**המחלקה להנדסת תוכנה - קורס מחשוב ענן**

**פרויקט סופי**



**מגישים:**

עובייד חדאד - 206989295

מוראד עסאקלה - 315688069

שאדי אבוריש - 208658898

אניל סלאמה - 316198522

ראיד חדאד - 207075383

[קישור לתיקיית ה GitHub](https://github.com/RaedHadad/Elephant)

[קישור ל Google Collab](https://colab.research.google.com/drive/1yLP_-KlpYJ-fEZDPG4vDyM0Z3-DZaAFS?usp=sharing)

1. **מהות המוצר ומרכיביו – מודולים ופיצ'רים מעניינים :**

המוצר שלנו הוא מערכת לניהול פרויקטים מה OnShapeהמתבססת על נתונים בזמן אמת שנשמרים ב Firebase, המערכת מאפשרת למשתמשים להעלות קבצי JSON המכילים נתוני פעולות שונות, לבצע חיפושים וסינונים על בסיס הנתונים, להפיק דוחות PDF ולהציג גרפים, ולבסוף גם לנהל צ'אט עם עוזר וירטואלי שמספק מידע רלוונטי על המידע.

**מרכיבי המוצר:**

1. מודול התחברות למערכת :

* על מנת להתחבר למערכת צריך להתחבר בעזרת שם משתמש וסיסמה ששמורים במערכת.

1. מודול העלאת נתונים:

* משתמשים יכולים להעלות קבצי JSON למערכת, והנתונים נשמרים ב Firebase.

1. מודול חיפוש וסינון נתונים :

* מאפשר למשתמשים לחפש ולסנן נתונים על פי תאריכים, משתמשים, כרטיסיות ותיאורים וכו'.

1. מודול צ'אט עם עוזר וירטואלי:

* המערכת כוללת עוזר וירטואלי המאפשר למשתמשים לשאול שאלות ולקבל מידע על הפרויקט בצורה אינטראקטיבית.

**פיצ'רים מעניינים :**

1. פיצ'ר Best Performer ::

בהתבסס על הנתונים שהמערכת אוספת מ-JSON, פיתחנו פיצ'ר ייחודי שמחשיב את המשתמש המוביל (Best Performer) על פי משקל של סוגי הפעולות שבוצעו.

לדוגמה:

* יצירת מסמך מקבלת ניקוד גבוה (5 נקודות).
* הוספה או עדכון של נתונים מקבל ניקוד בינוני (2-3 נקודות).
* מחיקה מקבלת ניקוד שלילי (-1 נקודה).

הפיצ'ר מחשב את סך הנקודות שהמשתמש צבר ובוחר את המשתמש המוביל על פי סכום הנקודות הגבוה ביותר.

בנספח א' של תרגיל בית 3, החישוב ל-Best Performer הוא בדיוק הפונקציה שתיארנו שמחשבת את הנקודות לפי סוגי הפעולות ומציגה את המשתמש המוביל על פי תוצאות החישוב הללו.

החלטנו שהפיצ'ר הזה ימומש בצ'אט בוט בכך שהמשתמש יכול לשאול את הבוט מי המשתמש המוביל והבוט מחשב את זה ונותן שם מדויק עם הניקוד של אותו משתמש.

אלה הקריטריונים שבחרנו לבחירת משתמש המוביל הקריטריונים אלה יכולים להשתנות בכל עת.

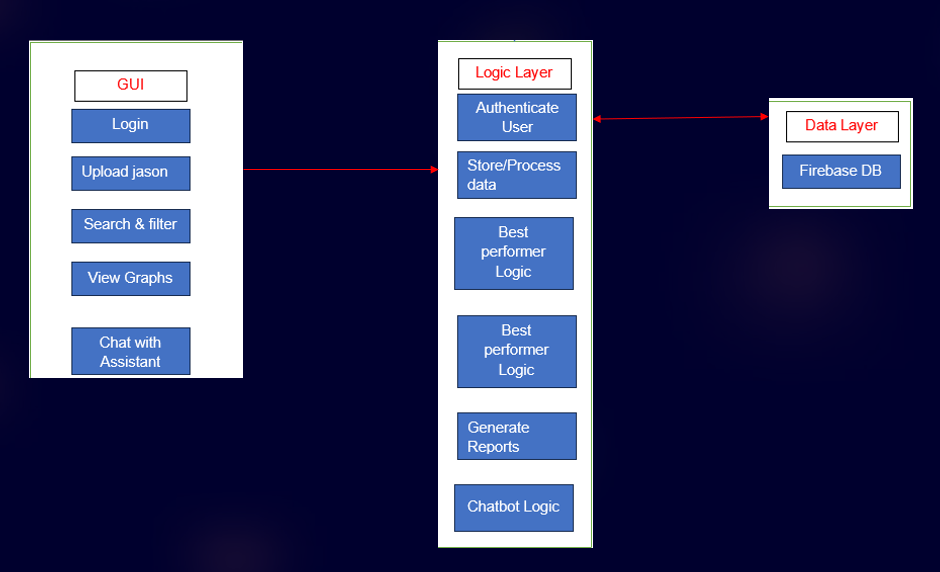
1. אפשרות הורדת ממידע מסונן כקובץ PDF :

פיצ'ר נוסף שחשבנו שיעניין את המשתמש הוא האפשרת להכין עבורו את המידע שביקש לתוך טבלה מסודרת ולהוריד כקובץ PDF . כך יוכל לשתף את המידע בצורה נוחה , וגם לשמור לעצמו לשימוש עתידי במקרה של תקלה טכנית במערכת / תקלת חיבור לרשת או כל תרחיש אפשרי אחר .

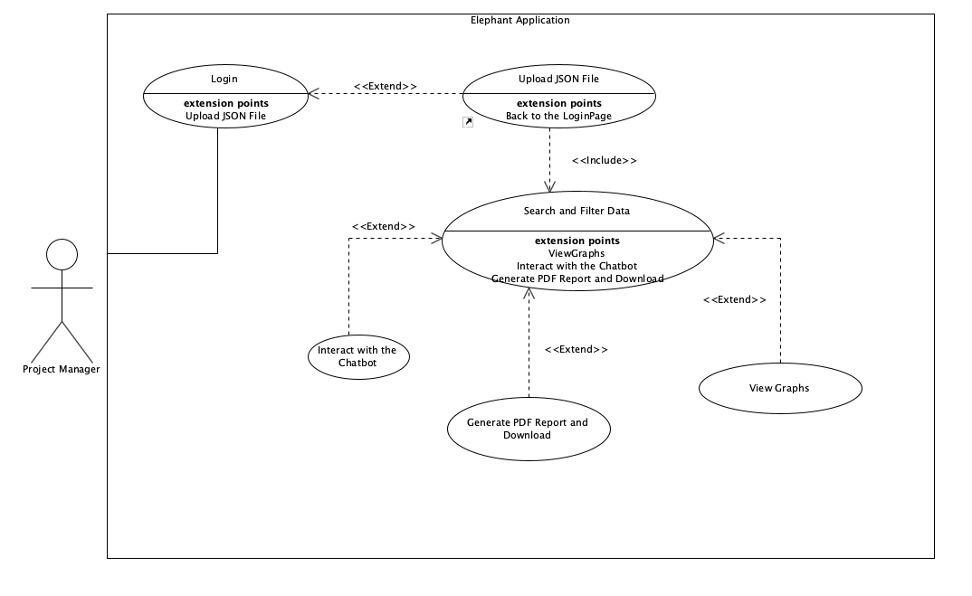
1. הצגת גרפים:

מאפשר למשתמש להפיק גרפים המתאימים לחיפוש שביצע , הגרפים כוללים ניתוח של כמות פעולות לפי משתמש, כרטיסייה, מסמך ועוד, ומציגים את ההתפלגות בזמן.

1. **ארכיטקטורת המערכת** :



1. **USE CASE :**



1. **דרישות לא פונקציונלית :**

**Usability :**

**ממשק משתמש**: המערכת מכילה ממשק משתמש אינטואיטיבי וידידותי למשתמש, כך שיהיה קל למשתמשים לנווט ולהשתמש בתכונות השונות ללא צורך בהכשרה נרחבת.

**Performance :**

**זמן תגובה**: המערכת מגיבה לפעולות משתמש בתוך 2 שניות, כולל טעינת דפים, סינון נתונים ויצירת קבצי PDF.

**Data integrity :**

**ניהול שגיאות**: המערכת מתמודדת עם שגיאות ומתעדת אותן, כגון חיבורים שנכשלו ל-Firebase או נתונים חסרים, מבלי לקרוס.

**עקביות נתונים**: הנתונים המתקבלים מ-Firebase וקלטי המשתמש צריכים תמיד להיות עקביים בכל רכיבי המערכת השונים.

**Security :**

**אימות**: רק משתמשים מורשים מסוגלים להיכנס ולגשת למערכת.

**Maintainability :**

**מודולריות**: המערכת מודולרית, כך שניתן יהיה לעדכן רכיבים שונים (כגון צ'אטבוט, יצירת PDF) באופן עצמאי מבלי להשפיע על שאר המערכת.

**איכות קוד**: הקוד מתועד היטב, עם שמות משתנים ברורים והערות, כדי להבטיח תחזוקה ופיתוח עתידי קלים.

**Portability :**

**תאימות לדפדפנים**: המערכת תאימה לדפדפני אינטרנט מרכזיים (Chrome, Firefox, Safari, Edge) ללא צורך בקונפיגורציות מיוחדות.

**Supportability :**

**ערוצי תמיכה**: למשתמשים יש ערוצי תמיכה (כגון דוא"ל, צ'אט) כדי לקבל עזרה עם בעיות או שאלות במערכת.

1. **תיק תחזוקה למערכת לניהול פרויקטים**

**קבצים מרכזיים:**

* **Elephant Final Project.ipynb** : קובץ הקוד הראשי שמכיל את כל הפונקציות והלוגיקה של המערכת, כולל ממשק המשתמש, אינטגרציה עם Firebase לוגיקת הצ'אטבוט והפונקציות הנוספות כמו יצירת PDF והצגת גרפים.

**אובייקטים מרכזיים:**

* **Fbconn:** אובייקט שמנהל את החיבור ל-Firebase זהו requests.Session() שמשמש לשליחת בקשות HTTP למסד הנתונים של Firebase
* **uploaded\_data**: משתנה גלובלי שמחזיק את הנתונים שהועלו על ידי המשתמש.
* **filtered\_df**: מחזיק את תוצאות החיפוש המסוננות.
* **Chatbot**: אובייקט מסוג Chat שמנהל את האינטראקציה עם המשתמשים בצ'אט.

**תיעוד קצר של כל פונקציה:**

1. **fetch\_data\_from\_firebase()**
   * **תיאור:** פונקציה זו מבצעת בקשה ל-Firebase כדי לאחזר את כל הנתונים המאוחסנים במסד הנתונים. הנתונים מוחזרים בצורת JSON.
   * **קלט:** אין.
   * **פלט:** מחזירה את הנתונים שהתקבלו מ-Firebase בצורת JSON.
2. **initialize\_chatbot()**
   * **תיאור:** הפונקציה מאתחלת את הצ'אטבוט על בסיס הנתונים שהתקבלו מFirebase הפונקציה מחשבת את המשתמש המצטיין על פי הלוגיקה המוגדרת(יתואר בהמשך), ומבצעת תהליך עיבוד נתונים ראשוני.
   * **קלט:** אין.
   * **פלט:** מאתחלת את אובייקט הצ'אטבוט.
3. **calculate\_user\_scores(data)**
   * **תיאור:** הפונקציה מחשבת את הניקוד של כל משתמש על בסיס סוגי הפעולות שביצע, על פי הקריטריונים שהוגדרו.
   * **קלט:** data רשימת פעולות משתמשים.
   * **פלט:** מחזירה מילון עם המשתמשים והניקוד שלהם.
4. **display\_login\_page()**
   * **תיאור:** מציגה את עמוד ההתחברות של המערכת. עמוד זה כולל טופס להזנת שם משתמש וסיסמא.
   * **קלט:** אין.
   * **פלט:** מציגה את עמוד ההתחברות.
5. **display\_upload\_page()**
   * **תיאור:** מציגה את עמוד העלאת הקבצים למערכת. משתמשים יכולים להעלות קובץ JSON שמכיל את הנתונים שלהם. הפונקציה כוללת גם בדיקה לפורמט הקובץ.
   * **קלט:** אין.
   * **פלט:** מציגה את עמוד העלאת הקבצים.
6. **display\_search\_page()**
   * **תיאור:** מציגה את עמוד החיפוש והסינון, שבו המשתמש יכול לבצע חיפושים בנתונים לפי קריטריונים שונים כמו תאריכים, משתמשים, כרטיסיות ותיאורים.
   * **קלט:** אין.
   * **פלט:** מציגה את עמוד החיפוש והסינון.
7. **filter\_data(change)**
   * **תיאור:** פונקציה זו מסננת את הנתונים על בסיס הקריטריונים שהמשתמש הזין בעמוד החיפוש (תאריכים, כרטיסיות, משתמשים ותיאורים).
   * **קלט:**  change משתנה המעיד על שינוי בשדה קלט כלשהו.
   * **פלט:** מציגה את התוצאות המסוננות בטבלה.
8. **display\_search\_page\_elements(no\_results\_message)**
   * **תיאור:** מציגה את האלמנטים של עמוד החיפוש, כולל טפסי החיפוש, כפתורים והודעות.
   * **קלט:** no\_results\_message - הודעה המוצגת אם אין תוצאות חיפוש.
   * **פלט:** מציגה את האלמנטים בעמוד החיפוש.
9. **display\_table(dataframe)**
   * **תיאור:** מציגה את הטבלה של תוצאות החיפוש בעמוד החיפוש.
   * **קלט:** dataframe של תוצאות החיפוש.
   * **פלט:** מציגה את הטבלה על המסך.
10. **display\_paginated(dataframe, rows\_per\_page=10)**
    * **תיאור:** מציגה את תוצאות החיפוש בטבלה עם אפשרות לדפדוף בין עמודים.
    * **קלט:** dataframe - DataFrame של תוצאות החיפוש. rows\_per\_page - מספר השורות בכל עמוד.
    * **פלט:** מציגה את הטבלה עם אפשרות דפדוף.
11. **display\_graphs(dataframe)**
    * **תיאור:** מציגה גרפים המבוססים על תוצאות החיפוש (התפלגות לפי משתמש, כרטיסייה, מסמך, זמן ועוד).
    * **קלט:** dataframe - DataFrame של תוצאות החיפוש.
    * **פלט:** מציגה את הגרפים.
12. **generate\_pdf(dataframe, filename='filtered\_data.pdf')**
    * **תיאור:** מייצרת דוח PDF המבוסס על תוצאות החיפוש.
    * **קלט:** dataframe - DataFrame של תוצאות החיפוש. filename - שם הקובץ של ה-PDF שנוצר.
    * **פלט:** יוצר קובץ PDF ומוריד אותו אוטומטית.
13. **display\_chatbot\_page()**
    * **תיאור:** מציגה את עמוד הצ'אט עם העוזר הווירטואלי. המשתמש יכול לשאול שאלות ולקבל תשובות על הפרויקט.
    * **קלט:** אין.
    * **פלט:** מציגה את עמוד הצ'אט.
14. **ביקורת עמיתים ממפגש סטודיו :**

**יש להגיש את הטבלה הבאה , תוך התייחסות למשובים שקיבלתם:**

|  |  |  |
| --- | --- | --- |
| הערת משוב | האם התבצע שינוי באפליקציה בעקבות ההערה? | נימוק |
| להוסיף dark mode | לא | התכונה נמצאת בבדיקה לפיתוח עתידי, אך כרגע אינה בעדיפות ראשונה , בשלב הזה העדפנו להשקיע זמן ומאמצים בפיצ'רים שימושיים יותר עבור המשתמש כפי שמתואר בחלק 3. |
| להוסיף צבעים ועיצוב גרפי למערכת | כן | עיצבנו מחדש את המערכת ודאגנו לממשק גרפי נוח , ידידותי למשתמש ועשיר בצבעים. |
| מימוש גרפים בצורה מסודרת יותר | כן | הגרפים במערכת שודרגו כדי להיות ברורים יותר. |
| הבנתי איך האתר עובד  ואהבתי את שיטת העבודה | לא | אין שינוי נדרש, ההבנה הייתה חיובית. |
| קצת מתוסבך ולא פשוט לשימוש | כן | המערכת הייתה בשלב אב טיפוס ראשוני , בעקבות ההערות דאגנו לממשק פשוט להבנה עם שלבי עבודה ברורים לנוחיות המשתמש. |

**יש לרשום את ציון ה SUS של המערכת שלכם :**

SUS Score = 70.14

בעקבות מפגש הסטודיו, שיפרנו את הממשק באמצעות שינויים בעיצוב, הוספת הסברים ברורים יותר וביצוע התאמות לממשק המשתמש כדי להקל על הניווט והשימוש. בנוסף, לאחר ביצוע השינויים, חישבנו את ציון ה-SUS (System Usability Scale) וקיבלנו ציון שמצביע על כך שהמערכת נחשבת קלה לשימוש ומספקת חוויית משתמש חיובית.

1. **אתגרים אשר עלו במהלך העבודה :**

**במהלך העבודה על הפרויקט התמודדנו עם מספר אתגרים משמעותיים:**

1. **תקופת מלחמה:** המצב הביטחוני המתוח השפיע על יכולת הריכוז והפרודוקטיביות של חברי הקבוצה, כאשר חלקם אף נמצאים בקו העימות. המצב דרש התאמות וגישות עבודה גמישות על מנת לעמוד בלוחות הזמנים שהוגדרו מראש, תוך מתן תמיכה הדדית ושמירה על הבריאות הנפשית והפיזית של חברי הקבוצה.
2. **חוסר ב Debugger :** היעדר אפשרות לשימוש ב-Debugger בכלי העבודה Google Colab הקשה על איתור בעיות ותקלות בקוד בזמן אמת, מה שחייב אותנו להשתמש בכלים ותוכנות חיצוניים או להיעזר בהדפסות רבות בקוד, מה שגזל זמן יקר.
3. **ארגון זמן בין חברי הקבוצה:** העבודה בקבוצה דרשה תיאום מדויק של זמן העבודה בין חברי הצוות, במיוחד לאור העובדה שכל אחד נמצא במקום אחר ובעל לוח זמנים שונה. תקופת המלחמה אף היא הוסיפה על הקושי בניהול הזמן, וכך נדרש מאיתנו לפתח אסטרטגיות חדשות לניהול משאבים וזמן עבודה משותפים בצורה מיטבית.

**7.א : מטלת Design Thinking אשר ביצענו בתרגיל בית יבש 1:**

בשלב הראשון של הפרויקט, התחלנו בתהליך של Design Thinking כדי להבין טוב יותר את הצרכים והחוויות של המשתמשים שלנו. התמקדנו ביצירת פרסונות והתרחישים :

("Thinks", "Does", "Feels", "Says") כדי לתאר את מה שהמשתמש הפוטנציאלי חושב, עושה, מרגיש ואומר בנוגע למערכת שלנו. התהליך הזה עזר לנו להבין את החשיבות של יצירת כלי שמאפשר מעקב נוח וברור אחר התקדמות הפרויקטים, כמו גם את הצורך בשקיפות וביכולת מעקב אחר ביצועי הצוות.

לדוגמה:

* הבנו שהמשתמש צריך כלי שיאפשר לו לעקוב בקלות אחר התקדמות הפרויקטים ולהציג דוחות וגרפים מותאמים אישית.
* הבנו גם את החשיבות של הצגת מידע עדכני ונגישות קלה למידע, מה שמעניק למשתמש תחושת ביטחון ושקיפות.

התובנות האלו הנחו אותנו בעיצוב הממשק ובפיתוח הפתרון, כך שיתאים בצורה מיטבית לצרכים שזיהינו.

**7.ב : התייחסות למשוב שקיבלנו ב CODE REVIEW:**

|  |  |  |
| --- | --- | --- |
| הערה | תגובה | האם בכוונת הצוות לבצע שינוי בעקבות ההערה? יש לפרט. אם כן, מה השינוי. אם לא, מדוע |
| 1. התאמה בין תאריכי התחלה וסוף בדף סינון מידע , לוודא תקינות קלט שתאריך התחלה הוא לפני תאריך סיום . | הערה חשובה והסכמנו שצריך לתקן. | יהיה שינוי בקוד , נוסיף בדיקת תקינות להפרשים בין התאריכים והצגת הודעת שגיאה רלוונטית למשתמש במידה ובחר תאריך סיום שהוא קודם לתאריך ההתחלה . |
| 1. הצגת הודעה למשתמש במידה ותוצאת הסינון היא ריקה , כדי להבדיל בין תוצאה תקינה לבין שגיאת מערכת . | אפשר להסכים על הערה זו , תמיד עדיף להציג מידע על התוצאה שמתקבלת ככל שתהיה כדי לתת את האינדקציה המיטבית למשתמש . | נעשה שינוי בקוד כך שאם מתקבלת טבלה ריקה של נתונים כתוצאה מהסינון , נציג הודעה רלוונטית למשתמש . |
| 1. הקפדה על העלאת קבצים בפורמט הנכון , והצגת הודעה למשתמש על סוג קובץ ונכונותיו . | הערה מאוד חשובה , לא נרצה שהמשתמש יעלה כל קובץ שירצה (תמונה למשל) , כאן צריך להגביל אותו להעלאת קבצים בפורמט תקין (במערכת שלנו קבצים מסוג JSON) . | שינויים בקוד יתבצעו על ידי בדיקת והגבלת סוג קובץ שמעלים למערכת , והתניית מעבר לדף סינון עם העלאה של קובץ תקין , תוך כדי הצגת הודעות רלוונטיות למשתמש . |

**סיכום שינויים בקוד:**

1. **בדיקת תאריכים:**
   * הוספנו הודעה באם יש מצב שהמשתמש רושם תאריכים לא הגיוני אז מוצגת הודעת שגיאה מתאימה למשתמש.
2. **שיפור הצגת נתונים:**
   * לפני היה לנו במקרה שהטבלת החיפוש היא ריקה אז פשוט רואים מסך ריק ובכך המשתמש לא יכול להבחין אם אין מידע או יש תקלה, ועכשיו עשינו שינויים בכך אם אין תוצאות אז מתקבלת טבלה בכל זאת אבל היא ריקה.
3. **שיפור העלאת קבצים:**
   * במידה ומשתמש מעלה קובץ שהוא לא מסוג JASON אז המערכת מתריעה לו שהקובץ אינו נתמך והוא יכול להעלות קובץ חדש שמתאים.
4. **שקיפות אלגוריתמית :**

בפרויקט שלנו, הבהרנו למשתמשים את השקיפות האלגוריתמית באמצעות מספר צעדים:

**שקיפות בנתונים הנאספים**: הצגנו למשתמשים בדיוק אילו נתונים נאספים (כגון שינויים בפרויקטים, ביצועי צוותים) והבהרנו כיצד הם משמשים באלגוריתמים.

**שליטה למשתמשים**: אפשרנו למשתמשים להגדיר אילו נתונים ייאספו ואילו יוצגו.

**דוחות וגרפים**: סיפקנו ממשק גרפי עם אפשרות לניתוח שקוף של הנתונים, כדי שהמשתמשים יוכלו לראות את ההשפעה של כל פעולה בצורה ויזואלית.

**ניהול שגיאות בהעלאת לוגים**: כאשר העלאת לוג נכשלת, המערכת מציגה למשתמש הודעת שגיאה ספציפית שמסבירה את הבעיה ומציעה דרכי פתרון.

**קוד פתוח ב -GitHub**: כיוון שהמוצר שלנו הוא קוד פתוח, כל משתמש יכול לצפות בקוד המערכת המלא ולבדוק את אופן פעולתה דרך ה-GitHub שלנו.

**פרסונליזציה של הדוחות**: המשתמשים יכולים לבחור אילו סוגי נתונים להציג בדוחות שלהם, ולקבל הסבר כיצד הבחירה הזו משפיעה על הדוחות הסופיים.

**דיווחי תקינות בזמן אמת**: בזמן שהמערכת מבצעת בדיקות תקינות (כגון תאריכי התחלה וסיום לא הגיוניים), המשתמשים מקבלים משוב מידי עם הודעות שמסבירות מה לא תקין ומדוע.

השקיפות הזאת מאפשרת למשתמשים לדעת בדיוק איך המערכת פועלת ולשלוט בצורה מודעת יותר בפעולות שלהם.

1. **משוב אחרי ההצגה :**

|  |  |  |
| --- | --- | --- |
| הערת משוב | האם לדעתכם יש צורך בשינוי במערכת בעקבות ההערה? | נימוק |
| קיבלנו הערה לשפר את תצוגת הגרפים , הגרפים שמוצגים היו דחוסים ולא ברורים | כן , אנחנו חושבים שיש צורך בשיפור את אופן התצוגה , נפריד בין הגרפים למראה יותר ברור ולא נציג את כולם בבת אחת , אלא לפי בקשת המשתמש . | הצגת הגרפים היא התוצאה שהמשתמש ביקש ולכן הערות על אופן הצגתן צריך לקחת בחשבון ולהתייחס אליהן . |
| לקבוע צבע theme למערכת וללכת לפיו | לא נבצע שינוי , מימשנו את המערכת בקולאב והדגשנו על הפונקציונליות . | לא רצינו לפתח את המערכת בצד שלישי ,והעדפנו להסתפק במימוש בקולאב. |
| בחירה של כמה סטודנטים אבל לא כולם | אין צורך בשינוי , פיצ'ר זה כבר נמצא בדף הסינון . | בדף הסינון יש את האפשרות לבחור מידע עבור סטודנט ספציפי , והמערכת תסנן את המידע  בהתאם . |
| הוספת פונקציות נוספות לצ'אטבוט | כן תמיד יש מקום לשיפור ביצועי הצ'אטבוט. | הוספת מבחר הפקודות שהצ'אטבוט יתן מענה אליהן ישפר את חווית המשתמש ויקל עבורו את תהליך החיפוש. |

1. **מקורות :**

במהלך מימוש המערכת נעשה שימוש במקורות רבים לצורך פיתוח ועיבוד נתונים, כולל שימוש בכלי AI. להלן פירוט של המקורות והפרומפטים ששימשו אותנו:

1. **Firebase (Google Cloud)**

* קישור [Firebase](https://firebase.google.com/)
* שימוש **:** שימש כמסד נתונים בזמן אמת לאחסון נתונים שנאספו מהמשתמשים.

1. **Python Official Documentation**

* קישור [Python Docs](https://docs.python.org/3/)
* שימוש **:** נעשה שימוש בתיעוד הרשמי של Python כדי להבין וליישם פונקציות שונות בשפת Python, כולל עבודה עם ספריות כמו Pandas, Requests, ועוד.

1. **NLTK (Natural Language Toolkit)**

* קישור [NLTK](https://www.nltk.org/)
* שימוש**:** נעשה שימוש בספריית NLTK ליצירת צ'אטבוט באמצעות עיבוד שפה טבעית (NLP).

1. **Matplotlib**

* קישור [Matplotlib](https://matplotlib.org/)
* שימוש **:** שימשה להצגת גרפים וויזואליזציה של נתונים שהועלו למערכת.

1. **ReportLab**

* קישור [ReportLab](https://www.reportlab.com/)
* שימוש**:** שימשה ליצירת דוחות PDF מהנתונים שנסוננו על ידי המשתמשים.

1. **Tailwind CSS**

* קישור [Tailwind CSS](https://tailwindcss.com/)
* שימוש**:** מסגרת עבודה לעיצוב ממשקי משתמש ששימשה אותנו כדי לעצב את הממשק בצורה מודרנית ואינטואיטיבית.

1. **Google Collab**

* קישור [collab](https://colab.research.google.com/)
* שימוש : להרצת קוד Python בסביבת ענן לצורך בדיקות וניסויים.

1. **ChatGPT**

* קישור [ChatGPT](https://chatgpt.com/)
* שימוש : במהלך הפיתוח השתמשנו ב-ChatGPT להכוונה ולתמיכה במספר מקרים.

1. **Wikipedia**

* קישור [WIKI](https://en.wikipedia.org/wiki/Non-functional_requirement)
* שימוש : נעשה שימוש במקור זה כדי להבין ולהסביר את המושג של דרישות לא-פונקציונליות, הכוללות היבטים כמו ביצועים, אבטחה, זמינות, ויכולת תחזוקה של המערכת.