תרגיל בית 3 – מחשוב ענן צוות **Wolf**

מגישים:

* **יובל כוגן - 207840042**
* **דניאל לחמקוב - 319041059**
* **יניב בודאגה – 314700428**
* **ליאור דגש – 314855404**
* **אלמוג רז – 206458911**
* **רון סלמה – 207475146**

**מהנדס המערכת:** דניאל לחמקוב

**חלוקת העבודה:** התפקידים בצוות הוגדרו בהתאם להעדפות ולחוזקות האישיות של כל אחד מהחברים, תוך שאיפה לנצל בצורה מיטבית את הכישורים של כל אחד. עם זאת, הקפדנו שכל חבר צוות ייחשף גם למשימות של שאר החברים, כדי לשמור על ראייה מערכתית רחבה ולהבטיח הבנה משותפת של כלל מרכיבי הפרויקט. לאורך כל העבודה שמרנו על תקשורת רציפה, תיאום מלא ושיתוף פעולה הדוק, אשר אפשרו חפיפה ותמיכה הדדית בין חברי הצוות לאורך כל שלבי הביצוע.

**איטרציה 1**

|  |  |  |  |
| --- | --- | --- | --- |
| שם חבר הצוות | משימות שהוקצו | משימות שהושלמו | Acceptence test – בדיקות קבלה |
| יובל כוגן | בניית צ'אטבוט | בניית צ'אטבוט | אינטרקציה בסיסית – הצ'אטבוט מגיב לשאלות כלליות.  מפתח Gemini-API מוחבא.  השמת Singleton לבוט כך שהשיחה איתו מופיעה בכל החלונות וההיסטוריה של השיחה נשמרת. |
| אלמוג רז | הרצה של בדיקות קבלה + כתיבה ותכנון של מסמך וורד למטלה 3. | הרצה של בדיקות קבלה + כתיבה ותכנון של מסמך וורד למטלה 3. | בדיקה של הבדיקות קבלה על מנת לוודא תקינות ולתת אישור סופי לקראת ההגשה.  יצירת סנכרון בין תוכן הפרויקט לבין התוכן שנכתב לתוך מסמך הוורד. |
| רון סלמה | רפקטורינג על ה-codebase | רפקטורינג על ה-codebase | שיפור קריאות הקוד.  שמירה על פונקציונליות קיימת.  הפחתת כפילויות בקוד היכן שניתן. |
| יניב בודאגה | UX & UI | UX & UI | הטמעה ויישום גורף של אלמנטים עיצוביים.  בדיקות שנעשו מול משתמשים אמיתיים על מנת לחוות מהם איך המערכת הרגישה להם וכך לבצע תיקונים היכן שאפשר. |
| דניאל לחמקוב | הטמעת אלגוריתמיקה למידע עתק במערכת | הטמעת אלגוריתמיקה למידע עתק במערכת | נכונות האלגוריתם בנתונים גדולים.  ביצועי האלגוריתם תחת עומס.  סקאלביליות האלגוריתם.  יעילות ניצול המשאבים. |
| ליאור דגש | בניית מיקרוסרוויסים | בניית מיקרוסרוויסים | פונקציות ליבה עבור כל מיקרוסרוויס.  עמידות וביצועים. |

1. **שימוש במיקרוסרוויסים:**

* **שירות Crawler:**  
  אחראי על הורדת דפי אינטרנט, ניתוח ה-HTML ושליפת טקסטים וקישורים להמשך סריקה.
* **שירות Indexer:**

מקבל את הטקסטים מה-Crawler, מנקה את המידע (הורדת stop words, סטמינג), מחשב את שכיחות המילים ושומר את האינדקס למסד הנתונים Firebase.

**היתרונות בשירותים אלו:**

* הפרדה ברורה בין שליפת המידע לבין עיבוד הטקסטים.
* מאפשר לפתח ולשפר כל שירות בנפרד.
* קל להרחיב בעתיד — למשל להפעיל מספר Crawlers במקביל.

העבודה עם Microservices מאפשרת לנו לפצל את המערכת בצורה גמישה, מודולרית וניתנת לתחזוקה.

1. **KPI ELEMENTS:**

ה-KPI אשר בהם עשינו שימוש בפרויקט הם:

* **User Experience:**

**שיעור שגיאות באפליקציה** קריטי ביותר למערכת שהיא דאשבורד אינטראקטיבי. נבדוק כמה פעמים ווידג'טים לא נטענים, גרפים נכשלים, החיבור לFirebase נופל או הצ'אטבוט מחזיר שגיאה? שיעור שגיאות נמוך מבטיח שהמהנדסים יכולים לסמוך על המידע המוצג.

**שביעות רצון המשתמשים:** למרות שאין מנגנון מובנה למדידה, חשוב לאסוף משוב מהמשתמשים (מהנדסים ומנהלים) לגבי קלות השימוש, בהירות הצגת הנתונים והתועלת של הכלים השונים.

* **Performance:**

**זמן תגובה** נחשב כמדד מרכזי. כמה זמן לוקח לדאשבורד להיטען? כמה מהר מתעדכנים הגרפים והסטטיסטיקות לאחר בחירת סנסור או מאפיין אחר? מהירות התגובה של הצ'אטבוט ושל מסך החיפוש משפיעה ישירות על חווית המשתמש.

**זמן השהיה** (Latency) חשוב במיוחד עבור נתוני הסנסורים. מהו פער הזמן בין קריאת נתון על ידי החיישן הפיזי לבין עליית הנתונים למערכת? עבור "מרכז בקרה", יש לשאוף לזמן השהיה נמוך ככל האפשר כדי להבטיח שהנתונים משקפים את המצב הנוכחי במעבדה.

* **זמינות ואמינות:**

**System Uptime:** האם הדאשבורד זמין ומתפקד כשהמהנדסים זקוקים לו? זהו מדד חשוב שכן רצוי לדעת האם ניתן להשתמש במערכת באופן יום יומי או שמא היא קורסת ותחבל בעבודתם של אנשי המעבדה.Z

**זמן ממוצע בין תקלות:** באיזו תדירות המערכת נתקלת בכשל קריטי למשל, התרסקות של המחשב שמריץ את התוכנה בענן.

**עלות וניצול משאבים:**

**Cost by service** הינו מדד חיוני לניהול. מהן העלויות החודשיות של שירותי הענן שבהם אתה משתמש? זה כולל:

* שימוש ב-Firebase
* עלות זמני הריצה של המערכת על גבי הפלטפורמה של Collab.
* עלות השימוש ב-Gemini API עבור הצ'אטבוט לפי כמות התוקנים.

**שיעור ניצול משאבים:** האם המערכת יעילה? לדוגמה, האם היא מבצעת קריאות מיותרות מ-Firebase ביצוע ניטור זה יכול לעזור באופטימיזציה והורדת עלויות.

**שקיפות אלגוריתמית – הושמה בקוד שלנו בדרכים הבאות**:

* **תיעוד פנימי מפורט**

הקוד עושה שימוש נרחב בתאי Markdown עבור כל חלק עיקרי בקוד.

התאים מכילים הסבר על מטרת כל סעיף ומה הוא מכיל.

כיצד זה תורם לשקיפות:

תיעוד פנימי מסוג זה מאפשר למפתחים אחרים, או למשתמשים עם רקע טכני, להבין את הלוגיקה שמאחורי הקוד. הוא אינו מיועד למשתמש הקצה הלא-טכני, אך הוא מהווה בסיס חשוב לשקיפות פנימית ופיתוח.

matplotlib, pandas, numpy, firebase\_admin ועוד.

* **שימוש בספריות מוכרות**

הפרויקט עושה שימוש בספריות פיתון נפוצות ובדוקות כמו matplotlib, numpy, firebase & pandas ספריות אלה ידועות, מתועדות היטב, וברובן הן בקוד פתוח.

כיצד זה תורם לשקיפות:

נגישות לבדיקה: העובדה שהספריות הן בקוד פתוח מאפשרת לכל מי שמעוניין לבדוק את אופן פעולתן הפנימית, כולל האלגוריתמים בהם הן משתמשות.

היכרות ותקן: משתמשים בעלי ידע בתכנות יכולים להניח בביטחון יחסי כי הספריות הללו פועלות באופן צפוי ומתועד.

קהילת מפתחים גדולות: לרוב לספריות הללו יש קהילות מפתחים גדולות, מה שאומר שניתן למצוא מידע רב על אופן פעולתן ועל שימוש בנתונים.

* **ארכיטקטורת הקוד ובניית אחריות**

ארכיטקטורת המערכת נעשתה בצורה מודולרית כך שלכל קטע קוד פונקציונליות שונה במערכת.

כיצד זה תורם לשקיפות:

מבנה ברור: קוד מובנה ומחולק היטב מאפשר הבנה קלה יותר של הפונקציונליות של כל חלק.

הפרדת קונספטים: הפרדה בין ייבוא ספריות, איסוף נתונים, עיבוד נתונים והצגה מאפשרת לזהות באופן ברור איזה חלק בקוד אחראי על איזה תהליך אלגוריתמי או איסוף נתונים. לדוגמה, אם יש סעיף המוקדש לאיסוף נתונים, ברור שבחלק זה מתבצע איסוף הנתונים.

* **גישה לנתונים**

הקוד מייבא את הספרייה firebase\_admin ומכך ניתן להסיק כי קיימת אינטגרציה בקוד מול המסד נתונים Firebase

כיצד זה תורם לשקיפות:

הצהרה על מקור הנתונים: עצם הייבוא של ספריית Firebase מציין למשתמשים ולמפתחים שמקור הנתונים הוא Firebase.

מדיניות נתונים של צד שלישי: ניתן להפנות את המשתמש למדיניות הפרטיות של Firebase עצמה לגבי איסוף ואחסון נתונים, ובכך להעביר את נטל השקיפות לחברה מוכרת ומוסדרת.

1. **מידע עתק:**

בפרויקט השתמשנו בנתונים שנאספו ממספר חיישנים, והחלטנו להטמיע big data במערכת על ידי ביצוע ניתוח אנליטי של מקדם המתאם בין החיישנים בחוץ ולחיישנים שבתוך המעבדה.

1. אתגרים שעלו במהלך הפרויקט:

**התמודדות עם כמויות גדולות של נתונים וביצועים איטיים**

* **האתגר שעלה:** בשלבים מסוימים, כמות הנתונים שרצינו לאחזר ולעבד החלה להיות גדולה, והדבר האט משמעותית את זמן הריצה של המחברת. לא יכולנו להרשות לעצמנו לחכות זמן רב לכל שאילתה.
* **איך התמודדנו:** למדנו להשתמש בשאילתות חכמות יותר ב-Firebase במקום למשוך את כל הנתונים, ביקשנו רק את מה שבאמת היה נחוץ לנו באותו רגע.

**עיצוב וייצוג חזותי ברור של הנתונים**

* **האתגר שעלה:** להציג נתונים זה דבר אחד, אבל להציג אותם בצורה שתהיה ברורה, אסתטית ומעבירה את המסר בצורה אפקטיבית זה אתגר בפני עצמו. לא תמיד היה קל לבחור את סוג הגרף המתאים ביותר, או לעצב אותו כך שיהיה קריא.
* **איך התמודדנו:** כדי לשמור על אחידות וקלות עבודה, הגדרנו מראש קבועים לעיצוב - צבעים, גופנים, גדלים - והשתמשנו בהם באופן עקבי בכל הגרפים. יצרנו גם פונקציות קטנות שעוזרות לנו ליצור גרפים נפוצים במהירות, וכך יכולנו להתנסות ולשפר את התצוגה עד שהיינו מרוצים.

***חלק שלישי : סגירת הפרויקט \_( 20 נקודות)***

*תיק למתכנת:* ***(ליאור)***

תיעוד פרויקט - המערכת שלנו

**קבצים:**

כל הקוד נמצא בתוך מחברת Google Colab בשם:

HW3\_WOLF.ipynb

המחברת כוללת את כל הרכיבים: סריקה, אינדוקס, חיפוש, עיבוד טקסט וסטטיסטיקות מילים.

**רכיבים ופונקציות עיקריות:**

**סורק אתרים ואינדיקס:**

* **:fetch\_page\_text(url)**  מוריד את תוכן ה-HTML של כתובת האתר

שניתנה.

* **:extract\_links(self, base\_url, soup)**הפונקציה מחלצת קישורים פנימיים מעמוד אינטרנט, מוודאת שהם שייכים לאותו דומיין ומסירה מזהים פנימיים כמו #. היא ממירה קישורים יחסיים לאבסולוטיים, מסננת כתובות שאינן תקינות, ושומרת קישורים ייחודיים בקבוצה. לבסוף, היא מחזירה את רשימת הקישורים התקפים לשימוש בסריקה נוספת.
* **:extract\_links(self, base\_url, soup)**הפונקציה מחלצת קישורים פנימיים מעמוד אינטרנט, מוודאת שהם שייכים לאותו דומיין ומסירה מזהים פנימיים כמו #. היא ממירה קישורים יחסיים לאבסולוטיים, מסננת כתובות שאינן תקינות, ושומרת קישורים ייחודיים בקבוצה. לבסוף, היא מחזירה את רשימת הקישורים התקפים לשימוש בסריקה נוספת.

**search - הפעלת מנוע החיפוש :**

* **perform\_search:** הפונקציה מחפשת מונח שהוזן על ידי המשתמש במסד נתונים, מזהה את הגרסה המותאמת שלו, ובודקת האם הוא מופיע באינדקס, תוך הצגת מידע על מספר הפעמים שנמצא וקישורים רלוונטיים.
* **: handle\_search** הפונקציה מטפלת בלחיצה על כפתור חיפוש, שולפת את הערך שהוזן על ידי המשתמש, מחפשת תוצאות באמצעות perform\_search, מנקה את התצוגה הקודמת ומציגה את התוצאות החדשות בפורמט HTML.

**:Sensor Data**

* **: generate\_sensor\_statistics**הפונקציה יוצרת תצוגה HTML המציגה את הנתונים האחרונים של חיישן מסוים יחד עם חישובים סטטיסטיים של

נתונים היסטוריים מאותו חיישן ב-5 הדקות האחרונות.

* **: update\_sensor\_data\_and\_plot**הפונקציה מעדכנת את תצוגת הסטטיסטיקות והגרפים בהתאם לחיישן ולמאפיין שנבחרו, טוענת נתונים חדשים, מציגה סטטיסטיקות מעודכנות, ומייצרת גרף היסטורי חדש.

* **: update\_sensor\_status**הפונקציה מעדכנת את תצוגת מצב החיישנים במסך הניהול, בודקת זמינות נתונים מהחיישנים, ומציגה את הסטטוס שלהם בהתאם.
* **: on\_sensor\_selector\_change**הפונקציה מעדכנת את אפשרויות הגרף בהתאם לחיישן שנבחר, מוודאת שהמאפיין שנבחר תקף, ואז מעדכנת את הסטטיסטיקות והגרף בהתאם.

**ממשק משתמש – :UI**

* **: create\_html\_widget** יוצרת רכיב HTML מותאם אישית להצגת תוכן בפורמט דינמי עם אפשרויות עיצוב.
* **: create\_button**יוצרת כפתור אינטראקטיבי עם תיאור, סגנון ועיצוב מותאם אישית לממשק המשתמש.
* **: create\_text\_input**יוצרת שדה קלט טקסט אינטראקטיבי עם תיאור והגדרות עיצוב מותאמות אישית.
* **: create\_output**יוצרת רכיב פלט דינמי לתצוגת מידע מתעדכן בממשק.
* **: create\_dropdown**יוצרת רשימה נפתחת עם אפשרויות בחירה ותיאור, תוך שימוש בהגדרות עיצוב מותאמות.
* **: create\_vbox**מארגנת ומציגה רכיבים בתיבה אנכית, תוך התאמה אישית של הסגנון והעיצוב.
* **: create\_hbox**מארגנת ומציגה רכיבים בתיבה אופקית, עם אפשרויות עיצוב מותאמות.

**תבניות עיצוב:**

השתמשנו בתבניות עיצוב של wrapper, singleton ו- factory

**קטעי קוד מעניינים:**

1. - Anomaly detectorהפונקציה בודקת ערכי חיישנים ומזהה חריגות על ידי השוואתם לטווחים נורמליים מוגדרים מראש, אם הערך מחוץ לטווח היא מוסיפה הודעה לרשימת החריגות.

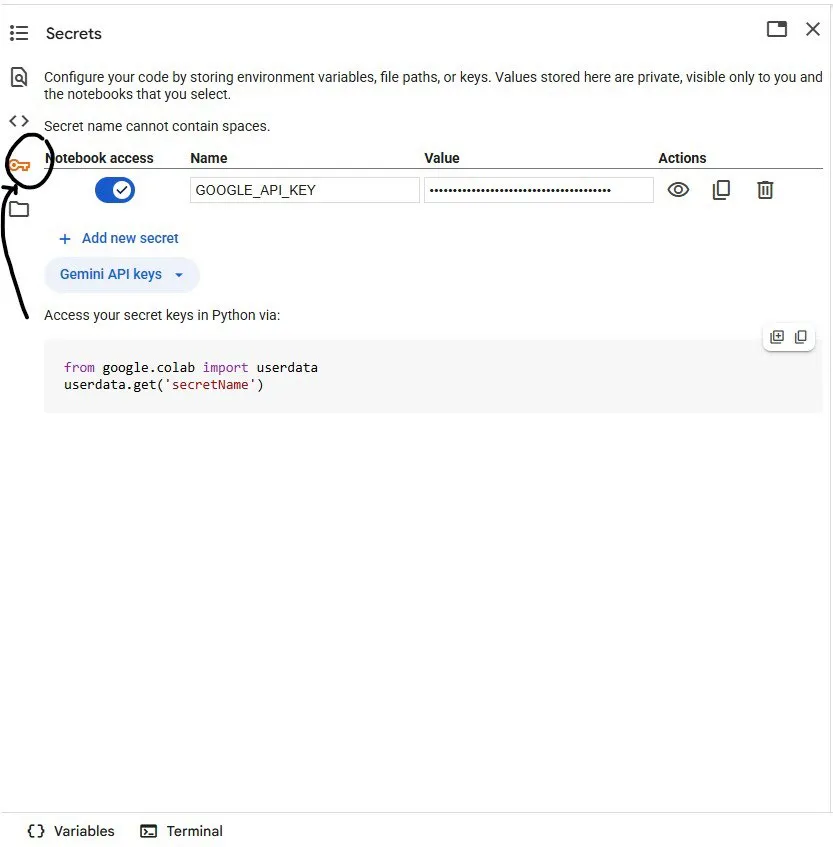


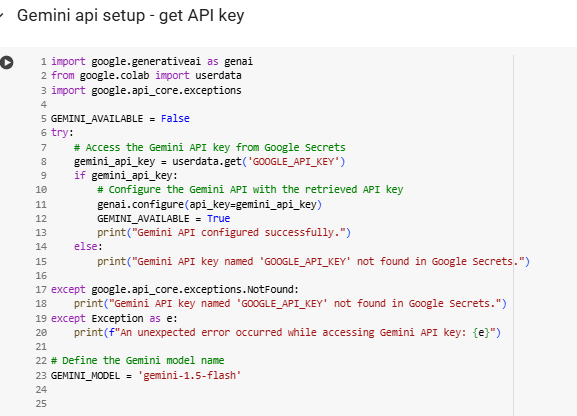
1. Chatbot

בתוכנה יש להטמיע מפתח API עבור 'gemini-1.5-flash'

את המפתח יש להפיק מהקישור הבא <https://aistudio.google.com/prompts/new_chat>

ולהכניס אותו תחת google secrets בשם google\_API\_key





A screen shot of a computer

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

A white screen with blue and purple lines

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

בחלק זה תכינו מסמכים המתארים את הפרויקט שלכם:

1. יש לבנות תיק למשתמש , הכולל הסבר כללי על המערכת , פירוט מסכים, מעברים בין מסכים והסבר על טעויות אפשרויות.
2. התייחסו בתיק המשתמש לשקיפות אלגוריתמית – כיצד הבהרתם למשתמשים את האלגוריתמים והנתונים הנאספים בקוד שלכם?
3. יש להכין סרטון קצר של 30-60 שניות, המתאר את השימוש במערכת. הסרטון משמש כ – elevator pitch  למערכת שלכם, כלומר יש לכלול בו הסבר מקצועי ועם זאת שיווקי , המדגיש את האלמנטים המיוחדים של המערכת שבניתם. יש להגיש את הסרטון בפורמט mp4.

הוראות הגשה:

1. יש להגיש את התרגיל בצוותים, בתיקיית ה –GIT שלכם, וכן בתיקייית התרגיל ב moodle. **חובה לכלול קישור לתיקיית הגיט בקובץ במוודל.**
2. יש להגיש במוודל קובץ זיפ הכולל קובץ וורד ובו מענה לשאלות, וקישור ל- notebook ובו הקוד שלכם (יש לוודא שהקישור פומבי ונגיש).
3. כותרתו של הקובץ תהיה HW3\_TEAMNAME

בהצלחה!