מדדי KPI לפרויקט:

* User Satisfaction
  + שהמשתמש יהיה מרוצה מתוצאת החיפוש, כלומר שהתוצאות חיפוש יכילו תוצאותנכונות.
  + מדידה: שתהיה לפחות תוצאה אחד נכונה \ רלוונטית ב-5 התוצאות הראשונות.
* Performance
  + שהמשתמש לא ימתין יותר מדי זמן לתוצאת החיפוש.
  + מדידה: אחרי לחיצה על כפתור החיפוש לא יעברו יותר מ-3 שניות עד שתוצג תוצאה כלשהי.
* Scalability & Elasticity
  + שהמערכת תהיה מודולרית, וקלה להרחבה.
  + מדידה: פיתוח מודולרי, שכל פונקציה ומחלקה תהיה עצמאית, כלומר בלי תלות במשתנים גלובלים.

בונוס: כיצד לדעתכם ניתן לממש את הפרויקט שלכם כ-serverless?

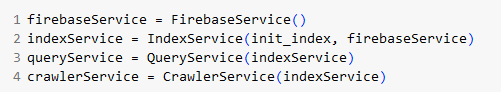
* ניתן לממש אותו כ-serverless על ידי שמירת כל השירותים לוקלית. ובכך מורידים את התלות שלהם בשרת.

הסבר על הארכיטקטורה:

חלקנו את הפונקציוניות העיקרית לשירותים, השירותים שלהם:

* FirebaseService: מטרתו לייעל את התקשור עם הבסיס נתונים, לבצע שליפת ועדכון נתונים בבסיס נתונים.
* IndexService: מטרת השירות לאחסן את האינדקסים לוקלית, הוא מאחסן אינדקס הפוך לצורך חיפוש, ואת האינדקסים של הדפים (לצורך סטטיסטיקה), והוא יאפשר לעדכן את האינדקס לפי בקשות (כגון הוספת מילה לאינדקס, הסרת דף מהאינדקס וכו'). הוא מאפשר שמירה ושליפה של האינדקס מבסיס הנתונים על ידי dependency-injection של שירות FirebaseService.
* CrawlerService: מטרתו לבצע שירותי crawling על דפים, ולעדכן את השירות אינדקס תוך כדי (בעזרת dependency-injection).
* QueryService: משתמש לביצוע שאילתות ומחזיר תוצאות חיפוש, הוא גם מאפשר קבלה של ההיסטוריית חיפוש. השירות צריך dependency-injection של indexService על מנת לבצע את השאילתות.
* GraphService: מטרתו ליצור את הגרפים בצורה אוטומטית עבור אינדקס הפוך.
* המשך בעמוד הבא

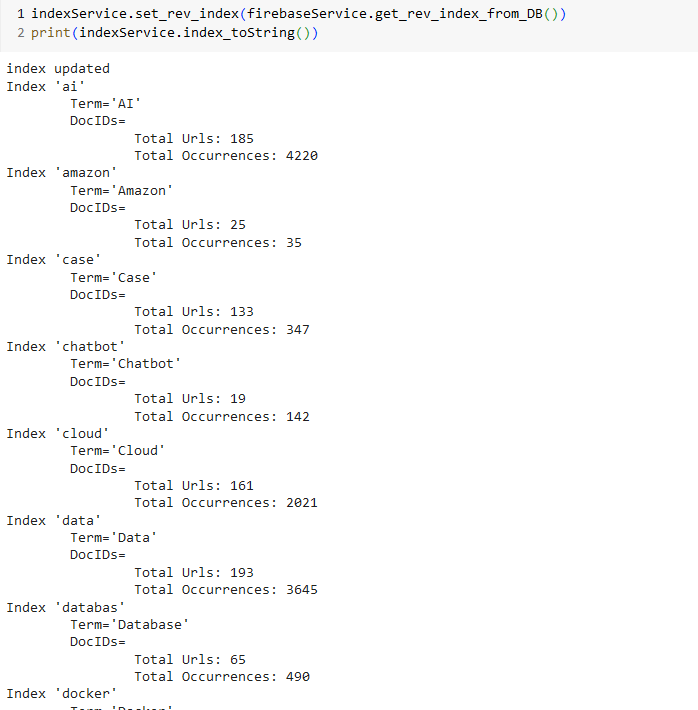
דוגמאות לשימוש:



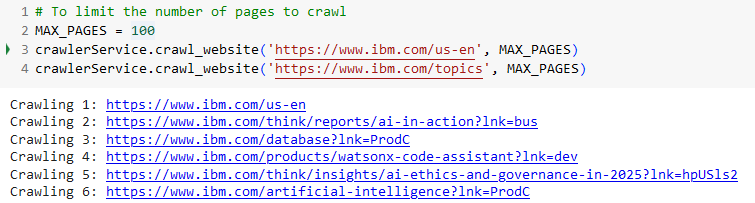
* שירות FirebaseService וב-IndexService:
  + עדכון אינדקס בבסיס נתונים



* + שליפת האינדקס מבסיס הנתונים:



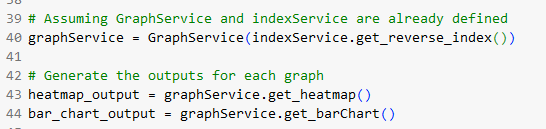
* דוגמא לשימוש ב-CrawlerService:



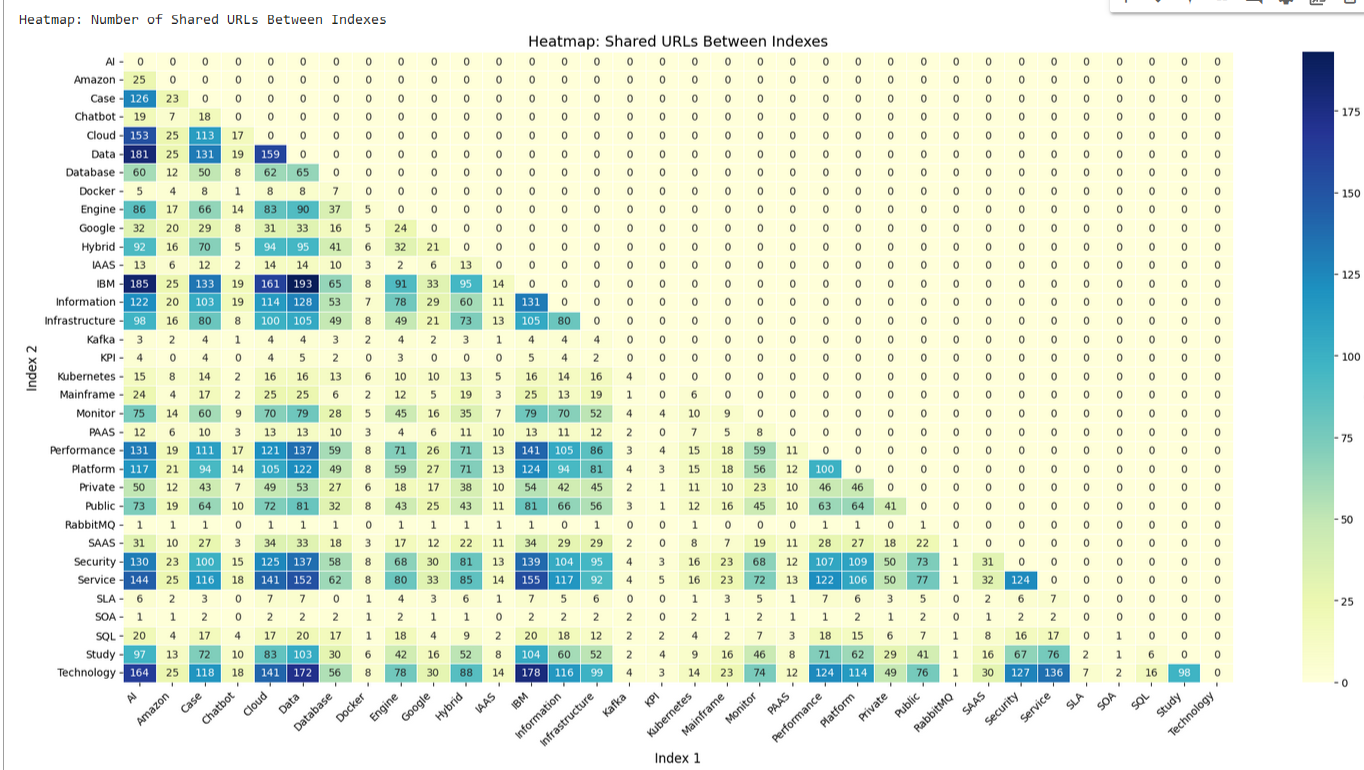
* דוגמא לשימוש ב-QueryService:

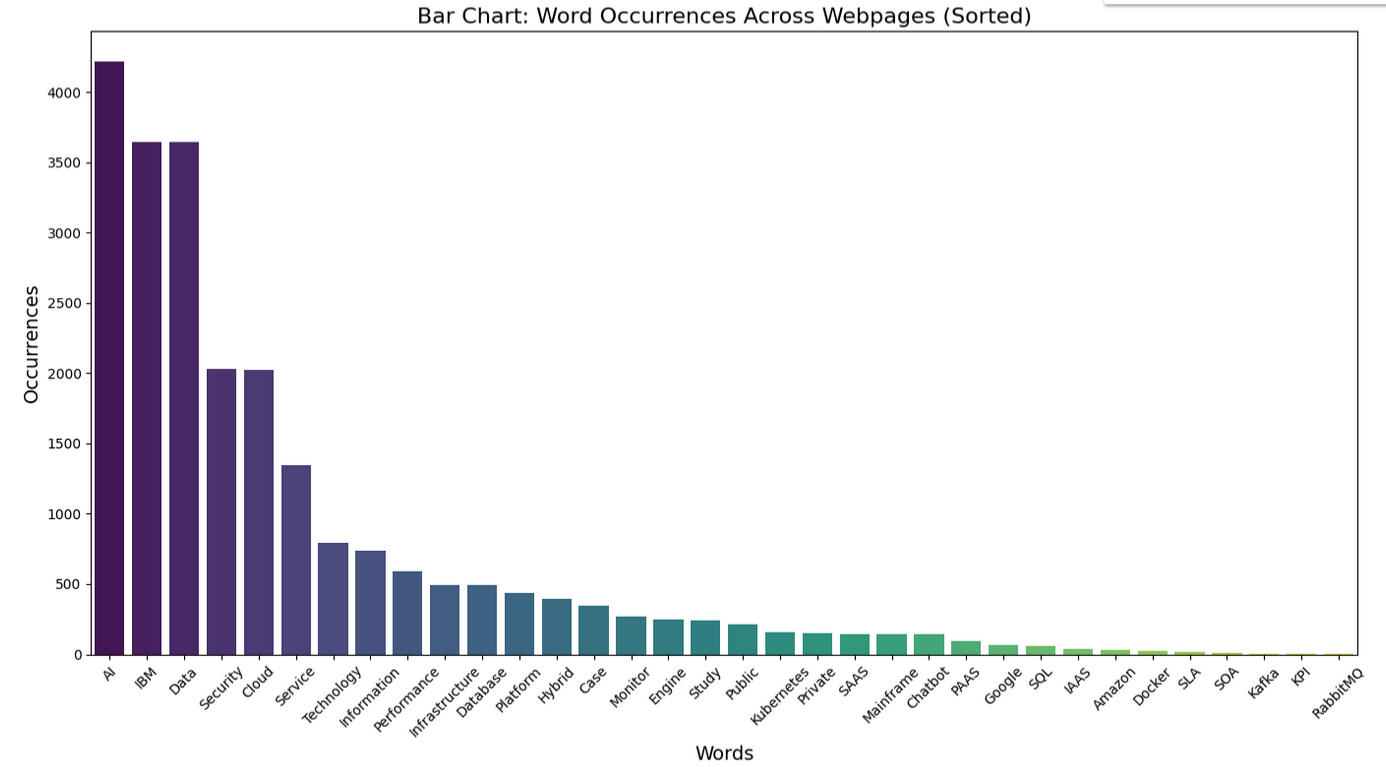


* המשך בעמוד הבא.
* דוגמא לשימוש ב-GraphService



* + התצוגה של הגרפים במסכים:

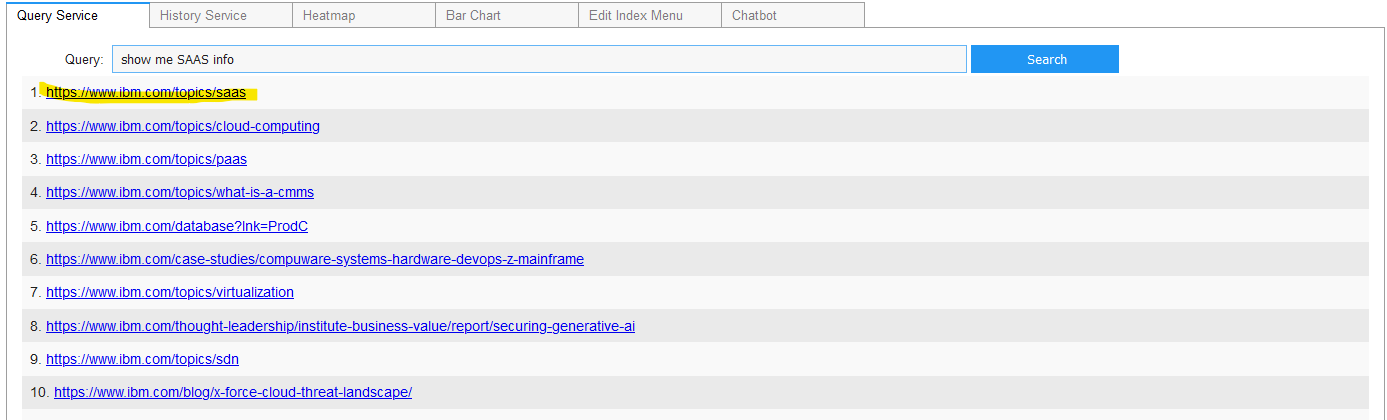




* המשך בעמוד הבא.

תוצאות בדיקות:

* בדיקת User Satisfaction:



* + תוצאה הכי נכונה הייתה עם הדירוג הכי גבוה.
* בדיקת Performance:
  + החיפוש לקח פחות משנייה.
* בדיקת Scalability & Elasticity:
  + בשירותים אין תלות במשתנים גלובלים, ואם יש תלות בשירות אחר אז הוא מקבל אותו כ-Dependency-Injection בבנאי מה שמאפשר לפיתוח יותר מודולרי.