docker Compose

```
# Press CTRL+C a few times
docker-compose stop
docker-compose up
```

View all of your Docker images

Run this in a second terminal tab.

```
docker images
```

View running Docker containers

Run this in a second terminal tab.

Both commands below do basically the same thing.

```
docker-compose ps
docker ps
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

6.4 Dealing with Configuration Settings

Start everything with Docker Compose

```
docker-compose up
```

Make a configuration change and check it in your browser

Follow along with the video.

Create instance/settings.py

Follow along with the video.

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 7

Blueprints and Jinja 2 Templates

7.1 Creating Our First Flask Blueprint

Start everything with Docker Compose

```
docker-compose up
```

Visit the site in your browser

http://localhost:8000/

- This will be the last warning that OSX or Windows users will need to replace `localhost` with their Docker Machine IP address.

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

7.2 Creating the Home Page

Start everything with Docker Compose

```
docker-compose up
```

Visit the site in your browser

http://localhost:8000/

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

7.3 Adding a Few Additional Pages

Start everything with Docker Compose

```
docker-compose up
```

Visit the site in your browser

http://localhost:8000/

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 8

Testing and Code Quality

8.1 Getting Comfortable Writing Tests

Build and start everything with Docker Compose

```
docker-compose up --build
```

Run the test suite

Run this in a second terminal tab.

```
docker-compose exec website py.test snakeeyes/tests
```

You will get 3 failing tests (that's ok).

Add SERVER_NAME to config/settings.py

Follow along with the video.

Run the test suite again

Run this in a second terminal tab.

```
docker-compose exec website py.test snakeeyes/tests
```

All 3 tests should pass. If not, make sure you use your Docker Machine IP address instead of localhost for `SERVER_NAME` if you're not running Docker natively.

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

8.2 Investigating Our Code Test Coverage

Start everything with Docker Compose

```
docker-compose up
```

Run the test coverage tool

Run this in a second terminal tab.

```
docker-compose exec website py.test --cov-report term-missing --cov snakeeyes
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

8.3 Performing Static Analysis on the Code Base

Start everything with Docker Compose

```
docker-compose up
```

Run the flake8 tool

Run this in a second terminal tab.

```
docker-compose exec website flake8 .
```

Run the flake8 tool but ignore init.py files

Run this in a second terminal tab.

```
docker-compose exec website flake8 . --exclude __init__.py
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 9

Creating a CLI Script

9.1 Getting Familiar with Click

Click's documentation

http://click.pocoo.org/6/#documentation-contents

9.2 Running the Commands We Created

Build and start everything with Docker Compose

```
docker-compose up --build
```

Get a list of commands

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes
```

Run flake8

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes flake8
```

Get the help menu for a specific command

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes flake8 --help
```

Run flake8 but don’t skip init.py files

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes flake8 --no-skip-init
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 10

Using Our First Flask Extension

10.1 Debug Toolbar

Build and start everything with Docker Compose

```
docker-compose up --build
```

Visit the site in your browser

http://localhost:8000/

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 11

Creating a Contact Form

11.1 Configuring the App to Send e-mail

Build and start everything with Docker Compose

```
docker-compose up --build
```

Flask-Mail's documentation

https://pythonhosted.org/Flask-Mail/

Input your real e-mail credentials into instance/settings.py

Follow along with the video.

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

11.2 Looking into the Templates and Macros

Start everything with Docker Compose

```
docker-compose up
```

11.2.1 Visit the contact page in your browser

http://localhost:8000/contact

11.2.2 Did you get a bad CSRF token error?

If you're using Docker Machine and you put in your IP address for the **`SERVER_NAME`** then chances are you received a CSRF token error if you're using Chrome or any other webkit based browser.

This is actually a bug with webkit which goes against the spec which should allow cookies to be set for IP based or non-correct TLD host names.

I wasn't affected on camera because I'm using **`localhost`** and Flask is hard coded to prevent this error for **`localhost`** specifically.

To Fix this error

Please open up your **`/etc/hosts`** file with elevated privileges such as sudo or Administrator.

OSX users can find it in **`/etc/hosts`** and can run **`sudo nano /etc/hosts`** to open it.

Windows users can find it in **`C:/Windows/System32/drivers/etc/hosts`** and will need to open it as an Administrator with Notepad or something else.

Add this line to the bottom of the file and save it:

`192.168.99.100 local.docker`

If your Docker Machine IP address is not **`192.168.99.100`** *then please replace it.*

The above change is going to map that IP address to the local.docker host name on your work station.

Then open up your `instance/settings.py` file and change `SERVER_NAME` to be `local.docker:8000` and save the file.

Then head over to your terminal and kill Docker Compose by pressing `CTRL+C` and then run `docker-compose stop`.

Then start everything up by running: `docker-compose up`.

Then access `http://local.docker:8000/contact` in your browser and it should work.

At this point, you'll never need to do these steps again. Every time you copy over your previous `instance/settings.py` file, you will have the correct `SERVER_NAME` in place.

From this point forward you'll always access `http://local.docker:8000` in your browser.

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

11.3 Our First Taste of Celery

Start everything with Docker Compose

```
docker-compose up
```

Visit the contact page in your browser

http://localhost:8000/contact

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

11.4 Running Celery with Docker Compose

Visit the Docker Hub

https://hub.docker.com
https://hub.docker.com/_/redis/

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

View your Docker networks

```
docker network ls
```

Remove the snakeeyes network (don't run it)

```
docker network rm snakeeyes_default
```

View your Docker volumes

```
docker volume ls
```

Remove the Redis volume (don't run it)

```
docker volume rm snakeeyes_redis
```

11.5 Confirming It Works with Tests

Start everything with Docker Compose

```
docker-compose up
```

Run the test suite

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes test
```

Run the test coverage

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes cov
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 12

Creating a Complete User System

12.1 Configuring the App to Handle Users

Visit the Docker Hub

https://hub.docker.com/_/postgres/

Copy your instance settings from the previous section

Goto the previous section's source code and copy your **`instance/settings.py`** *file's contents and replace them in this section's* **`instance/settings.py`** *file.*

12.2 Initializing the Database

Build and start everything with Docker Compose

```
docker-compose up --build
```

Check out the new db commands

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes db
```

Reset the database with a test database

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes db reset --with-testdb
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

12.3 Logging Users in and Out

Start everything with Docker Compose

```
docker-compose up
```

Visit the home page

http://localhost:8000

SQLAlchemy's documentation

http://docs.sqlalchemy.org/en/rel_1_0/

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

12.4 Registering New Users

Start everything with Docker Compose

```
docker-compose up
```

Visit the sign up page

http://localhost:8000/signup

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

12.5 Welcoming New Users

Start everything with Docker Compose

```
docker-compose up
```

Visit the welcome page

http://localhost:8000/welcome

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

12.6 Allowing Users to Update Their Settings

Start everything with Docker Compose

```
docker-compose up
```

Visit the settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

12.7 Dealing with Password Resets

Start everything with Docker Compose

```
docker-compose up
```

Visit the login page

http://localhost:8000/login

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

12.8 Modifications to Previous Blueprints

Start everything with Docker Compose

```
docker-compose up
```

Visit the home page

http://localhost:8000

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

12.9 Confirming It Works with Tests

Start everything with Docker Compose

```
docker-compose up
```

Run the test coverage

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes cov
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 13

Creating a Custom Admin Dashboard

13.1 Adding the Admin Blueprint

Copy your instance settings from the previous section

Goto the previous section's source code and copy your **`instance/settings.py`** *file's contents and replace them in this section's* **`instance/settings.py`** *file.*

13.2 Viewing the Main Dashboard

Build and start everything with Docker Compose

```
docker-compose up --build
```

Visit the login page

http://localhost:8000/login

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

13.3 Listing Users

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin page

http://localhost:8000/admin

Documentation for Moment.js

http://momentjs.com/

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

13.4 Editing Users

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin page

http://localhost:8000/admin

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

13.5 Generating Fake Users with the CLI

Start everything with Docker Compose

```
docker-compose up
```

Check out the new CLI command

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes add
```

Generate a bunch of users

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes add users
```

Visit the admin users page

http://localhost:8000/admin/users

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

13.6 Searching and Sorting Users

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin users page

http://localhost:8000/admin/users

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

13.7 Deleting Users

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin users page

http://localhost:8000/admin/users

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

13.8 Confirming It Works with Tests

Start everything with Docker Compose

```
docker-compose up
```

Run the test coverage

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes cov
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 14

Logging, Middleware and Error Handling

14.1 Tracking Response Times for All Requests

Copy your instance settings from the previous section

Goto the previous section's source code and copy your ***`instance/settings.py`*** *file's contents and replace them in this section's* ***`instance/settings.py`*** *file.*

Build and start everything with Docker Compose

```
docker-compose up --build
```

Visit the home page

http://localhost:8000

Documentation for gunicorn's access log format

http://docs.gunicorn.org/en/stable/settings.html#access-log-format

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

14.2 Using Flask's Logger

Start everything with Docker Compose

```
docker-compose up
```

Visit the home page

http://localhost:8000

A few Flask logging examples

```
current_app.logger.debug('Foobar debug log.')
current_app.logger.error('Foobar error log which is more serious than debug.')
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

14.3 Integrating Google Analytics

Start everything with Docker Compose

```
docker-compose up
```

Visit the home page

http://localhost:8000

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

14.4 Custom Error Pages

Start everything with Docker Compose

```
docker-compose up
```

Visit a page that does not exist

http://localhost:8000/asdf

Throwing custom errors in Flask to test error codes

```
# Add this to one of your routes
from flask import abort
abort(500)
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

14.5 Handling Exceptions in Production

Start everything with Docker Compose

```
docker-compose up
```

Visit the home page

http://localhost:8000

- Create invalid Python syntax in the home page's route
- Set `DEBUG = False` in `config/settings.py` and save the file
- Reload the page and check your e-mail for the exception
- Set `DEBUG = True` in `config/settings.py` and save the file

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 15

Quality of Life CLI Improvements

15.1 Creating Secure Tokens

Build and start everything with Docker Compose

```
docker-compose up --build
```

Generate a secure token

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes secret
```

Generate a 64 byte secure token

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes secret 64
```

Generate an invalid byte secure token

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes secret foo
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

15.2 Viewing All Route Endpoints

Start everything with Docker Compose

```
docker-compose up
```

Visit the home page

https://localhost:8000

View the routes

```
docker-compose exec website snakeeyes routes
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

15.3 Breaking down Lines of Code

Start everything with Docker Compose

```
docker-compose up
```

Count the lines of code

```
docker-compose exec website snakeeyes loc
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 16

Accepting Recurring Payments

16.1 Configuring the App to Handle Payments

Copy your instance settings from the previous section

Goto the previous section's source code and copy your **`instance/settings.py`** *file's contents and replace them in this section's* **`instance/settings.py`** *file.*

Stripe's API documentation for upgrades

https://stripe.com/docs/upgrades

16.2 Creating Subscription Plans

Build and start everything with Docker Compose

```
docker-compose up --build
```

Reset and seed the database

The database schema changed since the last section, so we need to do this.

```
docker-compose exec website snakeeyes db reset --with-testdb
docker-compose exec website snakeeyes add all
```

Stripe's API documentation for plans

https://stripe.com/docs/api#plans

Stripe's supported currencies

https://support.stripe.com/questions/which-currencies-does-stripe-support

Sync the plans to Stripe

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes stripe sync_plans
```

List plans that are active on Stripe

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes stripe list_plans
```

Delete the bronze plan on Stripe

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes stripe delete_plans bronze
```

List the plans again (notice bronze is missing)

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes stripe list_plans
```

Sync the plans to Stripe

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes stripe sync_plans
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.3 Rendering Pricing Tables

Start everything with Docker Compose

```
docker-compose up
```

Visit the home page

http://localhost:8000

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.4 Subscribing to a Plan (Front-End)

Start everything with Docker Compose

```
docker-compose up
```

Visit the create subscription page (expecting a redirect)

http://localhost:8000/subscription/pricing

- **Valid fake credit card**: 4242424242424242
- **Valid fake CVC**: 123
- Additional fake data you can use: https://stripe.com/docs/testing#cards

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.5 Subscribing to a Plan (Back-End)

Start everything with Docker Compose

```
docker-compose up
```

Visit the create subscription page (expecting a redirect)

http://localhost:8000/subscription/create?plan=gold

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.6 Updating Your Payment Method

Start everything with Docker Compose

```
docker-compose up
```

Visit the settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.7 Informing Users of Expiring Credit Cards

Start everything with Docker Compose

```
docker-compose up
```

Visit the settings page

http://localhost:8000/settings

Celery's documentation for periodic tasks

http://docs.celeryproject.org/en/latest/userguide/periodic-tasks.html

Running the Celery and Beat worker independently

This would be better suited for production:

```
  celery_beat:
    build: .
    command: celery beat -l info -A snakeeyes.blueprints.contact.tasks
    env_file:
      - '.env'
    volumes:
      - '.:/snakeeyes'
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.8 Updating Your Subscription Plan

Start everything with Docker Compose

```
docker-compose up
```

Visit the settings page

http://localhost:8000/settings

Stripe's explanation of how billing works when plans change

https://support.stripe.com/questions/handling-subscription-changes

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.9 Cancelling Your Subscription

Start everything with Docker Compose

```
docker-compose up
```

Visit the settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.10 Extending the Admin Dashboard

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin page

http://localhost:8000/admin

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.11 Managing Coupons in the Admin

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin page

http://localhost:8000/admin

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.12 Expiring and Subscribing with Coupons

Start everything with Docker Compose

```
docker-compose up
```

Visit the coupons page

http://localhost:8000/admin/coupons

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.13 Managing Subscriptions in the Admin

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin page

http://localhost:8000/admin

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.14 Generating Random Invoices

Start everything with Docker Compose

```
docker-compose up
```

Generate random invoices

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes add invoices
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.15 Reviewing Your Billing History

Start everything with Docker Compose

```
docker-compose up
```

Visit the settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.16 Integrating Stripe Webhooks

Start everything with Docker Compose

```
docker-compose up
```

Download and install ngrok for your OS

https://ngrok.com/download

Start ngrok

Run this in a second terminal tab.

```
ngrok http localhost:8000
```

- Use your Docker Machine IP address (or local.docker) instead of localhost if you need to

Change the `SERVER_NAME` in `instance/settings.py` for ngrok

```
# Make sure you replace '175c5f67' with your ngrok sub-domain
SERVER_NAME = '175c5f67.ngrok.io'
```

- Make sure to restart Docker Compose after making this change

Configure the webhook in Stripe's dashboard

https://dashboard.stripe.com/account/webhooks

- WebhookURL: http://175c5f67.ngrok.io/stripe_webhook/event
- Make sure you replace 175c5f67 with your ngrok sub-domain
- Make sure **mode** is set to **Test**
- Select the **invoice.created** event

Stop ngrok

Run this in a second terminal tab.

```
# Press CTRL+C
```

Revert the `SERVER_NAME` in `instance/settings.py` back to its original value

Follow along with the video.

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

16.17 Confirming It Works with Tests

Start everything with Docker Compose

```
docker-compose up
```

Run the test suite

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes test
```

16.18 Sales Charts, Refunds and More

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Visit your dashboard on Stripe's site

http://dashboard.stripe.com/

Chapter 17

Building the Snake Eyes Game

17.1 Configuring the App to Handle Betting

Copy your instance settings from the previous section

Goto the previous section's source code and copy your **`instance/settings.py`** *file's contents and replace them in this section's* **`instance/settings.py`** *file.*

Flask-Limiter's rate limiting strategies

http://flask-limiter.readthedocs.org/en/stable/#rate-limiting-strategies

17.2 Placing Bets (Front-End)

Build and start everything with Docker Compose

```
docker-compose up --build
```

Reset and seed the database

The database schema changed since the last section, so we need to do this.

```
docker-compose exec website snakeeyes db reset --with-testdb
docker-compose exec website snakeeyes add all
```

Visit the home page

http://localhost:8000

Wiki entry for Miscellaneous Symbols

https://en.wikipedia.org/wiki/Miscellaneous_Symbols

- Search the page for “Die face-1” and notice the Unicode value

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

17.3 Placing Bets (Back-End)

Start everything with Docker Compose

```
docker-compose up
```

Flask-Limiter’s documentation

http://flask-limiter.readthedocs.org/en/stable/

Visit the place bet page

http://localhost:8000/bet/place

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

17.4 Generating Random Bets

Start everything with Docker Compose

```
docker-compose up
```

Generate random bets

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes add bets
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

17.5 Viewing Betting History

Start everything with Docker Compose

```
docker-compose up
```

Visit the user settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

17.6 Modifying the Admin User Details

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin users page

http://localhost:8000/admin/users

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

17.7 Confirming It Works with Tests

Start everything with Docker Compose

```
docker-compose up
```

Run the test suite

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes test
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

17.8 Coding Exercise Homework Assignments

Please do not distribute the source code publicly because this course's content is only meant for students who purchase the course!

When you're finished with the assignment, just zip up the source code and e-mail it to me at nick.janetakis@gmail.com along with your name on Udemy.

17.8.1 1) Extend the custom admin with bets

- You should be able to click Bets in the nav bar as an admin
- You should be able to see a list of bets by all users
- You should be able to search by a user's e-mail address or username
- In the bet list page, it would be nice to see at least the wagered amounts and if they won or not. Fit as much as you can without crowding it
- You should be able to goto a user's details page and click a shortcut link which leads to a pre-searched result in the list bets page

17.8.2 2) Create the leaderboard

- You should be able to access the leaderboard page and see a searchable paginated ranked list of users based on how many coins they have

You'll need to do quite a few things, but here's the bigger components:

- You'll need a leaderboard model
- You'll likely want a `user_id` field to associate a user to a rank
- You'll likely want a `position` field in the model to keep track of ranks
- You should omit non-subscribed users from the leaderboard because being on the leaderboard is a perk of subscribing
- You'll likely want to adjust the admin user details page so you can adjust a user's rank manually
- You'll need to figure out how often the leaderboard should be updated because after each individual bet might have performance issues

The last one is interesting because you'll need to balance responsiveness and performance. This might be a job for a recurring Celery task, unless you decide to rank everyone in real time as user's coins change.

I recommend starting out with updating it every minute and then try to think through how that might perform with 10, 10,000 or 1 million users.

Also think about it from the user's POV. They want to see their results ASAP.

- A slick UI is important, so you should distinguish the top 10 or top 3 users with a distinct look from the rest

- You should consider adding the top 3 users on the home page

Chapter 18

Processing Microtransactions

18.1 Configuring the App for Purchases

Copy your instance settings from the previous section

Goto the previous section's source code and copy your `instance/settings.py` *file's contents and replace them in this section's* `instance/settings.py` *file.*

18.2 Accepting Payments

Build and start everything with Docker Compose

```
docker-compose up --build
```

Visit the settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

18.3 Viewing the New Invoice History

Start everything with Docker Compose

```
docker-compose up
```

Visit the settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

18.4 Adding to and Modifying the Custom Admin

Start everything with Docker Compose

```
docker-compose up
```

Visit the admin page

http://localhost:8000/admin

E-mail notifications for payments with Stripe

https://dashboard.stripe.com/account/emails

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

18.5 Confirming It Works with Tests

Start everything with Docker Compose

```
docker-compose up
```

Run the test suite

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes test
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 19

Database Migrations

19.1 Going over the requirements.txt Changes

Alembic's documentation

http://alembic.readthedocs.org/en/latest/

Copy your instance settings from the previous section

Goto the previous section's source code and copy your ***`instance/settings.py`*** *file's contents and replace them in this section's* ***`instance/settings.py`*** *file.*

19.2 Creating Our First Migration

Build and start everything with Docker Compose

```
docker-compose up --build
```

Create a new revision

Run this in a second terminal tab.

```
# Docker Machine users can ignore adding the --user flag.
docker-compose exec --user "$(id -u):$(id -g)" website \
  alembic revision -m "create foo table"
```

The `-user "$(id -u):$(id -g)"` flag ensures that the file being created is owned by your workstation's user. Without this flag then it would be owned by root. This only needs to be added for native Docker users.

Revision upgrade/downgrade code snippets

Copy and paste this snippet into the generated revision from above.

```
def upgrade():
    op.create_table('foos',
                    sa.Column('id', sa.Integer, primary_key=True),
                    sa.Column('created_on', AwareDateTime(),
                              default=tzware_datetime),
                    sa.Column('updated_on', AwareDateTime(),
                              default=tzware_datetime,
                              onupdate=tzware_datetime),
                    sa.Column('bar', sa.String(128), index=True))


def downgrade():
    op.drop_table('foos')
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

19.3 Sidetracking with pgAdmin III

Start everything with Docker Compose

```
docker-compose up
```

Download pgAdmin III

OSX http://www.pgadmin.org/download/macosx.php

Windows http://www.pgadmin.org/download/windows.php

Linux / Ubuntu https://wiki.postgresql.org/wiki/Apt
Follow everything in the Quickstart except for installing `postgresql-9.5`.

Linux / Other http://www.pgadmin.org/download/source.php

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

19.4 Running Our First Migration

Start everything with Docker Compose

```
docker-compose up
```

Move forward by performing an upgrade

Run this in a second terminal tab.

```
docker-compose exec website alembic upgrade head
```

Roll back 1 revision by downgrading

Run this in a second terminal tab.

```
docker-compose exec website alembic downgrade -1
```

Move forward again by performing an upgrade

Run this in a second terminal tab.

```
docker-compose exec website alembic upgrade head
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

19.5 Creating and Running a Second Migration

Start everything with Docker Compose

```
docker-compose up
```

Create a new revision

Run this in a second terminal tab.

```
# Docker Machine users can ignore adding the --user flag.
docker-compose exec --user "$(id -u):$(id -g)" website \
  alembic revision -m "add column to foos"
```

Revision upgrade/downgrade code snippets

Copy and paste this snippet into the generated revision from above.

```
def upgrade():
    op.add_column('foos', sa.Column('hello_on', AwareDateTime(),
                                    default=tzware_datetime))


def downgrade():
    op.drop_column('foos', 'hello_on')
```

Move forward again by performing an upgrade

Run this in a second terminal tab.

```
docker-compose exec website alembic upgrade head
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

19.6 Viewing the History of Your Migrations

Start everything with Docker Compose

```
docker-compose up
```

View the current status of our migrations

Run this in a second terminal tab.

```
docker-compose exec website alembic current
```

View the migration history

Run this in a second terminal tab.

```
docker-compose exec website alembic history --verbose
```

View the migration history's help menu

Run this in a second terminal tab.

```
docker-compose exec website alembic history --help
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

19.7 Auto-Generating Migration Scripts

Start everything with Docker Compose

```
docker-compose up
```

Roll back 2 revisions by downgrading

Run this in a second terminal tab.

```
docker-compose exec website alembic downgrade -2
```

If you experimented on your own and you cannot downgrade 2 levels, that's ok. Just make sure your database is at a state where `foos` does not exist.

Modify the User model

Copy and paste this line into your User model under line 51.

```
foobar = db.Column(db.String(128), index=True)
```

Auto-generate a migration script

Run this in a second terminal tab.

```
docker-compose exec --user "$(id -u):$(id -g)" website \
  alembic revision --autogenerate -m "add column to users"
```

Auto-generate's documentation

http://alembic.readthedocs.org/en/latest/autogenerate.html#what-does-autogenerate-detect-and-what-does-it-not-detect

Move forward again by performing an upgrade

Run this in a second terminal tab.

```
docker-compose exec website alembic upgrade head
```

Downgrade the migration

Run this in a second terminal tab.

```
# Remove the foobar field from the User model and then run:
docker-compose exec website alembic downgrade -1
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 20

Internationalization (i18n)

20.1 Configuring the App to Handle i18n

Copy your instance settings from the previous section

Goto the previous section's source code and copy your ***`instance/settings.py`*** *file's contents and replace them in this section's* ***`instance/settings.py`*** *file.*

20.2 Updating the User Blueprint to Support i18n

Build and start everything with Docker Compose

```
docker-compose up --build
```

Reset and seed the database

The database schema changed since the last section, so we need to do this.

```
docker-compose exec website snakeeyes db reset --with-testdb
docker-compose exec website snakeeyes add all
```

Visit the settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

20.3 Updating the Billing Blueprint to Support i18n

Start everything with Docker Compose

```
docker-compose up
```

Visit the settings page

http://localhost:8000/settings

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

20.4 Generating the Primary messages.pot File

Start everything with Docker Compose

```
docker-compose up
```

Extract the translations to the `messages.pot` file

Run this in a second terminal tab.

```
# Docker Machine users can ignore adding the --user flag.
docker-compose exec --user "$(id -u):$(id -g)" website snakeeyes babel extract
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

20.5 Updating Translations for Multiple Languages

Start everything with Docker Compose

```
docker-compose up
```

Update multiple languages to sync them to the new `messages.pot` file

Run this in a second terminal tab.

```
# Docker Machine users can ignore adding the --user flag.
docker-compose exec --user "$(id -u):$(id -g)" website snakeeyes babel update
```

Compile the translations

Run this in a second terminal tab.

```
# Docker Machine users can ignore adding the --user flag.
docker-compose exec --user "$(id -u):$(id -g)" website snakeeyes babel compile
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

20.6 Adding Additional Languages

Start everything with Docker Compose

```
docker-compose up
```

Initialize the Spanish language

Run this in a second terminal tab.

```
# Docker Machine users can ignore adding the --user flag.
docker-compose exec --user "$(id -u):$(id -g)" website snakeeyes \
  babel init --language es
```

Compile the translations

Run this in a second terminal tab.

```
# Docker Machine users can ignore adding the --user flag.
docker-compose exec --user "$(id -u):$(id -g)" website snakeeyes babel compile
```

Visit the settings page

http://localhost:8000/settings

Popular work flow for keeping a few languages up to date

- Extract (create a new primary messages.pot file)
- Update (sync the messages.pot file for each language)
- Compile (transform the translations into a format that babel understands)

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

20.7 Confirming It Works with Tests

Start everything with Docker Compose

```
docker-compose up
```

Run the test suite

Run this in a second terminal tab.

```
docker-compose exec website snakeeyes test
```

Stop Docker Compose

```
# Press CTRL+C a few times and ensure everything has stopped by running:
docker-compose stop
```

Clean up our Docker mess

```
# Remove stopped containers
docker-compose rm -f

# Remove "dangling" images
docker rmi -f $(docker images -qf dangling=true)
```

Chapter 21

Where to Go Next?

21.1 Congrats on Finishing This Course

Share this course

http://nickjanetakis.com/courses/build-a-saas-app-with-flask

Example Tweet if you want to show some love

Please double check that Twitter doesn't break the URL due to PDF formatting.

```
Just finished Build a #SAAS App with #Flask by @nickjanetakis, it was awesome!
Take a look at it here http://bit.ly/bsawfv2
```

Stay updated

http://nickjanetakis.com/

Follow me on Twitter

https://twitter.com/nickjanetakis

21.2 Deploying Your App to Production

Additional courses to take

Docker for DevOps https://www.udemy.com/the-docker-for-devops-course-from-development-to-production/?couponCode=BSAWF_20

Scaling Docker on AWS https://www.udemy.com/scaling-docker-on-aws/?couponCode=

Creating an asset pipeline with Flask-Webpack

https://github.com/nickjj/flask-webpack

In a previous section I mentioned I would include a resource to optimize your front-end assets when it comes time to deploying your app to production.

The above Flask extension will help you do that.