To create a pipeline for this process, you can use a combination of GitHub webhooks, a shell script, and Docker. Here's a basic outline of the steps:

**Set up a GitHub webhook:**

* 1. Go to your GitHub repository.
  2. Click on "Settings" -> "Webhooks" -> "Add webhook".
  3. Set the Payload URL to your Ubuntu machine's IP address or domain name followed by a specific port (e.g., http://your-ip:your-port/webhook).
  4. Set the Content type to "application/json".
  5. Select "Just the push event" for the "Which events would you like to trigger this webhook?" option.
  6. Add the webhook.

**Create a script to handle the webhook:**

* 1. On your Ubuntu machine, create a shell script to handle the webhook payload. This script will clone the repository and dockerize the app.
  2. For example, you can create a script named deploy.sh with the following content:

Bash

#!/bin/bash

# Path to your app directory

APP\_DIR="/path/to/your/app"

# Port to expose your app on

PORT=8080

# Clone the repository

git clone https://github.com/your-username/your-repo.git $APP\_DIR

# Dockerize the app

docker build -t your-app $APP\_DIR

docker run -d -p $PORT:80 your-app

· Make sure to replace /path/to/your/app, 8080, your-username, and your-repo with your actual values.

· · **Set up a simple HTTP server to receive webhook requests:**

* · You can use a tool like nc (netcat) to listen for incoming webhook requests and execute the deploy.sh script.
* Install nc if you don't have it: sudo apt-get install netcat
* Create a simple listener script (listener.sh):

Bash

#!/bin/bash

while true; do

# Listen for webhook requests

echo "Listening for webhook..."

nc -l -p your-port -c 'echo "Webhook received"; ./deploy.sh'

done

Replace your-port with the port you specified in the GitHub webhook configuration.

**Run the listener script:**

* + Make the listener script executable: chmod +x listener.sh
  + Run the listener script: ./listener.sh

**Testing:**

* + Make a commit to your GitHub repository.
  + Check if the changes are cloned and dockerized on your Ubuntu machine.
  + Access your app via the exposed port on your local machine.

Please note that this is a basic setup. Depending on your requirements, you may need to enhance security measures and error handling in your scripts.