**מבוא למחשוב ענן - סמסטר אביב התשפ"ה**

**תרגיל בית 2** – **עבודה בצוותי העבודה**

מועד הגשה: 28.5.25

*חלק ראשון : בניית מערכת (10 נקודות)*

קישור ל-notebook בקולאב: [HOMEWORK 2.ipynb](https://colab.research.google.com/drive/1n3a1p6eDCiSZaQiaR5192K5O9SVJrqtn#scrollTo=ysC0U6dpWAxq)

קישור לגיטהאב: <https://github.com/Serji0998/Parrot.git>

*בכל צוות על כל אחד לבחור אחד מהתפקידים הבאים (יש להחליף מתרגיל בית 1 ) (10 נקודות)*

*scrum master -מרכז את העבודה*

*,frontend developer – פיתוח החלק האחראי על הצגה ללקוח.*

*,backend developer – פיתוח מסד הנתונים והעבודה מולו.*

*product manager – ייצוג הלקוח בצוות (בהתאם לחשיבה העיצובית שבוצעה).*

*UI – עיצוב הממשק*

*QA – בדיקות התוכנה*

נא לרשום את שם הסטודנט.ית בתרגיל זה. על מהנדס.ת המערכת לכתוב כיצד נעשתה חלוקת העבודה מול הצוות, מה היו המשימות של כל חבר צוות, האם היה ממשק בין חברי הצוות, והאם המשימות מולאו:

| **איטרציה 1** | | |
| --- | --- | --- |
| **שם חבר הצוות ותפקיד בתרגיל זה** | **משימות שהוקצו** | **משימות שהושלמו** |
| סיראג ח'וטבא- QA | בניית אינדקס, מסך רביעי- פיצ'ר | כל המשימות הושלמו |
| לאון פלדמן- backend developer | מסך מנהל למערכת | כל המשימות הושלמו |
| לואי ח'דירי- UI | מסך סטטיסטיקות | כל המשימות הושלמו |
| חלא קאדרי- frontend developer | חלק 2- סעיפים 1-5 לאחר ההצגה | כל המשימות הושלמו |
| מור הודיה ממן- scrum master | מסך שאילתא למנוע החיפוש (הזנה והצגת תוצאות) | כל המשימות הושלמו |
| רזאן ח'טיב- product manager | מסך רביעי- פיצ'ר | כל המשימות הושלמו |

בניית אינדקס (20 נקודות)

באיטרציה זו עליכם לבנות את מסד הנתונים שמכיל את האינדקס של המילים המשמעותיות באתר [mqtt.org](http://mqtt.org), הפרוטוקול איתו אנו עובדים לקבלת מידע מהחיישנים.

מבנה האינדקס צריך להיות אחיד לכל הקבוצות , ולכלול **לפחות** את השדות הבאים  **( אין לשנות את שמות השדות!)**:

| שם השדה | הסבר |
| --- | --- |
| term | term |
| DocIDs | רשימת קישורים לדפים המכילים את ה- term, ממוספרים לפי בחירתכם |

משימות:

1. ממשו את האינדקס בקולאב.  
2.רשמו בצורה מפורשת את רשימת ה stop words שבחרתם, ונמקו מדוע בחרתם במילים אלו.

**{'a', 'an', 'the', 'and', 'or','in', 'on', 'at', 'to', 'for', 'of', 'with','is', 'this', 'that', 'it', 'as', 'by','mqtt', 'org','i', 'you','s'}**

**הפחתת רעש (Noise):** המילים הללו מופיעות כמעט בכל טקסט באנגלית, אך לא מוסיפות מידע משמעותי. השארתן באינדקס הייתה יוצרת עומס מיותר ומקטינה את הדיוק של החיפוש. מילה Steve's הופיעה הרבה ולכן הוספנו אות s.

**מיקוד במילים משמעותיות:** על ידי סינון מילות עצירה, נוכל להתמקד במילים שבאמת נושאות מידע חשוב – כמו מושגים, שמות רכיבים, פעולות או תכונות.

**התאמה לתוכן האתר:** נוספו מילים כמו "mqtt" ו־"org" כי מדובר באתר שמכיל אותן בתדירות גבוהה בכל עמוד, ולכן הן לא מייצגות תוכן שונה בין עמודים.

3. ציינו האם השתמשתם ב stem/lemmatization לצורך בניית האינדקס.

**במהלך תהליך בניית האינדקס, השתמשנו ב־Porter Stemmer מספריית nltk, שהיא אחת השיטות הפופולריות ביותר ל־Stemming בשפה האנגלית.**

**דוגמה לפעולתו:**

* devices → devic
* connecting → connect
* messages → messag

*חלק שני: בניית מסכים להצגה בכיתה (50 נקודות)*

בחלק זה תכינו 4 מסכים, אותם תציגו לחבריכם בפעילות שתתבצע בכיתה.

המסכים צריכים לכלול (לפחות) (20 נקודות):

מסך מנהל למערכת, מסך שאילתא למנוע החיפוש (הזנה והצגת תוצאות), מסך סטטיסטיקות מעניינות.

בשלב זה נדרש לממש במלואם את בניית מסך המנהל ומסך השאילתא. ניתן את התוצאות להציג כרגע עם data חלקי.מומלץ לממש ככל הניתן גם את מנוע החיפוש , כפי שלמדתם בתרגול 6 😊

את מסך הסטטיסטיקות עליכם לממש בצורה בסיסית. עם זאת כמובן ,שכל תוספת שתחליטו עליה, תוביל להערכה גבוהה יותר של המשימה.

בשבוע ההרצאות של 19-20.5.25 תציגו את המערכות שבניתם. המפגש יתנהל במתכונת סטודיו – כל צוות מגיעה **במלואו** לאחד המועדים עם לפטופ, כל הצותים מציגים במקביל. הסטודנטים מסתובבים בין הצוותים, **מתנסים** במערכת,וממלאים משוב .

ההצגה ומילוי המשוב הם **חובה.**

אנא השתבצו בהקדם לאחת מקבוצות ההרצאה על מנת לוודא שקיים איזון בין הצוותים:

https://docs.google.com/spreadsheets/d/1-grJWvFQGtMkCJ8lg9v9ZSYULBvA4ixaEskBCZglIiE/edit?gid=0#gid=0

לאחר ההצגה תקבלו באופן אנונימי את המשובים של חבריכם, וכן את המשוב שלנו.משימות:

1. התייחסו ל -8 כללי הזהב של שניידרמן (הוצגו בתרגול). כיצד המערכת שלכם מבטאת אותם? (5 נקודות)

**1.שמור על עקביות** ממשק המשתמש אחיד בכל המסכים ,אותם צבעים (ירוק = תקין, אדום = חריגה), אותם סוגי כפתורים והצגת גרפים עקבית.

**2.אפשר קיצורי דרך למשתמשים מנוסים** משתמשים יכולים לבצע חיפוש באמצעות הקלדה ולחיצה על Enter, לעבור בין תחנות בלחיצה אחת, ולבחור משתנים ותחנות מתוך Dropdown מהיר.

**3.ספק משוב ברור ומיידי** כל פעולה מניבה תוצאה מיידית – למשל חיווי גרפי לערכים, טקסטים ברורים עם סמלים ,ותוצאות חיפוש שמופיעות מיידית עם פירוט.

**4.עצב תהליכים עם התחלה, אמצע וסיום ברור** כל מסך בממשק נבנה כתהליך שלם לדוגמה: חיפוש מתחיל מהכנסת שאילתה, קבלת תוצאה, ופירוט המסמכים.

**5.הצע טיפול פשוט בשגיאות** המערכת בודקת תקינות של ערכים פיזיים בזמן אמת, מציגה חיווי על חריגות, ומונעת קלט ידני שגוי.

**6.אפשר ביטול או שינוי של פעולה** המשתמש יכול תמיד לחזור ולבחור תחנה או משתנה אחר בלי לרענן או להתחיל מחדש, הכל בלחיצה פשוטה.

**7.תן למשתמש שליטה על מה שקורה** המשתמש שולט בכל שלב , מתי לעבור תחנה, איזה משתנה לבחור, איזה סוג תצוגה (ממוצע/גרף), ואפילו מצב דיבאג.

**8.הפחת עומס על הזיכרון של המשתמש** אין צורך לזכור ערכים או שמות ,כל הבחירות מוצגות בתפריטים ברורים, הכפתורים ברורים.

1. יש להגיש את הטבלה הבאה , תוך התיחסות למשובים שקיבלתם (5 נקודות):

| הערת משוב | האם התבצע שינוי באפליקציה בעקבות ההערה? | נימוק |
| --- | --- | --- |
| מסך סטטיסטיקות- המשתמשים רצו לראות התפתחות לאורך זמן, לא רק ערכים עכשוויים | השינוי: הוספה של טווח לאורך זמן בסטטיסטיקות. | שינינו את הכפתור השעתי והיומי לשלושה כפתורים של שעה,יום וחודש |
| כאשר עוברים בסטטיסטיקות צריך כל פעם לטעון מחדש, במקום שיטען אוטומטית לבד במעברים בין מסך למסך | השינוי: טעינה אוטומטית של מסך הסטטיסטיקות. | מחקנו את הכפתור "תראה נתונים" ועכשיו זה טוען אוטומטית כאשר משתנה בקשה לנתון כלשהו בגרף |
| חלק מהמשתמשים חשו שהעיצוב פשוט מדי | שינוי בעיצוב המסכים הכללי | שינינו את העיצוב הכללי של המסך הראשי. כעת הכל הרבה יותר נוח למשתמש ונעים לעין. |
| מסך סטטיסטיקות- להוסיף אפשרות לבחור טווח זמן לצפייה בגרף | השינוי: הוספת בחירת טווח זמן לצפייה בגרף | הוספנו בחירת טווח זמן לצפייה בגרף. |
| הגרפים היו ברורים ועזרו להבין את הנתונים | אין שינוי | הערה לשימור, לכן אין מה לשנות |

1. יש לרשום את ציון ה SUS של המערכת שלכם.מה מעיד הציון?(5 נקודות)

הציון הוא 72.

**המשמעות היא שהמערכת:**

שימושית ונוחה לרוב המשתמשים

נחשבת ברורה ואינטואיטיבית

יכולה להמשיך להשתפר בעיצוב וחווית משתמש, אך בבסיסה היא מצליחה להעביר חוויית שימוש טובה

1. הגדירו 3 מדדים להצלחת המערכת (הרצאה 3).(5 נקודות)

| מדד | הסבר |
| --- | --- |
| שביעות רצון המשתמשים | נאסף דרך שאלון SUS ומשוב איכותני. מדד זה בודק האם המשתמשים נהנים מהשימוש במערכת, מרגישים שהיא נוחה, אינטואיטיבית ושימושית. הציון שקיבלנו – 72 – מעיד על שימושיות טובה |
| זיהוי חריגות מדויק | מדד טכני שנבדק לפי כמה פעמים המערכת הצליחה לזהות ערכים פיזיים מחוץ לטווח התקין (טמפ’, לחות, לחץ) ולסמן אותם כחריגים בצורה ברורה. |
| זמן תגובה ומעבר בין מסכים | נמדד לפי כמה מהר המשתמש מקבל תוצאה לאחר פעולה (כמו בחירת תחנה, חיפוש מונח, מעבר תצוגה). ככל שהזמן קצר יותר כך המערכת יעילה יותר. המשתמשים ציינו לטובה את התצוגה בזמן אמת. |

1. הציגו דיאגרמת ארכיטקטורה של המערכת שלכם. הסבירו באיזה סוג ארכיטקטורה השתמשתם (הצאה 7), ופרטו את חלקי הקוד המתייחסים לכל חלק בארכיטקטורה.(10 נקודות)

### 

### סוג הארכיטקטורה:

המערכת שלנו בנויה משילוב של שלוש ארכיטקטורות עיקריות:

1. **Layered Architecture** – המבנה מחולק לשכבת תצוגה (ipywidgets), לוגיקה (פונקציות פייתון), נתונים (Firebase), ומקור מידע (חיישנים מדומים).
2. **Pipe & Filter Architecture** – המידע זורם כצינור בין רכיבים
3. **Serverless Architecture** – המערכת רצה בסביבת Google Colab ו־Firebase, ללא שרת עצמאי.

כל שכבה ממומשת בקוד: ממשק המשתמש מבוסס ipywidgets, הלוגיקה כוללת פונקציות כמו run\_query() ו־generate\_reading(), הנתונים נשמרים ב־Firebase באמצעות get()/post()/put(), והחיישנים המדומים נבנים בפונקציית generate\_reading().

*חלק שלישי : פיצ'ר לבחירתכם* (10 נקודות)

הוסיפו פיצ'ר מעניין למערכת, אשר לא נדרש מכם, לבחירתכם.שימו לב - הכוונה לפיצ'ר פונקציונאלי (ולא עיצובי, או שימוש בשרת כדי להעלות לענן את האתר).

כתבו מספר משפטים להסבר התוספת, וציינו היכן בקוד הוא ממומש. כמו כן הסבירו כיצד הוא מתבטא בחלק המוצג למנהל.

ـــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــــ

**פיצ’ר נוסף: ניטור סביבתי חכם והתרעות בזמן אמת**

**מטרת הפיצ’ר:**

במערכות ייצור מדויקות (כמו פס ייצור של רכיבים אלקטרוניים או מערכות פנאומטיות), תנאי סביבה יציבים הם קריטיים.

לכן, המערכת פותחה כך שתבצע ניטור רציף של תנאי סביבה – טמפרטורה, לחות ולחץ – ותזהה מיידית חריגות מהטווחים התקינים.

1**. איסוף נתונים:**

המערכת קוראת ערכים היסטוריים (או חיישנים חיים בעתיד) של:

**🌡טמפרטורה**

**💧 לחות**

**🧭 לחץ**

2**. בדיקת תקינות אוטומטית:**

כל ערך נבדק מול טווחים מוגדרים מראש:

* טמפרטורה: 15°C – 30°C
* לחות: 30% – 60%
* לחץ: 980 – 1050 hPa

אם הערך חורג, תופיע התראה מיידית

**3. שמירת היסטוריה:**

כל קריאה נשמרת עם חותמת זמן (`timestamp`) – דבר שמאפשר **מעקב גרפי לאורך זמן**.

איך זה מוצג למנהל הדשבורד (UI/UX):

3 גרפים דינמיים (טמפרטורה / לחות / לחץ לאורך זמן)

כל ערך צבוע לפי מצב:

✅ **ירוק**: ערך תקין

🚨 **אדום**: ערך חריג או מסוכן

אפשרות לסנן לפי תקופת זמן או אירועים חריגים.

**ערך מוסף :**

* **תחזוקה מונעת**: זיהוי מראש לפני שנגרם נזק.
* **צמצום עצירות פס ייצור**.
* **חיסכון כספי**: פחות תקלות ופגמים במוצרים.
* **שיפור אמינות ויציבות המערכת**.
* **עזרה למהנדסים**: הצגה ברורה של הבעיה, באיזו תחנה ומתי.

המשך פיתוח אפשרי:

* שליחת התראה באימייל או SMS למהנדסים בעת חריגה
* שמירת לוג אוטומטי של התראות
* ממשק ניהולי אינטראקטיבי (עם סינון לפי תחום חריגה או חיישן)

**הוראות הגשה:**

1.ש להגיש במודל קובץ זיפ הכולל קובץ וורד ובו מענה לשאלות, וקישור ל- notebook ובו הקוד שלכם (יש לוודא שהקישור פומבי ונגיש). **אין לבצע שינויים במחברת לאחר ההגשה!**

**2.** הקוד צריך לרוץ במלואו מהמחברת בלבד. לא יתקבלו הגשות הכוללות הרצה באתר חיצוני (בפרט slack), או צורך להעלות קבצים למחברת על מנת שתרוץ. הגשות כאלו יקבלו ציון אפס על מרכיב הקוד

3.יש להגיש את התרגיל בצוותים, בתיקיית ה –GIT שלכם (צרפו קישור), וכן בתיקייית התרגיל ב moodle.כותרתו של הקובץ תהיה HW2\_TEAMNAME

4. שימו לב כי כל העבודות חייבות להיות שונות זו מזו. עבודות שייראו דומות ייפסלו ויינתן עליהן ציון 0.

בהצלחה!