תרגיל בית 2 - להגשה עד התאריך 9.5.2022

התרגיל

בתרגיל זה ניצור לנו אפליקציה בסיסית מאוד של שרת WEB ומסד נתונים, ואז נפרוש אותה בענת בתצורת containers בעזרת terraform.
בנוסף לקבצים המבוקשים בתרגיל, אנא צרפו מסמך קצר הכולל:

1. תיאור של הקבצים המצורפים
2. דיאגרמת ארכיטקטורה פשוטה שמסבירה את האפליקציה
3. כל הסבר נוסף שנדרש על מנת להבין את מה שביצעתם

יצירת האפליקציה

מטרת חלק זה היא:

1. ליצור container שמריץ RDBMS כלשהו, ולייצר בו תוכן כלשהו (לפחות טבלה אחת עם לפחות עמודה אחת ולפחות שורה אחת).
2. ליצור container שמריץ שרת WEB כלשהו, שניתן לבצע אליו קריאת REST כלשהי שתביא מידע מה-RDBMS.

סטודנטים שמרגישים בנוח לבצע את שני הצעדים הללו ללא הנחיות נוספות רשאים לעשות זאת. יש להגיש כל קוד שכתבתם, כולל DOCKERFILE ו- Docker Compose וכן הוראות כיצד יש להרים את האפליקציה באופן מקומי על מחשב ווינדווס, מק או לינוקס (אחד מהם מספיק), איך לבצע את קריאת ה REST ומה התוצאה המצופה.
סטודנטים שרוצים הנחיה, יכולים ללכת לפי ההוראות הבאות ליצירת container שמריץ שרת מבוסס flask ומתחבר ל-PostgreSQL:
https://haseebmajid.dev/blog/simple-app-flask-sqlalchemy-and-docker
שימו לב שבנוסף למדריך, עדיין צריך להכין Dockerfile עבור שרת ה-flask.

פרישת האפליקציה באמצעות Terraform ו-Managed Instance Group

כעת אנחנו רוצים לפרוש את האפליקציה שיצרנו לענן.
בשלב הראשון, החלטנו לעשות זאת באופן הבא:

1. את שרת ה-WEB נשים ב compute instance עם container optimized OS
2. ננהל מספר שרתי WEB יחדיו ב-Managed Instance Group שיכול לגדול ממכונה אחת לעשר בסביבת ה-Production (בפיתוח מספיק שיגדל עד שתי מכונות)
3. את ה-RDBMS נקבל משירות מנוהל, CloudSQL

הדרכה כיצד ליצור PostgreSQL מנוהל ב-CloudSQL ניתן למצוא כאן:
https://cloud.google.com/sql/docs/postgres/create-instance
כמובן, מכיוון שהקשבנו למשהו כמו ארבע שעות הרצאה על Terraform, אנחנו רוצים להגדיר את התשתיות ב-Terraform.

כדי לאפשר לנו הפרדה בין סביבת הפיתוח לסביבת ה-Production, נרצה לייצר שתי סביבות נפרדות.
לכן, בחלק זה בתרגיל:

1. צרו Terraform Config שיתאר את פריסת השירות כפי שתואר למעלה
    a. יש לדאוג שהפרישה תריץ את הקונטיינרים שיצרתם. ניתן לארח את הקונטיינרים בשירות GCP Container Registry
2. יש לכלול סביבת Dev וסביבת Production נפרדות, עם ההבדלים שתוארו למעלה
3. יש להשתמש במודולים כדי להמנע משכפול קוד
4. יש לוודא שה-REST Endpoint אכן קיים, נגיש ומתנהג כמצופה
5. הקפידו שלא לכלול סודות ומידע רגיש בקונפיגורציה. ניתן להזין מידע זה חיצונית ממשתני סביבה
6. צרפו להגשה את:
    a. קוד ה-Terraform שכתבתם
    b. פלט פקודת Terraform plan שמתבצעת מול פרוייקט ריק

כעת ברצוננו לעדכן את הקוד בשרת ה-WEB. בצעו שינוי כלשהו בקוד השרת, צרו גרסא חדשה של הקונטיינר, ודחפו את השינוי לסביבת הענן באמצעות Terraform.
יש לצרף את פלט פקודת Terraform plan וכן הסבר כיצד ביצעתם את שינוי הגרסא.
לאחר שסיימתם הקפידו לבצע terraform destroy ולמחוק את הקונטיינרים שלכם מה-container registry כדי שלא לסיים את כל התקציב בפרוייקט.
שימו לב שלעיתים יצירת database ב-cloudsql לוקחת זמן רב, ואם זה מפריע כדאי ליצור את ה-database ידנית ולאחר מכן לבצע terraform import.