NAME – KASHISH

ROLL NUMBER – 102117150

TIET, PATIALA

RAPIDFORT ASSESSMENT

Offline Project

- Create a web application that processes Word documents (.docx) and converts them to PDF.
- The user should be able to upload a doc file, view file metadata, and download the converted PDF.
- Bonus points for providing us with a hosted endpoint where we can run tests, adding password protection support for PDF, and implementing microservice architecture.
- **Rubric for program**
    - Create a repo and store your program on GitHub or Bitbucket.
    - Documentation.
    - Exception handlers.
    - Add a simple UI to this web application.
    - Dockerize the application.
    - Add a GitHub actions or equivalent pipeline to build its docker image.
    - Create a bash script with instructions to run the container.
    - Create Kubernetes manifest files to host the web server.

DOCX to PDF Conversion Project

1. **Problem Statement**

To create a web application that allows users to:

- Upload .docx files for processing.
- Convert .docx files to PDF.
- View file metadata (e.g., size, author, creation date).
- Download the converted PDF file.

Additional features include:

- Password protection for PDF files.
- Microservices-based architecture for scalability.
- Hosting the solution with containerization and orchestration.

2. **Objectives**

- Develop a user-friendly web interface for document processing.
- Ensure smooth file uploads and downloads.
- Use Docker to containerize the application for portability.
- Implement a CI/CD pipeline using GitHub Actions for automatic Docker builds.
- Use Kubernetes for deployment and scaling.

3. **Technologies and Tools Used**

- **Programming Language:** Python
- **Framework:** Flask (for backend web application)
- **Frontend Tools:** HTML, CSS
- **Containerization:** Docker
- **CI/CD Pipeline:** GitHub Actions
- **Orchestration:** Kubernetes
- **Version Control:** Git and GitHub

4. **Steps to Build the Project**

Step 1: Setting Up the Python Application

- A Python Flask application was developed to:
  - Handle file uploads via HTTP requests.
  - Convert .docx files to PDF using the python-docx and reportlab libraries.
  - Display file metadata.

Command: python app.py

Step 2: Creating a Docker Image

- A Dockerfile was created to containerize the Python application.
  The Dockerfile included:
  - A base image (python:3.9-slim).
  - Dependencies installed via requirements.txt.
  - The Flask app running inside the container.

Commands: docker build -t docx-to-pdf .
docker run -p 5000:5000 docx-to-pdf

Step 3: Running the Application on Kubernetes

- Created Kubernetes manifest files (deployment.yml and service.yml) for deploying the application.
  - **Deployment**: Defined the pod template and container image.
  - **Service**: Exposed the application to external traffic.

Commands:

kubectl apply -f deployment.yml

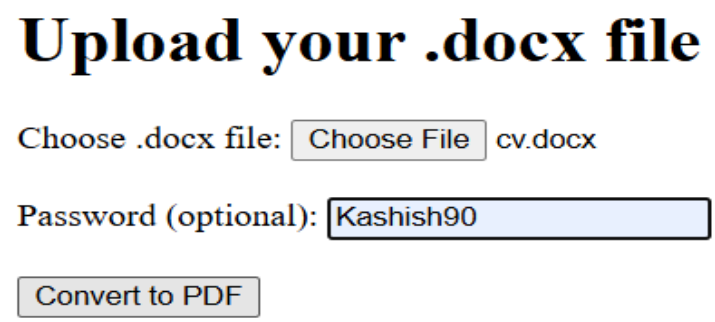
kubectl apply -f service.yml

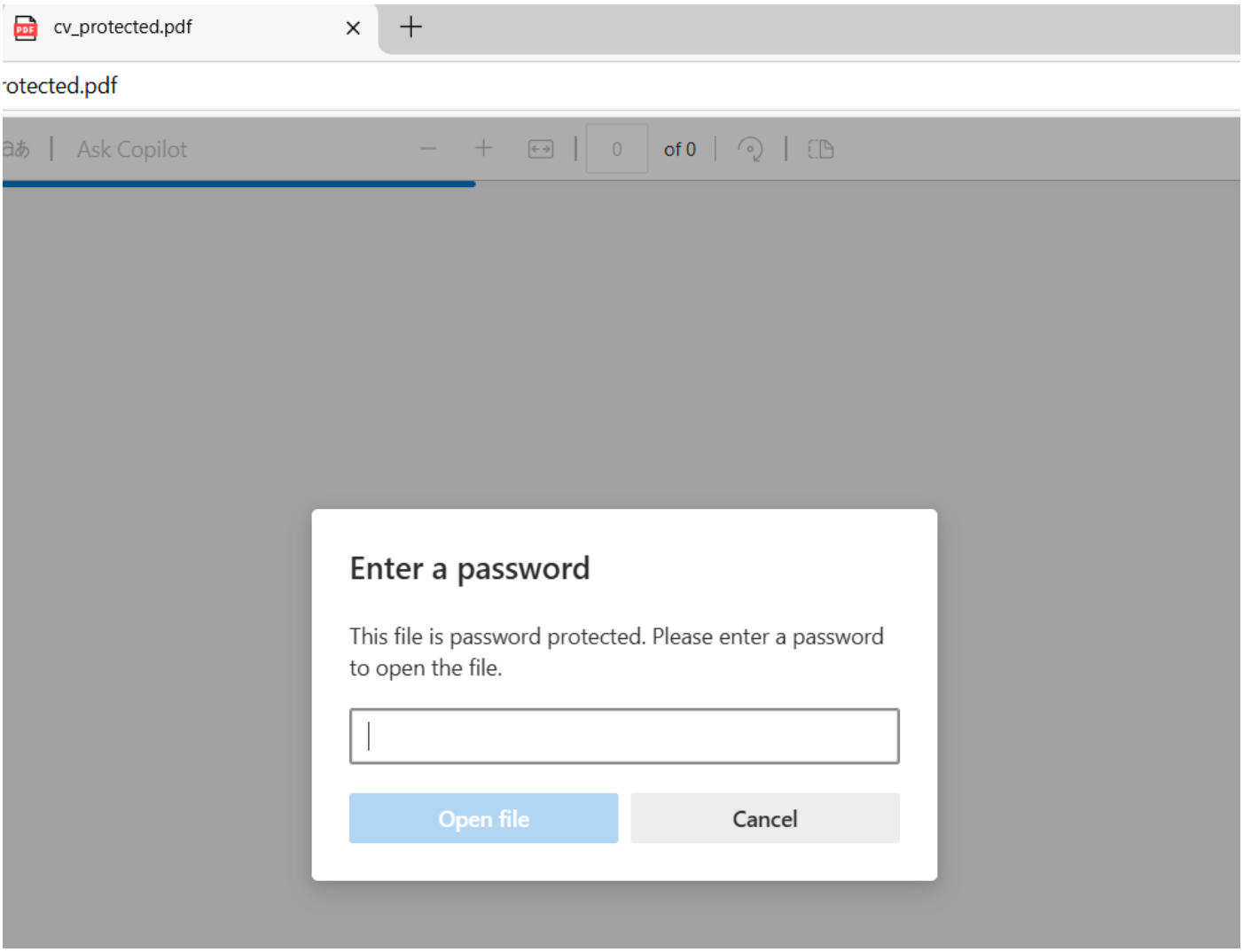
kubectl get pods # Check running pods

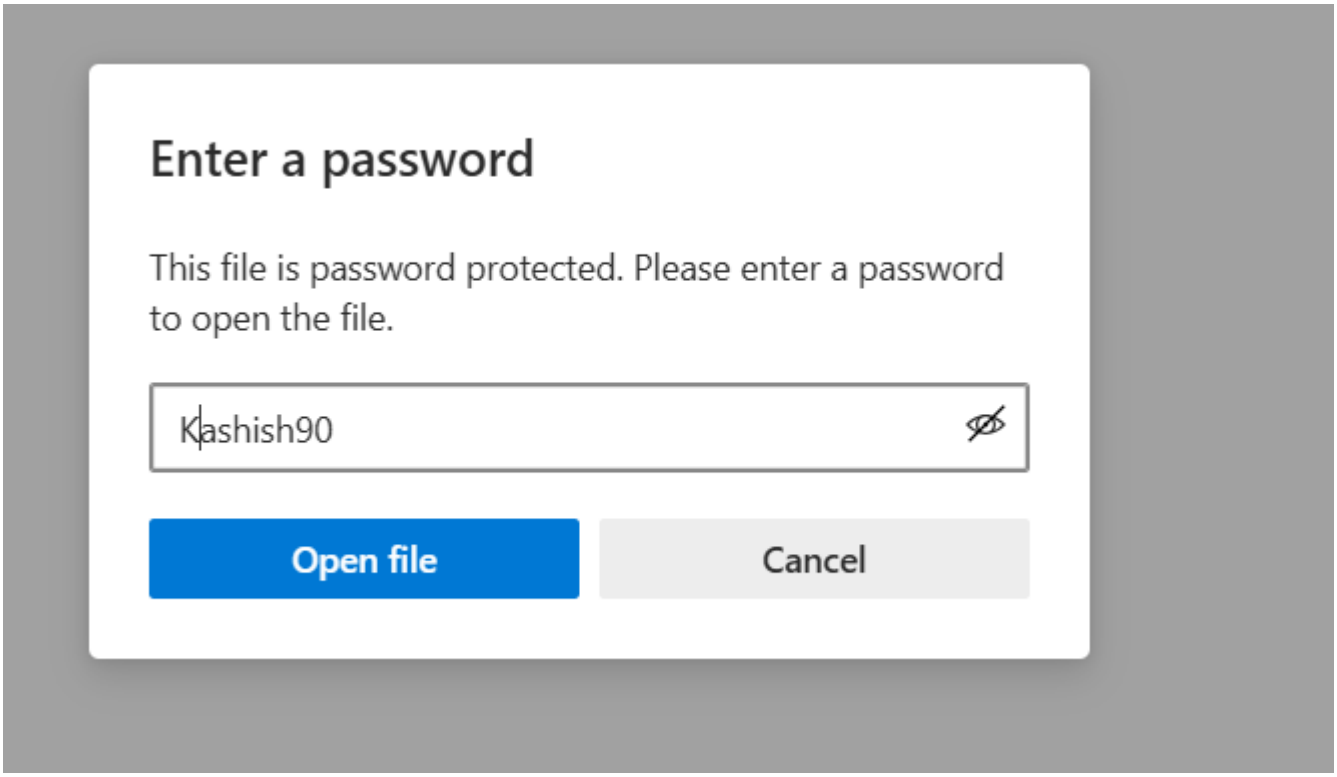
kubectl port-forward service/docx-to-pdf-service 8080:80

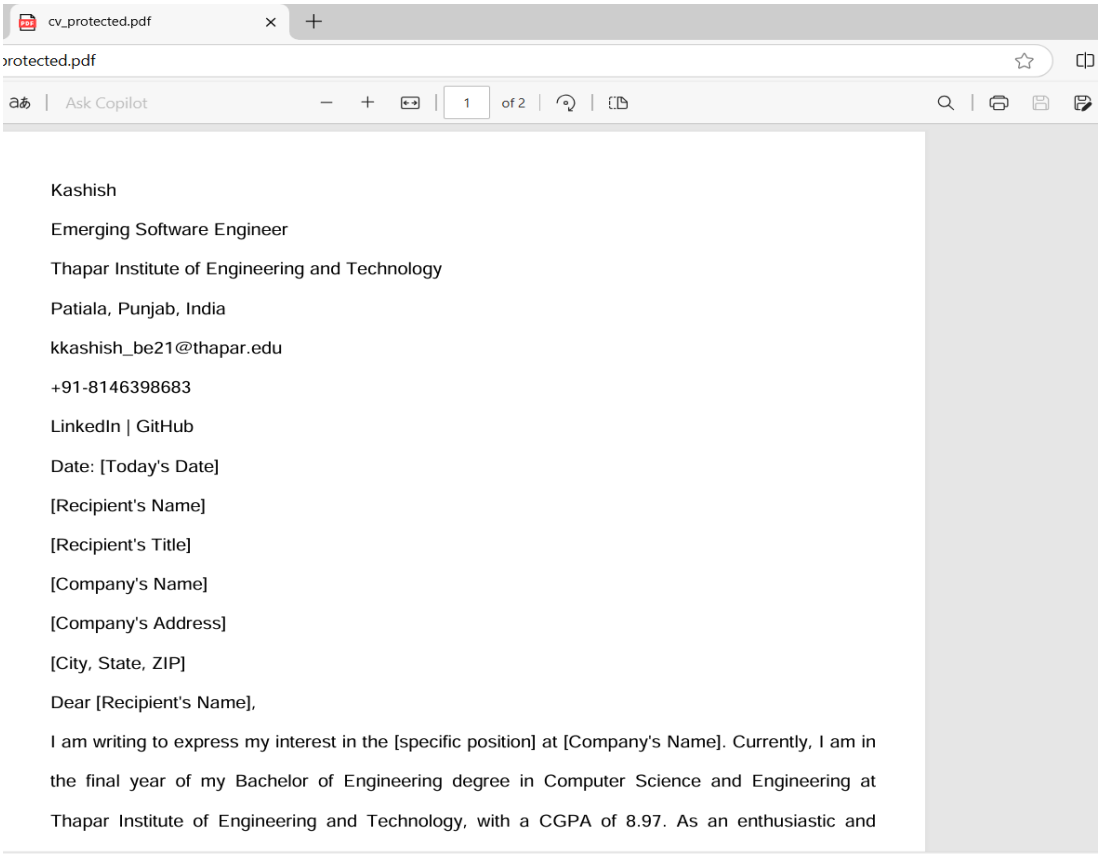
Website Page:

Working of Website:









cv_protected.pdf

protected.pdf

Ask Copilot 1 of 2

Kashish

Emerging Software Engineer

Thapar Institute of Engineering and Technology

Patiala, Punjab, India

kkashish_be21@thapar.edu

+91-8146398683

LinkedIn | GitHub

Date: [Today's Date]

[Recipient's Name]

[Recipient's Title]

[Company's Name]

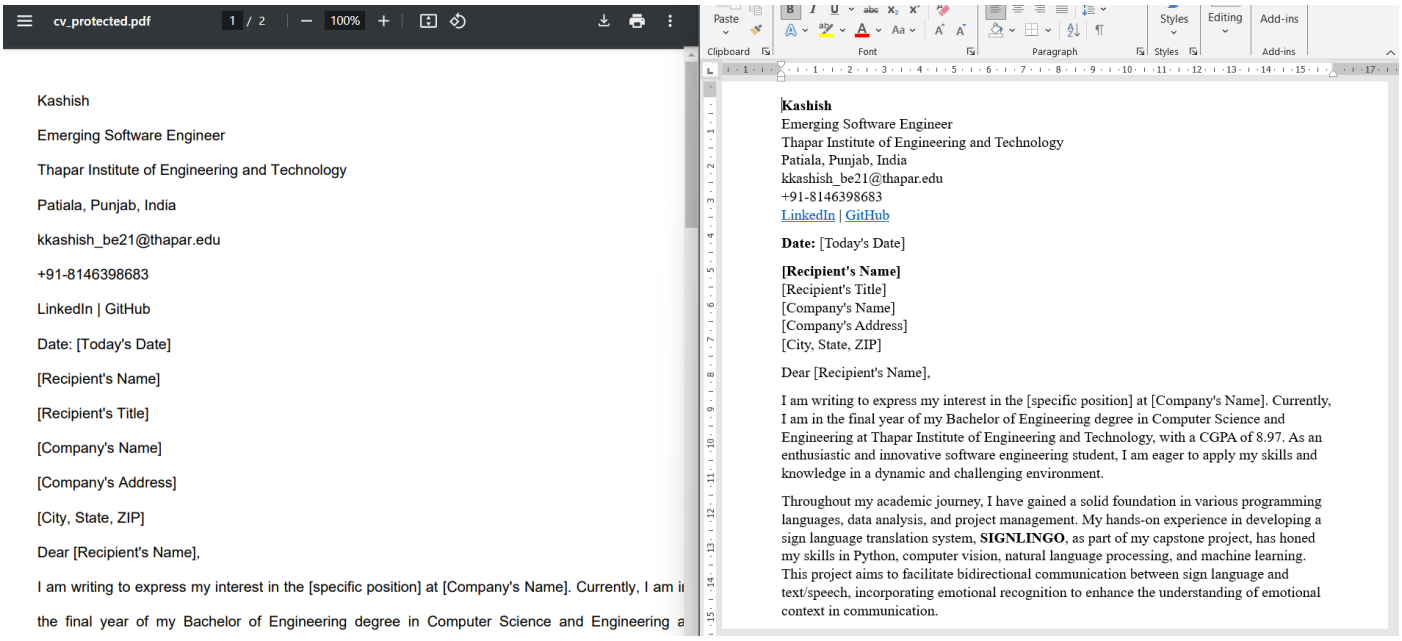
[Company's Address]

[City, State, ZIP]

Dear [Recipient's Name],

I am writing to express my interest in the [specific position] at [Company's Name]. Currently, I am in the final year of my Bachelor of Engineering degree in Computer Science and Engineering at Thapar Institute of Engineering and Technology, with a CGPA of 8.97. As an enthusiastic and

Compare docs and pdf file:



Github link: https://github.com/Kashishsingla111/docxtopdf.git

Step 4: Setting Up GitHub Actions for CI/CD

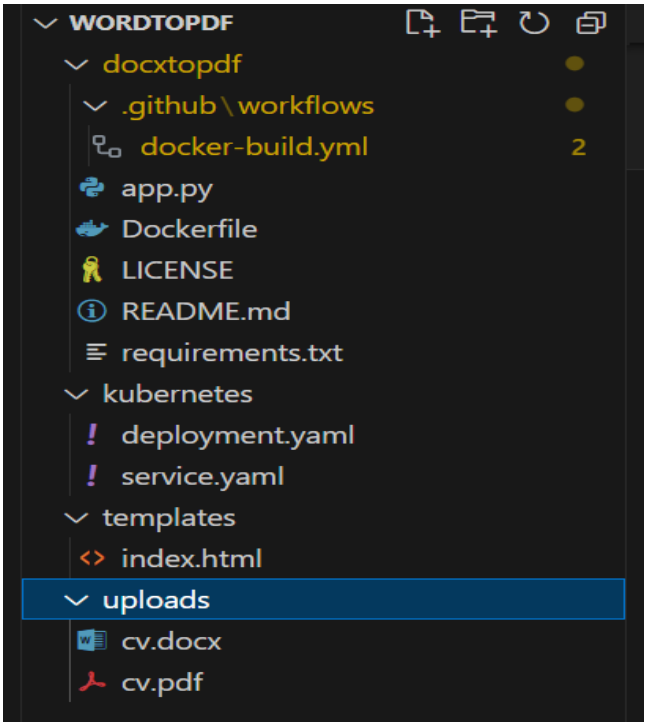
Github Action Workflow:

After Kubernetes, when you need to add github workflow action, follow the steps:

Create a new github repository, then clone it to your local machine, move the necessary files.

```
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf> git clone https://github.com/Kashishsingla111/docxtopdf
Cloning into 'docxtopdf'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (4/4), done.
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf> cd docxtopdf
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf\docxtopdf> git init
Reinitialized existing Git repository in C:/Users/hp/OneDrive/Desktop/wordtopdf/docxtopdf/.git/
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf\docxtopdf> mkdir -p .github/workflows
>>
```

Directory will look like this:



Go to github repo, **Settings** > **Secrets and variables** > **Actions** > **New repository secret**.

Add the following secrets:

- DOCKER_USERNAME: Your Docker Hub username.
- DOCKER_PASSWORD: Your Docker Hub password or access token.

Then in VS Code:

Inside docker-build.yml, enter the following code:

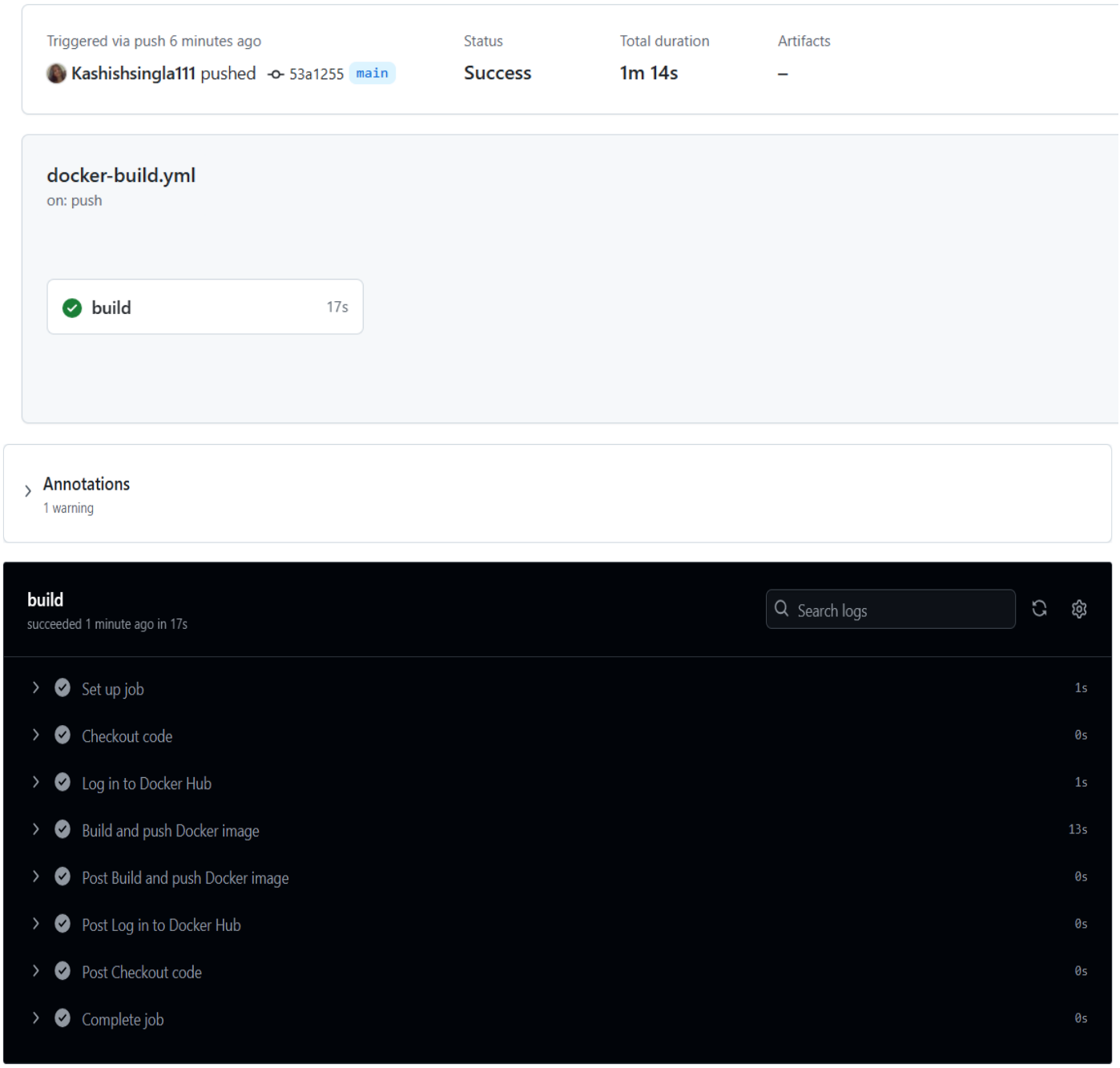
```yaml
name: Docker Build and Push

on:
  push:
    branches:
      - main

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v3

    - name: Log in to Docker Hub
      uses: docker/login-action@v2
      with:
        username: ${{ secrets.DOCKER_USERNAME }}
        password: ${{ secrets.DOCKER_PASSWORD }}

    - name: Build and push Docker image
      uses: docker/build-push-action@v4
      with:
```

```
      push: true
      tags: kashishsingla50/docxtopdfwf:latest
```

After that, commit and push files:

```
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf\docxtopdf> git add .
>>
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf\docxtopdf> git commit -m "Add GitHub Actions workflow for Dock
er"
>>
[main 53a1255] Add GitHub Actions workflow for Docker
 4 files changed, 109 insertions(+)
 create mode 100644 .github/workflows/docker-build.yml
 create mode 100644 Dockerfile
 create mode 100644 app.py
 create mode 100644 requirements.txt
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf\docxtopdf> git push origin main
>>
```

We can see this Workflow Action in Github under Actions tab:



Set up job

1s

```
1  Current runner version: '2.320.0'
2  ▶ Operating System
6  ▶ Runner Image
11 ▶ Runner Image Provisioner
13 ▶ GITHUB_TOKEN Permissions
17 Secret source: Actions
18 Prepare workflow directory
19 Prepare all required actions
20 Getting action download info
21 Download action repository 'actions/checkout@v3' (SHA:f43a0e5ff2bd294095638e18286ca9a3d1956744)
22 Download action repository 'docker/login-action@v2' (SHA:465a07811f14bebb1938fbed4728c6a1ff8901fc)
23 Download action repository 'docker/build-push-action@v4' (SHA:0a97817b6ade9f46837855d676c4cca3a2471fc9)
24 Complete job name: build
```

Checkout code

0s

```
1  ▶ Run actions/checkout@v3
14 Syncing repository: Kashishsingla111/docxtopdf
15 ▶ Getting Git version info
19 Temporarily overriding HOME='/home/runner/work/_temp/581fd755-b5d7-4a8d-8884-2407f52dad3b' before making global git config changes
20 Adding repository directory to the temporary git global config as a safe directory
21 /usr/bin/git config --global --add safe.directory /home/runner/work/docxtopdf/docxtopdf
22 Deleting the contents of '/home/runner/work/docxtopdf/docxtopdf'
23 ▶ Initializing the repository
37 ▶ Disabling automatic garbage collection
39 ▶ Setting up auth
45 ▶ Fetching the repository
69 ▶ Determining the checkout info
70 ▶ Checking out the ref
74 /usr/bin/git log -1 --format='%H'
75 '53a125552e5cb44418c21ed148bf7aa5e4c20545'
```

Log in to Docker Hub

```
1  ▶ Run docker/login-action@v2
7  Logging into Docker Hub...
8  Login Succeeded!
```

```
Build and push Docker image                                   13s

Post Build and push Docker image                              0s
1  Post job cleanup.
2  ▶ Removing temp folder /tmp/docker-actions-toolkit-KvXcF4

Post Log in to Docker Hub                                     0s
1  Post job cleanup.
2  /usr/bin/docker logout
3  Removing login credentials for https://index.docker.io/v1/

Post Checkout code                                            0s
1  Post job cleanup.
2  /usr/bin/git version
3  git version 2.47.0
4  Temporarily overriding HOME='/home/runner/work/_temp/e1eb95bd-b2ed-4a13-bb05-59cf9ddb9bcb' before making global git config changes
5  Adding repository directory to the temporary git global config as a safe directory
6  /usr/bin/git config --global --add safe.directory /home/runner/work/docxtopdf/docxtopdf
7  /usr/bin/git config --local --name-only --get-regexp core\.sshCommand
8  /usr/bin/git submodule foreach --recursive sh -c "git config --local --name-only --get-regexp 'core\.sshCommand' && git config --local --unset-all
   'core.sshCommand' || :"
9  /usr/bin/git config --local --name-only --get-regexp http\.https\:\/\/github\.com\/\.extraheader
10 http.https://github.com/.extraheader
11 /usr/bin/git config --local --unset-all http.https://github.com/.extraheader
12 /usr/bin/git submodule foreach --recursive sh -c "git config --local --name-only --get-regexp 'http\.https\:\/\/github\.com\/\.extraheader' && git config --
   local --unset-all 'http.https://github.com/.extraheader' || :"
```

```
Complete job

1  Cleaning up orphan processes
```

After the workflow completes, go to your Docker Hub account.

Confirm that the image has been uploaded with the tag latest.

Docker Hub:

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	21 minutes ago	21 minutes ago

See all

kashishsingla50/docxtopdfwf:latest

MANIFEST DIGEST sha256:a0c020a67b1b06066609df29363ad2b847d39ff8ae316fe48b2977134fe64bb2

OS/ARCH	COMPRESSED SIZE	LAST PUSHED	TYPE	MANIFEST DIGEST
linux/amd64	61.98 MB	7 minutes ago by kashishsingla50	Image	sha256:a0c020a6...

Image Layers | Vulnerabilities

Image Layers

1	ADD rootfs.tar.xz / # buildkit	27.78 MB
2	CMD ["bash"]	0 B
3	ENV PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/b…	0 B
4	ENV LANG=C.UTF-8	0 B
5	RUN /bin/sh -c set -eux;	3.35 MB
6	ENV GPG_KEY=E3FF2839C048B25C084DEBE9B26995E310250568	0 B
7	ENV PYTHON_VERSION=3.9.20	0 B
8	ENV PYTHON_SHA256=6b281279efd85294d2d6993e173983a57464c0133956fbbb5536ec9646beaf0c	0 B

Command

```
ADD rootfs.tar.xz / # buildkit
```


8	ENV PYTHON_SHA256=6b281279efd85294d2d6993e173983a57464c0133956fbbb5536ec9646beaf0c	0 B

9	RUN /bin/sh -c set -eux;	14.24 MB
10	RUN /bin/sh -c set -eux;	255 B
11	CMD ["python3"]	0 B
12	WORKDIR /app	93 B
13	COPY requirements.txt requirements.txt # buildkit	177 B
14	RUN /bin/sh -c pip install	16.61 MB
15	COPY . . # buildkit	2.14 KB
16	EXPOSE map[5000/tcp:{}]	0 B
17	CMD ["python" "app.py"]	0 B

Test Docker Image Locally:

```
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf\docxtopdf> docker pull kashishsingla50/docxtopdfwf:latest
latest: Pulling from kashishsingla50/docxtopdfwf
2d429b9e73a6: Already exists
4920a3bd5f7e: Already exists
77edb37367fa: Already exists
02c34c079cc8: Already exists
ad977e12f2ab: Pull complete
c75c14d6ddac: Pull complete
7b58b4fd8aef: Pull complete
67e688090170: Pull complete
Digest: sha256:a0c020a67b1b06066609df29363ad2b847d39ff8ae316fe48b2977134fe64bb2
Status: Downloaded newer image for kashishsingla50/docxtopdfwf:latest
docker.io/kashishsingla50/docxtopdfwf:latest
(base) PS C:\Users\hp\OneDrive\Desktop\wordtopdf\docxtopdf> docker run -p 80:80 kashishsingla50/docxtopdfwf:lat
est
Starting Flask application...
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server i
nstead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.17.0.2:5000
Press CTRL+C to quit
```

```
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 482-263-761
```

Output:

Upload your .docx file

Choose .docx file: Choose File No file chosen

Password (optional): Enter password

Convert to PDF

5. **Challenges Faced**

- Resolving merge conflicts during Git operations.
- Ensuring the CI/CD pipeline ran successfully on GitHub Actions.
- Debugging errors in the Kubernetes manifests and Dockerfile.

6. **Output**

- A fully functional application running on Kubernetes that allows users to upload .docx files, view metadata, and download PDFs.

7. **Bash Script for Running the Docker Container**

```
#!/bin/bash
# Build the Docker image
docker build -t docx-to-pdf .

# Run the Docker container
docker run -p 5000:5000 docx-to-pdf
```