**מבוא למחשוב ענן – קבוצת Tiger**

קישור לגוגל קולאב: <https://colab.research.google.com/drive/1FIctTIYxFi1hUt1UKq2afl7wPgEtWITm>

קישור לתיקיית הגיט: <https://github.com/goodpvp90/CloudComputing>

**מהות המוצר ומרכיביו:**

* 1. מודולים ופיצ'רים מעניינים:

הפיצ'ר המעניין שהוספנו חוץ מצ'אט בוט מבוסס Gemini, הוא בדיקה לכמות המילים שהם Action vs. Restriction, ובנינו גרפים מעניינים שמראים כי האתר של Vercel מציג שפה שמאפשרת למשתמשים הרבה יותר מאשר שהיא מגבילה אותם (בקונטקסט של בניית אתרים).

A blue and orange rectangular graph

Description automatically generated

* 1. Microservices שנעשה בהם שימוש:

תשובה:בחלק הראשון של הקוד (Backend), חילקנו את פעולות הזחילה שאנחנו מבצעים לארבעה מיקרו שירותים:

* **Upload object service:** מיקרו שירות שמאפשר לעדכן אובייקט ששמור במחלקה ולאחר שנעדכן אותו כרצוננו, הוא מעלה אותו לשרת Firebase. הוא מכיל בתוכו פונקצית העלאה, שבה מעלים את האובייקט השמור לענן, עם כל המילים שנשלפו, הלינקים שבהם המילים מופיעות, כמות המופעים שלהן, וה-score שלהן לכל דף.   
  היתרון בשירות זה הוא הפשטות בשמירת כל הנתונים, כיוון שאנחנו קוראים לו בכל פעם שאנחנו רוצים לשמור בו רשימת מילים והלינקים שבהם הן מופיעות, וכשנרצה, נקרא לפונקציה שמעלה לשרת.
* **Dictionary creation service:** מיקרו שירות שבנינו שמאפשר למשתמש ליצור מילון לכל המילים שקיימות בדף. כאשר נפנה למחלקה זו עם הקלט של כל המילים בדף, היא בונה לנו את המילון של כל המילים הקיימות, כמות המופעים שלהן, ואת המיקום (האינדקס) המדויק בהן הן נמצאות.

בנוסף, יש לה פונקציה שמחשבת את ה-score של כל מילה בדף מסויים. מילה שמופיעה כמות רבה של פעמים ובתחילת הדף, תקבל ערך גבוה יותר ממילה שמופיעה הרבה בסוף הדף. היתרון בשירות הוא שניתן לפנות למחלקה הזאת עם מערך של מילים, והיא תחזיר לנו מילון מסודר ועם כל המידע שנצטרך.

* **Pre-process service:** מיקרו שירות שמחזיק פונקציות לניקוי ועיבוד טקסט. מורכבת מפונקציה מרכזית שמשתמשת ב-5 נוספות. היא מקבלת מחרוזת של מילים, ויודעת לעבד את המחרוזת לרשימה של כל המילים לאחר ניקוי של סימני פיסוק, מספרים, ועוד.

היתרון בשירות זה הוא שהוא בסך הכל מקבל מחרוזת ארוכה, ומחזיר רשימה של מילים "נקיות" למשתמש המבקש.

* **Post-process service:** מיקרו שירות שלוקח את המידע ה-raw ומוסיף metadata עבור כל מונח (word count, page count). כלומר, עבור כל מונח שקיים באתר, נוסיף לו את כמות המופעים שלו בכל הדפים, וכמות הדפים בהם המונח נמצא.

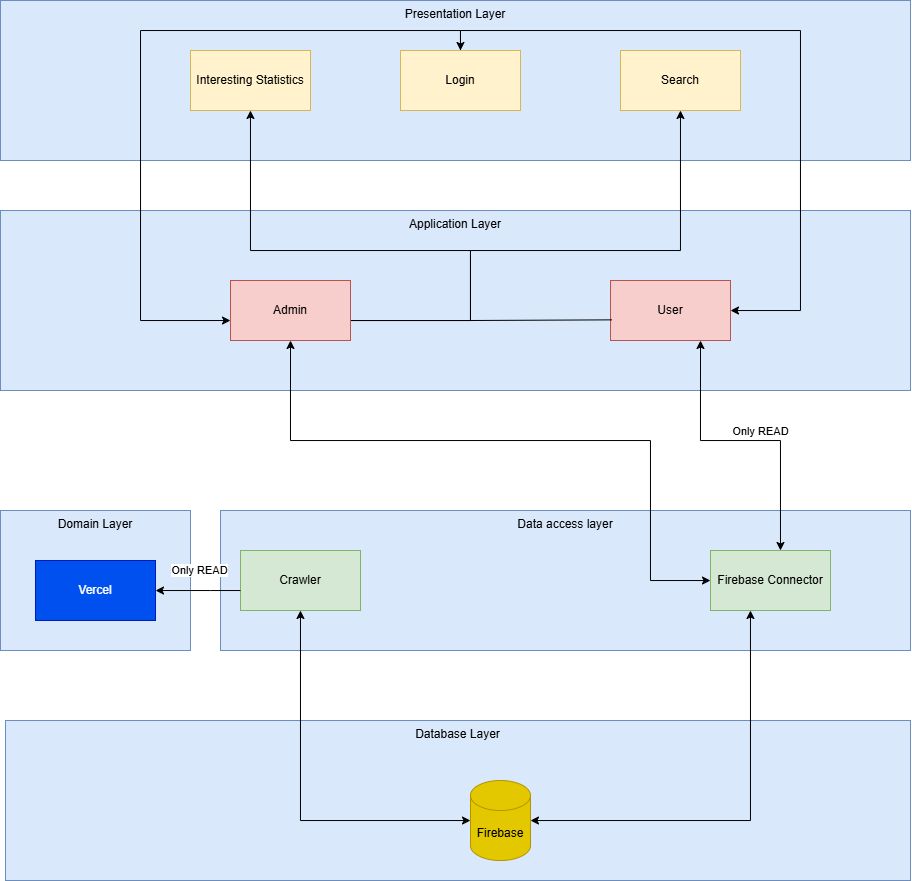
ובנוסף הוא מחשב metadata של הסטטיסטיקות המעניינות (מידע הנחוץ במקרה של שינוי ידני של מסד הנתונים), והסטטיסטיקות עצמן. היתרון של שימוש במיקרו שירות זה הוא פיצול והפרדת האחריות של עיבוד המידע, והוא מרכז פיתוח, תחזוקה ועדכון של הלוגיקה הזאת באופן בלתי תלוי במחלקות אחרות.

* 1. KPI מרכזיים:

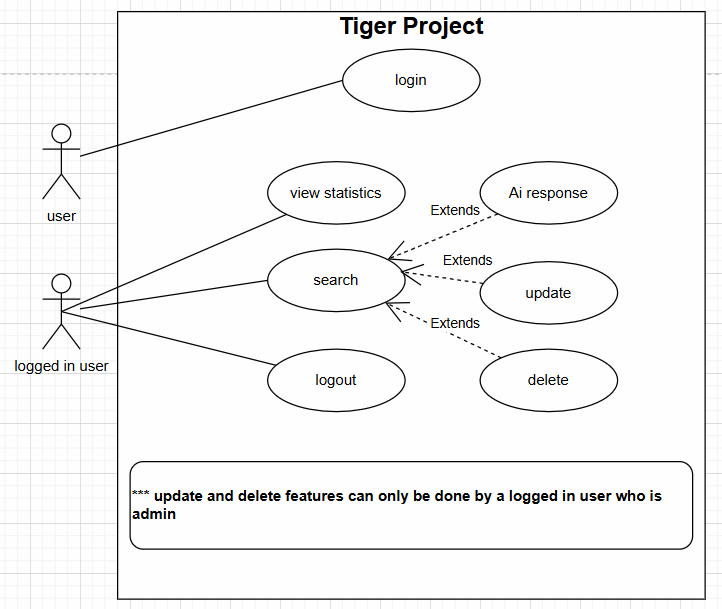
**תשובה**: המערכת שלנו עומדת בשלושה Key Performance Indicators מרכזיים:

* **User experience:** הממשק של המערכת שלנו מינימליסטי, מאורגן, פשוט מאוד לשימוש.
* **Performance:** שליפת הנתונים בעת פנייה לשרת על ידי המשתמשים מתבצעת בצורה מהירה מאוד, מסתכמת בזמני ריצה של ms700-900 בסה"כ.
* **Security:** למשתמשים רגילים אין גישה למסד הנתונים, אלא רק בתהליך הפנייה לצורך החיפוש. משתמש פשוט אינו יכול לגשת לגישת המנהל ולשנות מידע במסד. רק מנהל יכול להתחבר לגישת המנהל ע"י שם משתמש וסיסמא ייחודיים.

1. **ארכיטקטורת המערכת:**
   1. תרשים ומאפיינים מרכזיים:



* 1. Use Case מעודכן של פונקציונליות המערכת:



* 1. דרישות פונקציונליות:

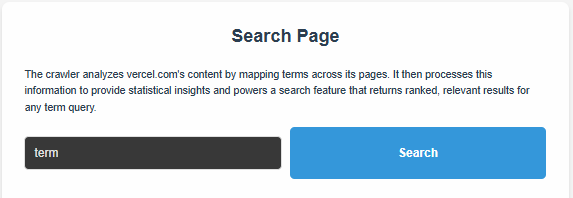
1. המערכת תשלוף מידע מהאתר.
2. המערכת תאפשר חיפוש באתר לפי מילים.
3. המערכת תשמור על ביטחון מידע.
4. המערכת תאפשר הצגת מידע בצורה ויזואלית.
5. המערכת תאפשר הרשמה והתחברות.
6. דרישות לא פונקציונליות:
   1. **Performance**: המידע ישלף באופן יעיל ומהיר.
   2. **Data Integrity**: עדכון הdb יתבצע ע"י משתמשים מורשים (admin) והזחלן בלבד.
   3. **Availability**: השירות יהיה זמין תמיד כתוכנת ענן.
   4. **Security**: הזיהוי יתבצע ע"י הזנת נתוני שם משתמש וססמא.
   5. **Visualization**: המידע יוצג בצורה גרפית לפי שכיחות המילים שהוזנו.
7. ביקורת עמיתים אשר ניתנה במהלך הצגת הפרויקט (שבוע9), וכיצד התמודדתם איתה. נא לחשב גם את ציון SUS ולהתייחס אליו:

|  |  |
| --- | --- |
| שאלה | ציון **SUS** משוקלל |
| הייתי רוצה להשתמש במערכת זו לעיתים תכופות. | 2.86 |
| מצאתי כי המערכת מסובכת ללא סיבה. | 2.59 |
| חשבתי שהמערכת קלה לשימוש. | 3.25 |
| אזדקק לתמיכת איש טכני כדי שאוכל להשתמש במערכת זו. | 3.62 |
| מצאתי כי הפונקציות השונות של המערכת היו מתואמות היטב. | 3.21 |
| חשבתי כי היה יותר מידי חוסר עקביות במערכת זו. | 3.17 |
| לדעתי רוב האנשים יהיו מסוגלים ללמוד להשתמש במערכת זו בקלות. | 3.1 |
| מצאתי כי המערכת מאוד מסורבלת לשימוש. | 3.24 |
| חשתי בטחון רב כאשר השתמשתי במערכת. | 3.03 |
| עלי ללמוד הרבה דברים לפני שאוכל להשתמש במערכת זו. | 3.28 |
| סה"כ לאחר כפל ב2.5. | **78.3** |

**הסבר**: הציון שקיבלנו הוא מעל הממוצע העולמי שעומד על 68, ולכן אנחנו מסיקים שהמערכת שלנו היא ידידותית למשתמש, קלה ומסודרת, אבל בגלל שהציון לא עולה על 80 שהוא ציון מצוין למערכת, לדעתנו ניתן לשפר אותו באמצעות שיפור והוספת אלמנטים עיצוביים לממשק הסופי שאותו נגיש.

1. תיק התחזוקה מצורף בתיקיית ההגשה בגיט ובנוסף תיקיית ההגשה במודל.
2. תיק המשתמש מצורף בתיקיית ההגשה בגיט ובנוסף תיקיית ההגשה במודל.
3. שקיפות אלגוריתמית – כיצד הבהרתם למשתמשים את האלגוריתמים והנתונים הנאספים בקוד שלכם?

**תשובה**: לפני ובמהלך ביצוע חיפוש בSearch Page אנו מבהירים למשתמש על איך הזחלן שלנו עובד בצורה מונגשת למשתמש במסך החיפוש. לפי דעתנו, אין צורך להסביר למשתמש איך עובד אלגוריתם הניקוד (ranked words) שלנו, ומספיק לספק לו את הניקוד עצמו (score) ואיך שהמידע נאסף מהאתר של Vercel.



1. אתגרים אשר עלו במהלך העבודה, וכיצד התמודדתם איתם.

תשובה:נתקלנו בשני קשיים מרכזיים במהלך תכנון וביצוע הפרויקט:

**Frontend:** במטלה 2 החלטנו לבצע אינטגרציה בין קוד HTML לבין וויג'טים של פייתון. הקוד היה רץ בצורה טובה אבל העיצוב היה מינימלי ולא מאורגן היטב.

במהלך העבודה על תרגיל בית 3, נתקלנו בבעיה בשילוב בין ה-html והוויג'טים וגילינו שלא ניתן לבצע אינטגרציה נקייה בין שתי המערכות בלי לוותר על איכות הניראות של הפרויקט. לכן, החלטנו לשנות את כל הקוד הישן ולהמיר את הוויג'טים ל-HTML, דבר שלקח זמן רב אבל השתלם לנו לבסוף.

**Backend:** נדרש זמן לחשוב על אלגוריתם לדירוג המילים בכל דף. בנינו 5 נוסחאות מתמטיות לחישוב התיעדוף של המילים אבל ללא הצלחה.

לבסוף, ביצענו סיעור מוחות באחת מפגישות הצוות והגענו לאלגוריתם יחסית פשוט שנותן ערך גבוה יותר למילה ככל שהיא נמצאת גבוה יותר בדף, ומגדילה את הערך הזה עוד יותר אם היא מופעה כמה פעמים בדף. האלגוריתם הזה והחישוב שלו מפורטים לעומק בתיק המתכנת, בחלק 1: Backend, במחלקת DictionaryCreationService, פונקציה get\_score\_of\_word\_in\_page.

1. יש להגיש את הטבלה הבאה , תוך התיחסות למשובים שקיבלתם:

|  |  |  |
| --- | --- | --- |
| הערת משוב | האם לדעתכם יש צורך בשינוי במערכת בעקבות ההערה? | נימוק |
| שליפה ותצוגה של הנתונים מהשרת מתבצעת בזמן תגובה של ms700-900. | לא | לדעתנו, מכיוון שהשליפה מתבצעת מהשרת של Firebase, לא ניתן לייעל את מהירות השליפה. מסד הנתונים שלנו מכיל מילים מ-300 דפים ולדעתנו מהירות השליפה וביצוע הפעולות עם המסד הזה כזה היא מספקת. |
| לשפר את אחידות התצוגה ולשנות את הצבעים של הכפתורים. | כן | ביצענו תיקונים קטנים בעיצוב המערכת, כמו שינוי צבע ומיקום הכפתורים. |
| עיצוב דף חיפוש המילים. | כן | שינוי כפתורי העדכון המחיקה וה-back בוצעו בעקבות ההערה. |
| החלוקה של קוד הבאקאנד ל-Microservices. | כן | לדעתנו, כדאי לחלק גם את הקוד של הפרונטאנד למיקרו שירותים בפרויקטים עתידיים. |

הערות מה-Code review:

|  |  |  |
| --- | --- | --- |
| **הערה** | **תגובה** | **האם בכוונת הצוות לבצע שינוי בעקבות ההערה? יש לפרט. אם כן, מה השינוי. אם לא, מדוע?** |
| פשטות | מסכימים | לא משנים מאחר והקוד עומד בדרישות. |
| מימוש | מסכימים | המימוש לא היה מספק מאחר והייתה אפשרות לחיפוש של מונח יחיד בלבד, לכן קיבלנו את ההערה ועדכנו את המערכת לעבוד עם חיפוש של מספר מונחים ככה שהלינקים שמופיעים הם אלו שכל המונחים נוכחים בהם. |
| מודלריות | מסכימים | לא משנים מאחר והקוד עומד בדרישות. |
| יעילות | מסכימים | לא משנים מאחר והקוד עומד בדרישות. |
| באגים | מסכימים | הובהר במהלך הישיבה בין 2 הצוותים כי הטעות של ה-ai response היה בעיית הבנת הדרישה. במקור הפיצ'ר נתן רק הערה קלה בכל חיפוש של term ועודכן להיות צ'אטבוט מלא. |
| טיפול בשגיאות | מסכימים | לא משנים מאחר והקוד עומד בדרישות. |

1. מקורות שהשתמשנו בהם:

Vercel.com

Gemini.google.com

stackoverflow.com

Wikipedia.com