Project:
Operationalize a Machine Learning...

SEARCH

RESOURCES

CONCEPTS

- ✓ 1. Introduction
- ✓ 2. Project Overview
- ✓ 3. Project Structure & Files
- ✓ **4. Create the Project Environment**
- ✓ 5. Detailed Project Tasks
- 6. Project: Operationalize a Machine ...

Project:
Operationalize a Machine Learning...

SEARCH

RESOURCES

CONCEPTS

- ✓ 1. Introduction
- ✓ 2. Project Overview
- ✓ 3. Project Structure & Files
- ✓ **4. Create the Project Environment**
- ✓ 5. Detailed Project Tasks
- 6. Project: Operationalize a Machine ...

Mentor Help

Project:
Operationalize a Machine Learning...

SEARCH

RESOURCES

CONCEPTS

- ✓ 1. Introduction
- ✓ 2. Project Overview
- ✓ 3. Project Structure & Files
- ✓ **4. Create the Project Environment**
- ✓ 5. Detailed Project Tasks
- 6. Project: Operationalize a Machine ...

Mentor Help

Project:
Operationalize a Machine Learning...

SEARCH

RESOURCES

CONCEPTS

- ✓ 1. Introduction
- ✓ 2. Project Overview
- ✓ 3. Project Structure & Files
- ✓ **4. Create the Project Environment**
- ✓ 5. Detailed Project Tasks
- 6. Project: Operationalize a Machine ...

Project:
Operationalize a Machine Learning...

SEARCH

RESOURCES

CONCEPTS

- ✓ 1. Introduction
- ✓ 2. Project Overview
- ✓ 3. Project Structure & Files
- ✓ **4. Create the Project Environment**
- ✓ 5. Detailed Project Tasks
- 6. Project: Operationalize a Machine ...

Mentor Help
Ask a mentor on our Q&A platform

Create and activate an environment

All of these instructions are to be completed via a terminal/command line prompt.

1. Create and Activate an Environment

Git and version control

These instructions also assume you have `git` installed for working with Github from a terminal window, but if you do not, you can download that first from this Github installation page.

Now, you're ready to create your local environment!

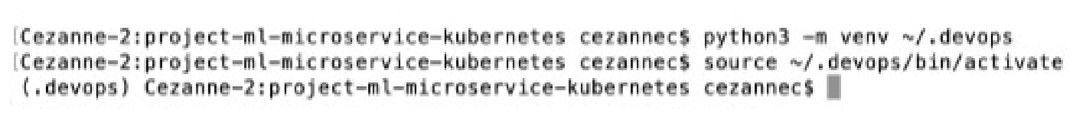
1. If you haven't already done so, clone the project repository, and navigate to the project folder.

```
git clone https://github.com/udacity/DevOps_Microservices.git
cd DevOps_Microservices/project-ml-microservice-kubernetes
```

2. Create (and activate) a new environment, named `.devops` with Python 3. If prompted to proceed with the install `(Proceed [y]/n)` type y.

```
python3 -m venv ~/.devops
source ~/.devops/bin/activate
```

At this point your command line should look something like: `(.devops) <User>:project-ml-microservice-kubernetes<user>$`. The `(.devops)` indicates that your environment has been activated, and you can proceed with further package installations.



```
Cezanne-2:project-ml-microservice-kubernetes cezannec$ python3 -m venv ~/.devops
Cezanne-2:project-ml-microservice-kubernetes cezannec$ source ~/.devops/bin/activate
(.devops) Cezanne-2:project-ml-microservice-kubernetes cezannec$
```

3. Installing dependencies via project `Makefile`. Many of the project dependencies are listed in the file `requirements.txt`; these can be installed using `pip` commands in the provided `Makefile`. While in your project directory, type the following command to install these dependencies.

```
make install
```

Now most of the `.devops` libraries are available to you. There are a couple of other libraries that we'll be using, which can be downloaded as specified, below.

Other Libraries

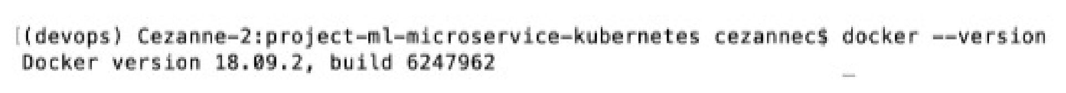
While you still have your `.devops` environment activated, you will still need to install:

- Docker
- Hadolint
- Kubernetes (Minikube)

Docker

You will need to use Docker to build and upload a containerized application. If you already have this installed and created a docker account, you may skip this step.

1. You'll need to create a free docker account, where you'll choose a unique username and link your email to a docker account. **Your username is your unique docker ID.**
2. To install the latest version of docker, choose the Community Edition (CE) for your operating system, on docker's installation site. It is also recommended that you install the latest, **stable** release:
3. After installation, you can verify that you've successfully installed docker by printing its version in your terminal: `docker --version`



```
(devops) Cezanne-2:project-ml-microservice-kubernetes cezannec$ docker --version
Docker version 18.09.2, build 6247962
```

Run Lint Checks

This project also must pass two lint checks; `hadolint` checks the Dockerfile for errors and `pylint` checks the `app.py` source code for errors.

1. Install `hadolint` following the instructions, on hadolint's page:

For Mac:

```
brew install hadolint
```

For Windows:

```
scoop install hadolint
```

2. In your terminal, type: `make lint` to run lint checks on the project code. If you haven't changed any code, all requirements should be satisfied, and you should see a printed statement that rates your code (and prints out any additional comments):

```
------------------------------------
Your code has been rated at 10.00/10
```

Install Minikube

To run a Kubernetes cluster locally, for testing and project purposes, you need the Kubernetes package, Minikube. This operates in a virtual machine and so you'll need to download two things: A virtual machine (aka a hypervisor) then minikube. Thorough installation instructions can be found here. Here is how I installed minikube:

1. Install VirtualBox:

For Mac:

```
brew cask install virtualbox
```

For Windows, I recommend using a Windows host.

2. Install minikube:

For Mac:

```
brew cask install minikube
```

For Windows, I recommend using the Windows installer.

That's it!

Setting up an environment is an important part of development. Now you are ready to start working on the project files to containerize a machine learning application!

You should return to this page if you need to troubleshoot any dependency issues.

Troubleshooting

- In general, you can verify installation by checking the version of a library, ex. `kubectl version` or `docker --version`. If there is no package found, you may need to install that library.
- **Mac issue**: If you get an error `error: invalid active developer path`, that means you need to install some Xcode developer tools. You can do this on Mac by running this terminal command: `xcode-select --install`