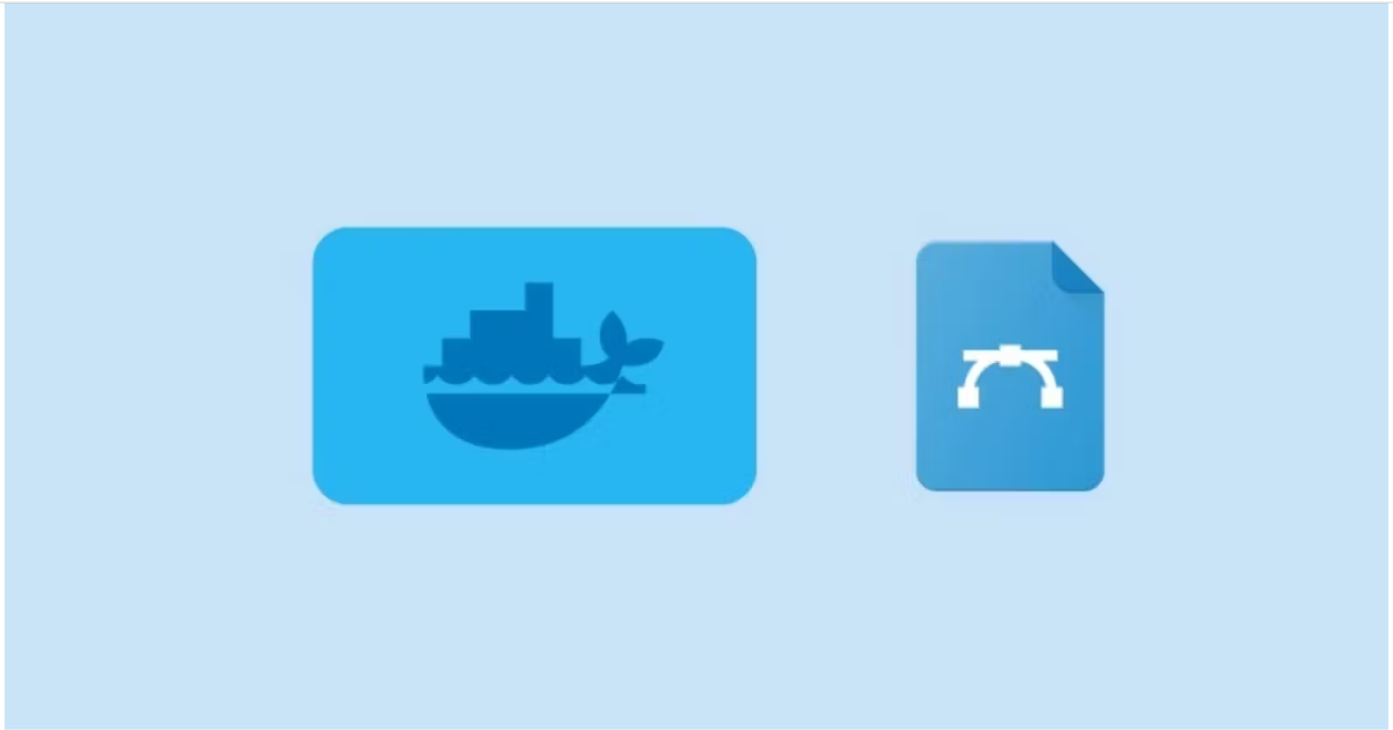


# Dockerfile in Docker for DevOps

## Day 17 : 90Days of DevOps Challenge

SS Shivraj Salunkhe · Apr 26, 2023 · 2 min read

TABLE OF CONTENTS

> **Task :**

A Dockerfile is a text file that contains instructions for building a Docker image. The Dockerfile defines the environment in which an application runs, including the operating system, libraries, and dependencies. Here are some key points to know about Dockerfiles:

1. **Syntax**: Dockerfiles use a specific syntax for defining instructions, which include keywords such as `FROM`, `RUN`, `COPY`, and `CMD`. These keywords are used to specify the base image, install packages, copy files, and set the default command, among other things.
2. **Layering**: Dockerfiles use a layering system to optimize the build process. Each instruction in a Dockerfile creates a new layer, which can be cached and reused in subsequent builds. This makes building Docker images fast and efficient.
3. **Reproducibility**: Dockerfiles make it easy to reproduce a specific environment by defining the exact set of instructions required to build an image. This ensures that the same image can be built consistently across different environments.
4. **Versioning**: Dockerfiles can be version-controlled using a tool such as Git. This makes it easy to track changes to the Dockerfile over time and revert to previous versions if necessary.
5. **Best Practices**: There are best practices for writing Dockerfiles, such as minimizing the number of layers, cleaning up after each command, and using specific versions of packages. Following these best practices can result in smaller, more efficient images.

These are some of the key points to know about Dockerfiles. With their focus on syntax, layering, reproducibility, versioning, and best practices, Dockerfiles are an essential tool for building and managing Docker images.

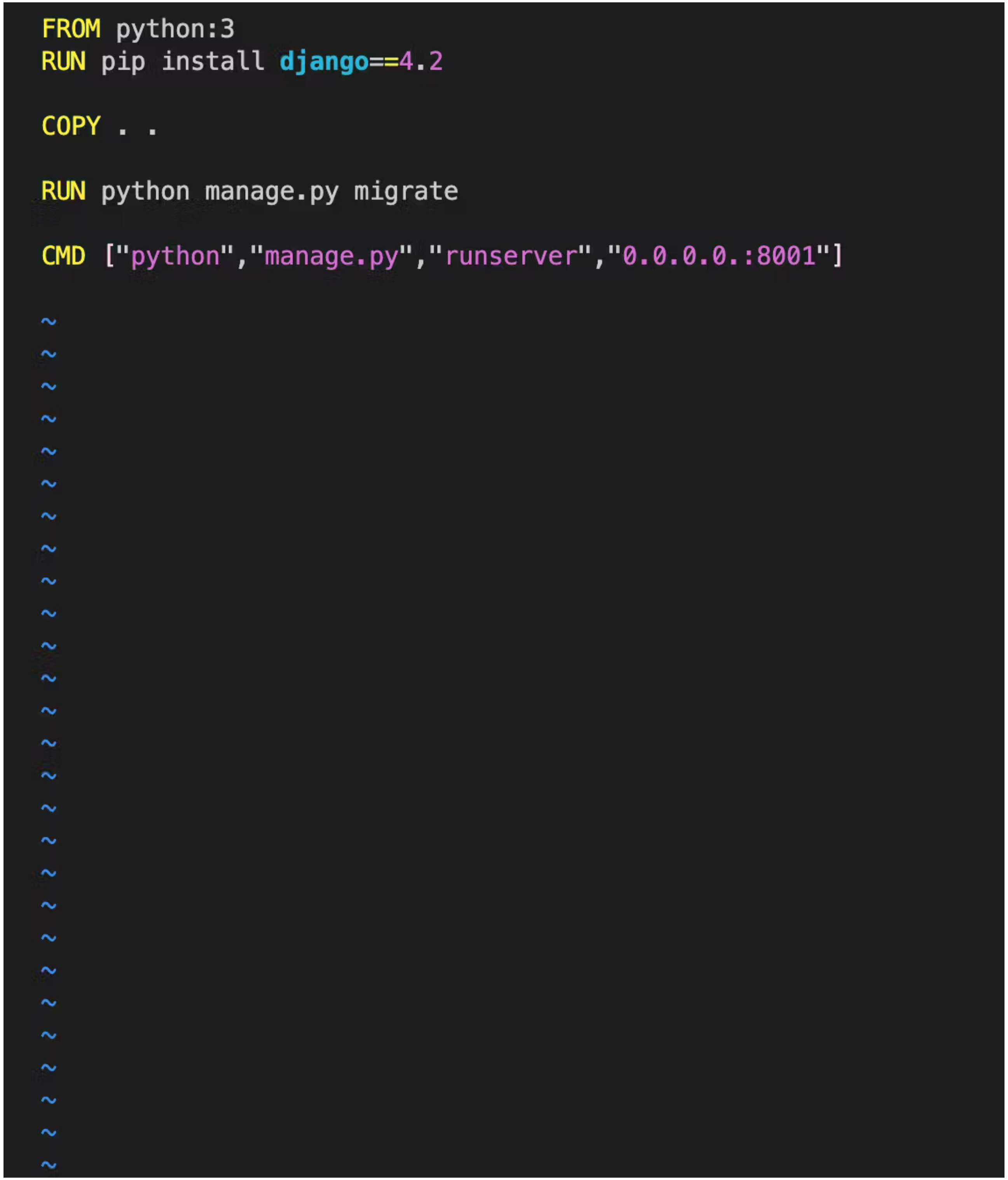
# Task :

**-> Create a Dockerfile for a simple web application (e.g. a Node.js or Python app)**

-- Making new Docker file named "Dockerfile" for simple web application

```
vi Dockerfile
```

-- writing commands in the Dockerfile to automate the process.

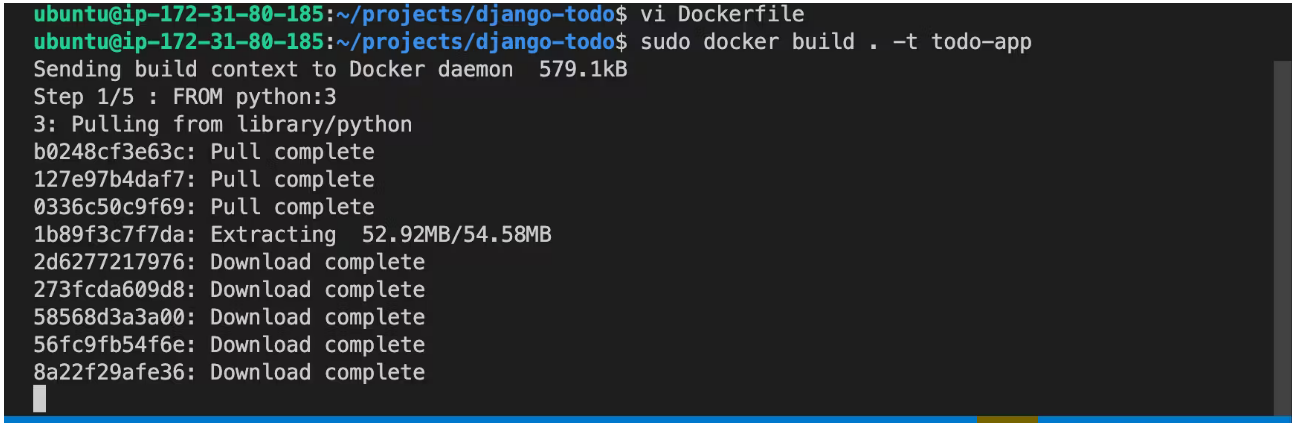


```
FROM python:3
RUN pip install django==4.2

COPY . .

RUN python manage.py migrate

CMD ["python","manage.py","runserver","0.0.0.0.:8001"]
```

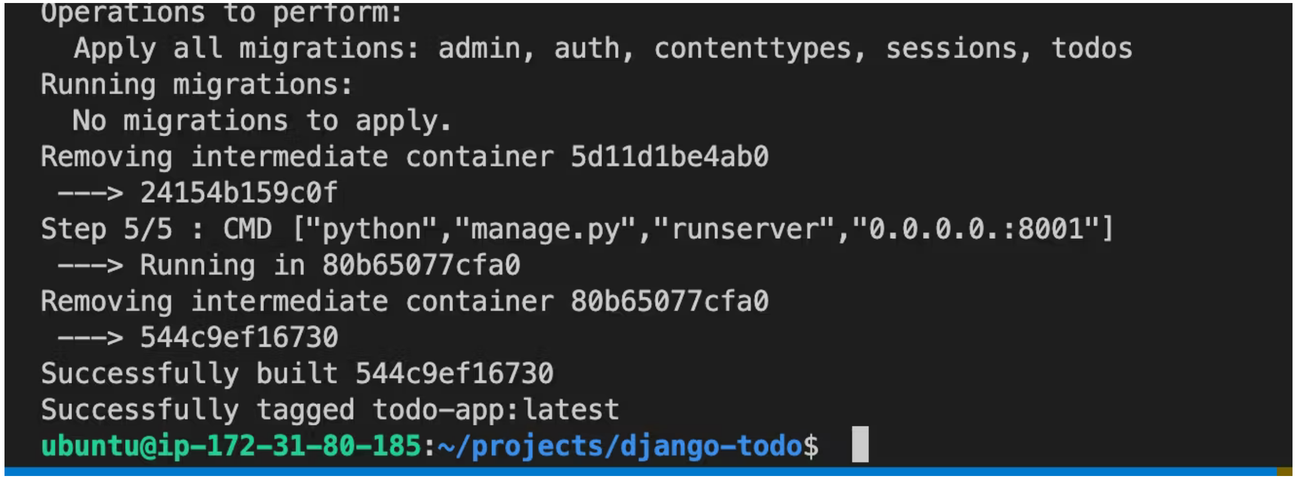
```
~
~
~
:wq
```

## -> Build the image using the Dockerfile and run the container

```
sudo docker build . -t todo-app
```



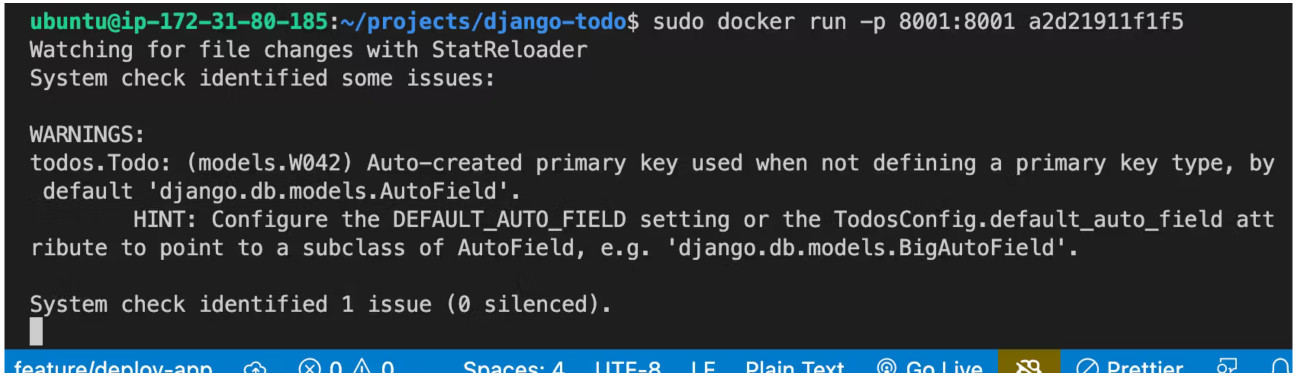
```
ubuntu@ip-172-31-80-185:~/projects/django-todo$ vi Dockerfile
ubuntu@ip-172-31-80-185:~/projects/django-todo$ sudo docker build . -t todo-app
Sending build context to Docker daemon  579.1kB
Step 1/5 : FROM python:3
3: Pulling from library/python
b0248cf3e63c: Pull complete
127e97b4daf7: Pull complete
0336c50c9f69: Pull complete
1b89f3c7f7da: Extracting  52.92MB/54.58MB
2d6277217976: Download complete
273fcda609d8: Download complete
58568d3a3a00: Download complete
56fc9fb54f6e: Download complete
8a22f29afe36: Download complete
```



```
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions, todos
Running migrations:
  No migrations to apply.
Removing intermediate container 5d11d1be4ab0
 ---> 24154b159c0f
Step 5/5 : CMD ["python","manage.py","runserver","0.0.0.0.:8001"]
 ---> Running in 80b65077cfa0
Removing intermediate container 80b65077cfa0
 ---> 544c9ef16730
Successfully built 544c9ef16730
Successfully tagged todo-app:latest
ubuntu@ip-172-31-80-185:~/projects/django-todo$
```

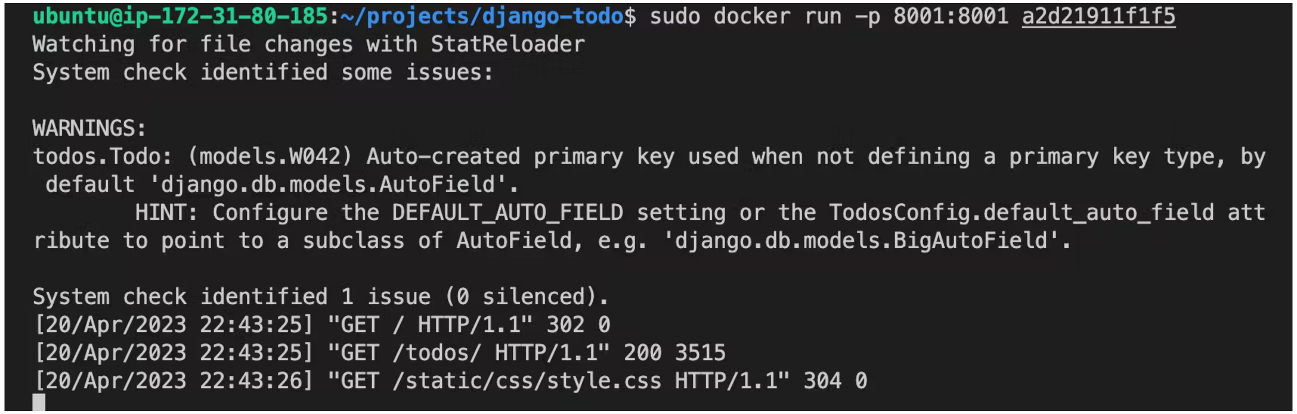
-> To Run the Image and Making the Container.

```
sudo docker run -p 8001:8001 (your_docker_image_id)
```

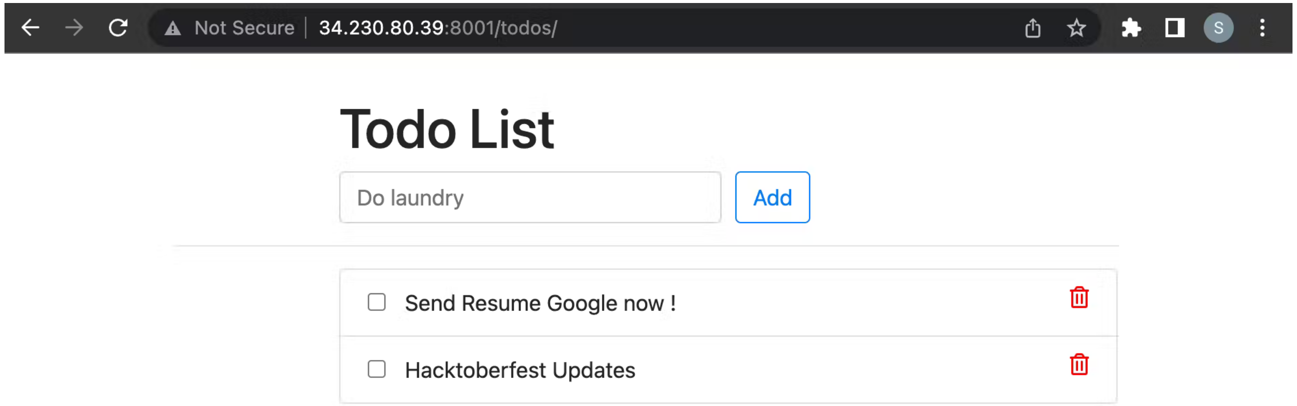


```
ubuntu@ip-172-31-80-185:~/projects/django-todo$ sudo docker run -p 8001:8001 a2d21911f1f5
Watching for file changes with StatReloader
System check identified some issues:

WARNINGS:
todos.Todo: (models.W042) Auto-created primary key used when not defining a primary key type, by
 default 'django.db.models.AutoField'.
        HINT: Configure the DEFAULT_AUTO_FIELD setting or the TodosConfig.default_auto_field att
ribute to point to a subclass of AutoField, e.g. 'django.db.models.BigAutoField'.

System check identified 1 issue (0 silenced).
```

-> Verify that the application is working as expected by accessing it in a web browser.



```
ubuntu@ip-172-31-80-185:~/projects/django-todo$ sudo docker run -p 8001:8001 a2d21911f1f5
Watching for file changes with StatReloader
System check identified some issues:

WARNINGS:
todos.Todo: (models.W042) Auto-created primary key used when not defining a primary key type, by
 default 'django.db.models.AutoField'.
        HINT: Configure the DEFAULT_AUTO_FIELD setting or the TodosConfig.default_auto_field att
ribute to point to a subclass of AutoField, e.g. 'django.db.models.BigAutoField'.

System check identified 1 issue (0 silenced).
[20/Apr/2023 22:43:25] "GET / HTTP/1.1" 302 0
[20/Apr/2023 22:43:25] "GET /todos/ HTTP/1.1" 200 3515
[20/Apr/2023 22:43:26] "GET /static/css/style.css HTTP/1.1" 304 0
```



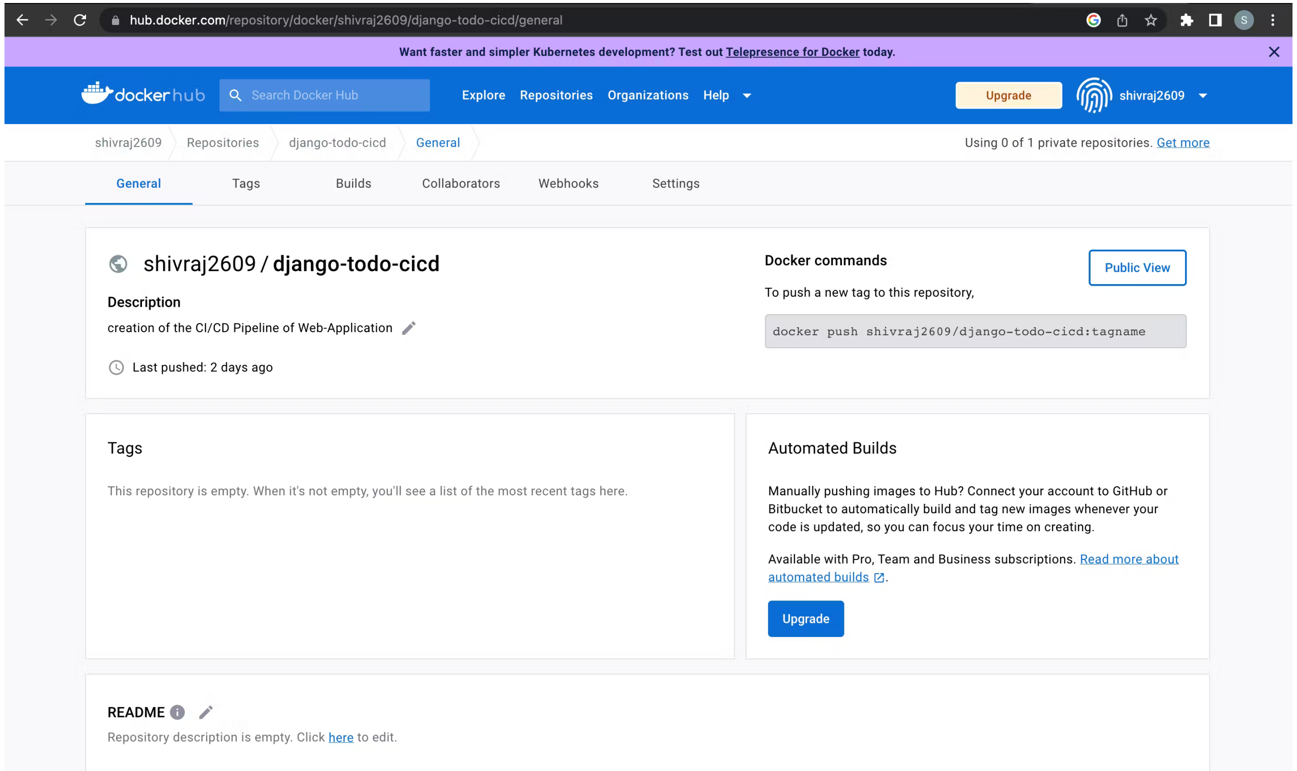
Todo List

Do laundry

□ Send Resume Google now !

□ Hacktoberfest Updates

-> Push the image to a public or private repository (e.g. Docker Hub )



hub.docker.com/repository/docker/shivraj2609/django-todo-cicd/general

Want faster and simpler Kubernetes development? Test out Telepresence for Docker today.

shivraj2609 › Repositories › django-todo-cicd › General

Using 0 of 1 private repositories. Get more

shivraj2609 / django-todo-cicd

Description

creation of the CI/CD Pipeline of Web-Application

Last pushed: 2 days ago

Docker commands

To push a new tag to this repository,

```
docker push shivraj2609/django-todo-cicd:tagname
```

Tags

This repository is empty. When it's not empty, you'll see a list of the most recent tags here.

Automated Builds

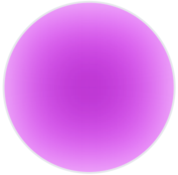
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. Read more about automated builds.

README

Repository description is empty. Click here to edit.

Docker Devops containers Linux Dockerfile



WRITTEN BY

## Shivraj Salunkhe

Follow

Dedicated and hardworking undergraduate student pursuing a degree in Information Technology with a passion for learning, leadership experience, and real-world skills through internships and part-time jobs in IT sector. Actively looking for new Opportunities and committed to do personal and professional growth.

MORE ARTICLES

**Shivraj Salunkhe**

**Shivraj Salunkhe**

**Shivraj Salunkhe**

### Python Libraries for DevOps

-> Reading JSON in Python To read JSON files in Python, you can use the json module. import json # ...

### Docker for DevOps Engineers

Docker is a popular tool used by DevOps Engineers for containerization and managing application depl...

### Create CI/CD Pipeline of a web-app using Github, Jenkins, AWS and Docker.

This Blog helps you to create the CI/CD Pipeline of Web-Application named "Todo-List" to your Github...