DevConnect

Django WebApp

07-08.11.2023

https://github.com/TamarNidam/Devconnect-App

Tamar Nidam
tamar.nidam1@gmail.com

תוכן עניינים:

Overview

In a rapidly evolving digital landscape, DevOps has become the driving force behind innovation and efficiency in the world of software development and IT operations.

However, as the DevOps field expands, professionals are finding it increasingly challenging to connect with like-minded individuals, share best practices, and keep up with the latest industry trends. This gap in communication and collaboration has sparked the creation of DevConnect, a revolutionary social web app designed to bring the global DevOps community together.

With a clear vision in mind, the founders assembled a dedicated team of DevOps engineers, software developers, and designers to bring DevConnect to life.

The founders chose you as a DevOps engineer for their biggest product The DevConnect web app.

You are an essential part of this ambitious project, entrusted with the responsibility to ensure the smooth deployment, scaling, and maintenance of the platform. Your expertise in automation, continuous integration, and continuous delivery is critical for the success of

DevConnect.

Go "DevOps" your way through this project, following best practices and GCP technologies to create an outstanding platform that unites the global DevOps community.

Goals

1. Dockerization
2. Deployment
3. CI/CD

The app: http://localhost:8000/

Clusters:https://console.cloud.google.com/kubernetes/list/overview?project=devconnect-final-project

Instance:
https://console.cloud.google.com/compute/instances?project=devconnect-final-project&pageState=(%22instances%22:(%22p%22:2))

הקמת סביבה:

Django היא מסגרת אינטרנט ברמה גבוהה של Python המאפשרת למפתחים לבנות במהירות יישומי אינטרנט. הוא עוקב אחר הדפוס הארכיטקטוני של MVC (Model-View-Controller) ומתמקד בשימוש חוזר ובפיתוח מהיר. Django מספקת קבוצה של כלים וספריות המפשטות משימות פיתוח אינטרנט נפוצות, כגון טיפול בניתוב כתובות אתרים, ניהול מסדי נתונים, טיפול בטפסים ואימות משתמשים.

הורדת קובץ הZIP. יצירת docker file מתאים, וקובץ requirements.txt עבור הפרויקט DJANGO.

```
requirements.txt
Django==4.2.8
django-crispy-forms==2.0
Pillow==10.1.0
```

הקובץ טקסט נוצר כדי להבטיח שכל התלות הדרושות מותקנות בעת הגדרת פרויקט Django.

וכן בדוקרפייל מריצים אותו כדי להבטיח שכל הקוד והתלויות הנדרשות יתקנו בתוך הקונטיינר.

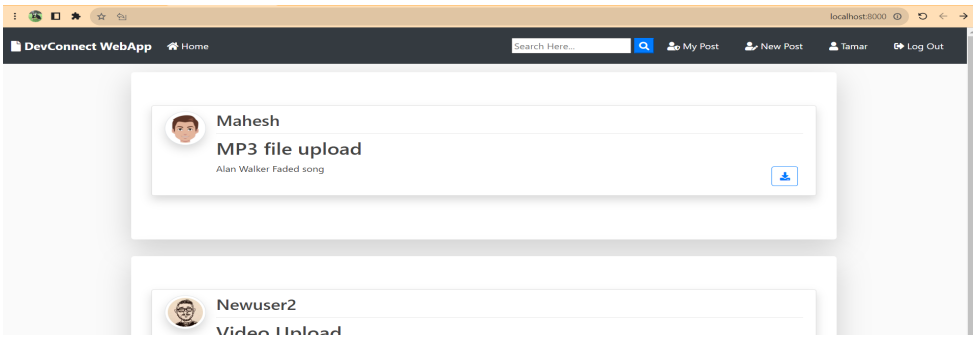
```
Dockerfile > ...
# Use an official Python runtime as the base image
FROM python:3.9

# Set the working directory in the container
WORKDIR /app


# Copy the requirements file to the container
COPY requirements.txt ./

# Install the project dependencies
RUN pip install -r requirements.txt

# Copy the project files to the container
COPY . .

# Expose the port that Django runs on
EXPOSE 8000

# Define the command to run when the container starts
CMD ["python", "django_web_app/manage.py", "runserver", "0.0.0.0:8000"]
```

ניתן להחליף את הפקודה CMD ב:ENTRYPOINT
CMD משמש כדי לספק ארגומנטים של ברירת מחדל שניתן לעקוף, בעוד ENTRYPOINT מגדיר את

הפקודה הראשית שתבוצע. CMD משמש בדרך כלל להגדרת התנהגות ברירת מחדל או מתן ארגומנטים, בעוד ENTRYPOINT שימושי כאשר ברצונך להגדיר את הפקודה הראשית עבור המיכל.

בנייה והרצה בDOCKER.

יצירת קבצי INIT וDELETE לאוטומציה.
בדיקת פעילות תקינה של האפליקציה.



הפעלת קונטיינר עם נפח מותקן.

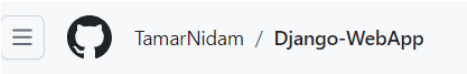
הפעלת קונטיינר עם אמצעי אחסון מותקן כרוכה בחיבור ספרייה ספציפית במחשב המארח לספרייה מתאימה בתוך הקונטיינר. זה מאפשר למכולה לקרוא ולכתוב נתונים למערכת הקבצים של המארח, ולספק עמידות נתונים גם אם המיכל נעצר או נמחק.

הרכבה של אמצעי אחסון מועיל מכיוון שהיא מבטיחה ששינויים שנעשו בתוך המכולה (למשל, עדכוני נתונים, שינויי תצורה) מאוחסנים חיצונית, בנפרד מהאחסון החולף של המכולה. גישה זו מאפשרת למיכל לשמור על מצבו במהלך הפעלה מחדש או כאשר מכולה חדשה מסתובבת. זה גם מאפשר גישה קלה וגיבוי של נתונים מחוץ לקונטיינר. על ידי מיפוי נפח, אתה יכול לשמור על עקביות ושלמות הנתונים, להבטיח חוויה חלקה למשתמשים באינטראקציה עם האפליקציה.

לכן כשניכנס לאפליקציה לא תדרש ממנו הרשמה נוספת..

אחרי בניית התמונה(לפני ההרצה) נריץ את הפקודה `docker volume create db`

העלאת הפרויקט לGITHUB



תקלות:

.1

```
python: can't open file '/app/manage.py': [Errno 2] No such file or directory
```

פתרון:

שינוי מיקום הקובץ בדוקרפייל ל: `django_web_app/manage.py`

.2

הקונטיינר רץ כרגיל אבל לא מוצג כראוי בדפדפן

פתרון:

בעיה בקובץ BASE.HTML:

שינוי השורה ל **`{% load static %}`**

בגרסאות קודמות של Django (לפני 1.9), אפליקציית הקבצים הסטטיים כונתה staticfiles. לכן, ספריית תגי התבניות שטענה את הקבצים הסטטיים הייתה {% load staticfiles %}.

החל מ-Django 1.9, השם של אפליקציית הקבצים הסטטיים שונה לstatic. כתוצאה מכך, ספריית תגיות התבניות לטעינת קבצים סטטיים שונתה גם היא ל{% load static %} כדי לשקף את מוסכמות השמות החדשה הזו.

.3

הכניסה להרשמה והתחברות עם תקלותת

פתרון:

הסרת משתני הסביבה **ENV PYTHONUNBUFFERED=1** מהדוקרפייל

והרצה שוב ושובבב

הוספת השורה:

```
DEFAULT_AUTO_FIELD = 'django.db.models.AutoField'
```

.4

python: can't open file '/app/django_web_app/manage.py': [Errno 2] No such file or directory

פתרון:

סידור ניתוב התקייה בה הפרויקט יורץ לתיקייה בה יש את קבצי הפרוייקט

```
docker run -d -p 8000:8000 -v db:/app django-webapp
```

.5

.remote: Permission to smahesh29/Django-WebApp.git denied to TamarNidam

fatal: unable to access 'https://github.com/smahesh29/Django-WebApp.git/': The requested URL returned error: 403

פתרון:

עריכה שנית של הרפוזטורי:

git remote set-url origin https://github.com/TamarNidam/Django-WebApp<

DEPLOYMENT:

יצירת ארטיפקט רג'יסטרי.

tamar-nidam-artifacts	Docker	Standard	me-west1 (Tel Aviv)

יצירת קלאסטר.

נגדיר TAINT.

טיינטס הם מאפיינים של Nodes ב-Kubernetes, המאפשרים למנהלי מערכת לשלוט בכך שרק פודס מסוימים יוכלו להקצות לעצמם את הצמתים הללו.

פה- פוד עם התגית webapp=mywebapp יוכל להקצות לעצמו את הצומת, אך פוד עם התגית webapp=otherwebapp לא יוכל.

טיינטס יכולים לשמש למספר מטרות, כגון:

למנוע מפודס מסוימים להקצות לעצמם צמתים עם מאפיינים מסוימים. לדוגמה, ניתן להשתמש בטיינט כדי למנוע מפודים עם דרישה גבוהה למשאבים להקצות לעצמם צמתים עם משאבים מוגבלים.

לאלץ פודים מסוימים להקצות לעצמם צמתים עם מאפיינים מסוימים. לדוגמה, ניתן להשתמש בטיינט כדי לאלץ פודים עם דרישות אבטחה מסוימות להקצות לעצמם צמתים עם סט מסוים של עדכונים.

✓	tamar-nidam-cluster	northamerica-northeast2-b	1	2	1 GB	⚠ Pods unschedulable	–

חיבור בין הדוקר לארטיפקט רג'יסטרי.

https://www.googlecloudcommunity.com/gc/Developer-Tools/Permission-quot-artifactregistry-repositories-uploadArtifacts/m-p/543870

gcloud config unset auth/impersonate_service_account

gcloud auth login

gcloud config set project devconnect-final-project

יצירת קובץ DEPLOY

```
$ deploy.sh
image="tamarnidam-webapp"
artifact_registry="me-west1-docker.pkg.dev/devconnect-final-project/tamar-nidam-artifacts/$image"
gcloud auth configure-docker me-west1-docker.pkg.dev
docker tag "$image:latest" "$artifact_registry:latest"
docker push "$artifact_registry:latest"
```

```
1.0: digest: sha256:866a7f1daa5b790c2a2f7ee76a2b32a488d43baf6c0ba98a4b2cf4cd193d784e size: 2844
```

יצירת production_NAMESPACE ע"י יצירת workloads -> deployment

replicas=1

https://kubernetes.io/docs/concepts/workloads/controllers/deployment/

הnamespace נוצר בקוברנטיס קלאסטר. הוא מייצג קבוצה של משאבי קוברנטיס כמו Pods, Services, Deployments וכו

הוא מאפשר ארגון המשאבים בצורה לוגית ולארגן קבוצות עם הגדרות משותפות.

Deployment.yaml

tamar-nidam-deployment

יצירת SERVICE המציין את הסוג כ-LoadBalancer וממפה אותו לפריסה.

על ידי שימוש בשירות Load Balancer, אפשר להבטיח שהאפליקציה שלך נגישה מכל מקום באינטרנט. זה עוזר לנהל את התעבורה הנכנסת, מפיץ אותה על פני מספר תרמילים עורפיים, ומספק דרך אמינה וניתנת להרחבה לגשת לאפליקציה דרך דפדפן אינטרנט.

service.yaml

Name	Type	Endpoints
tamar-nidam-deployment-service	Load balancer	35.203.124.122:80

קבלת כתובת הIP

:loadBalancer

:ingress

- ip: 35.203.124.122

תקלות:

1.

לאחר יצירת הקלאסטר לא ניתן לערוך אותו ולהוסיף לו NODE

התקלה נוצרה עקב חוסר בכתובות IP

פתרון:

יצרו VPC נוסף (DEFUALT2) וניצור לשם את הקלאסטר

]	tamar-nidam-cluster	us-west1-c	1	2	4 GB	

.2

	tamar-n-cluster	us-west1-c	1	2	4 GB	–
	tamar-nidam-cluster	northamerica-northeast1-c	0	0	0 GB	

בדיקת ZONES זמינים:

\ curl -X GET

'https://compute.googleapis.com/v1/zones?filter=region:us-central1'

.3

```
   "marketplace.gcr.io": "gcloud",
docker.pkg.dev/devconnect-final-project/tamar-nidam-artifacts/tamarnidam-webapp]
f933c247acec: Waiting
f933c247acec: Pushing [>                                                  ]  528.9kB/97.37MB
7ac62b459cf5: Pushing [=>                                                 ]  2.226MB/63.37MB
a51bd168424c: Layer already exists
79052241abb2: Layer already exists
fa83a371445d: Pushed
70866f64c03e: Layer already exists
2d788bc47240: Pushing [====>                                              ]  3.747MB/39.39MB
36e50e0709ee: Pushed
12b956927ba2: Waiting
266def75d28e: Waiting
29e49b59edda: Pushing [===>                                               ]  3.017MB/48.44MB
1777ac7d307b: Pushing [=>                                                 ]   3.25MB/116.5MB
```

```
twork diagnostic detects and fixes local network connection issues.
ecking network connection...
```

בעיות רשת...

.4

Cannot schedule pods: Preemption is not helpful for scheduling.

פתרון:

הוספת:

:tolerations

"key: "webapp -

"operator: "Equal

"value: "mywebapp

"effect: "NoExecute

הבלוק tolerations מאפשר לך לציין תנאים מסוימים עבור הפודים שנוצרו על ידי הפריסה. זה עוזר לקבוע אילו צמתים באשכול יכולים להפעיל את התרמילים האלה. במקרה זה, הבלוק tolerations כולל תנאי יחיד עם המאפיינים הבאים:

- "key: "webapp": תנאי זה תואם לתכונה ספציפית הנקראת "webapp".

- "operator: "Equal": זה מציין שערך התכונה חייב להיות התאמה מדויקת.

- "value: "mywebapp": זה מגדיר את הערך הצפוי של התכונה ל-"mywebapp".

- "effect: "NoExecute": זה מציין את הפעולה שיש לנקוט אם התרמיל אינו עומד בדרישות הסבילות. במקרה זה, האפקט מוגדר ל-"NoExecute", כלומר הפוד לא יתזמן בצמתים אלא אם כן יש להם את התכונה שצוינה עם הערך "mywebapp".

על ידי הוספת הסבילות הללו לקובץ ה-Deployment, אתה מאפשר לתזמן את הפודים רק בצמתים בעלי תכונה ספציפית בשם "webapp" עם הערך "mywebapp". אם לצומת אין תכונה זו, הפוד לא יתוזמן באותו צומת. זה עוזר להבטיח שהתרמילים נפרסים על צמתים מתאימים בהתבסס על הסבילות שצוינו.

:CI/CD

יצירת Compute engine instance

tamar-nidam-jenkins, service account : DevOps-sa, e2-medium, me-west1(Tel-Aviv).
:.Automation - install docker engine

bin/bash/!#

apt-get update

apt-get install -y docker.io

✓	tamar-nidam-jenkins	me-west1-a	10.208.15.229 (nic0)	34.165.175.150 (nic0)	SSH

✓ Create VM instance "tamar-nidam-jenkins" and its boot disk "tamar-nidam-jenkins" in 1 minute
DevConnect-final-project

יצירת jenkins_lab מקומי

יצירת תקיה

יצירת דוקרפייל

jenkins_lab > dockerfile > ...

```
FROM jenkins/jenkins
USER root
RUN apt-get update && apt-get install -y apt-transport-https ca-certificates curl gnupg lsb-release
RUN curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
RUN echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -c
RUN apt-get update && apt-get install -y docker-ce docker-ce-cli containerd.io
USER jenkins
```

ניתן להוסיף RUNS כמו:

```
RUN groupadd docker

# RUN touch /var/run/docker.sock
# RUN chmod 666 /var/run/docker.sock

RUN usermod -aG docker jenkins

# Downloading gcloud package
RUN curl -O https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-cloud-cli-453.0.0-linux-x86_64.tar.gz

RUN tar -xf google-cloud-cli-453.0.0-linux-x86_64.tar.gz

RUN ./google-cloud-sdk/install.sh -y

RUN rm -rf google-cloud-cli-453.0.0-linux-x86_64.tar.gz

USER jenkins
```

דוקר TAG+PUSH לרגיסטר-קובץ DEPLOY

SSH into the Compute Engine instance

sudo mkdir /var/jenkins_home

sudo mount /dev/sdb /var/jenkins_home

sudo nano /etc/docker/daemon.json

הוספת JSON:

```
{
"data-root": "/var/jenkins_home/docker"
{
```

DOCKERIGNORE:

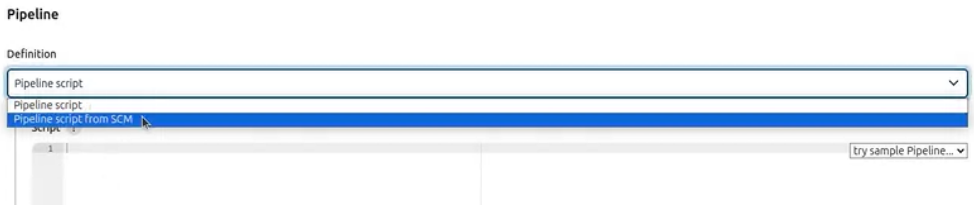
```
.dockerignore
.git
.gitignore
```

sudo systemctl restart docker

פתיחת ג'נקינס ויצירת פרויקט PIPLINE

NEW ITEM->PIPELINE->CREATE

יצירת PIPELINE SCRIPT



docker run -p 8080:8080 -v jenkins home:/var/jenkins_home v/var/run/docker.sock:/var/run/docker.sock -d-name jenkins me-west1-docker .pkg.dev/devconnect-project/tamar-nidam-artifacts/jenkins:1.8.0

תקלות:

1

Create VM instance "tamar-nidam-jenkins" and its boot disk "tamar-nidam-jenkins"in 1 minute DevConnect-project

The CPUS-per-project-region quota maximum in region me-west1 has been exceeded. Current limit: .24.0. Metric: compute.googleapis.com/cpus

פתרון:

יש חריגה מהגבול המותר של כמות המעבדים המרבית בפרויקט שבאזור me-west1. הגבול הנוכחי הוא 24.0 מעבדים. יצירת ה-VM חורגת מגבול זה.

ולכן ניצור אינסטנס עם בחירת ריג'ן אחר בו אפשר ליצור...

.2

חוסר בהרשאות מתאימות לאימות עם Registry Artifact

פתרון:

gcloud auth configure-docker tamar-nidam-artifact

.3

בעיה בהרשאות DOCKER_SOCKET

פתרון:

adjust it permissions from the instance host

```
sudo chmod 666 /var/run/docker.sock
```

ואז

```
docker exec -it jenkins bash
```