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

קבוצת שבלול

***קישור למחברת :***

<https://colab.research.google.com/github/Niv0902/Shablool/blob/main/ShabloolSearchEngineHM2.ipynb>

***קישור לGITHUB :***

<https://github.com/Niv0902/Shablool/tree/main>

**שם משתמש וסיסמא למנהל   
Username : Niv  
Password : Niv   
  
שם משתמש וסיסמא לעובד  
UserName : Eli  
Password : Eli**

המשימה בתרגיל זה: בניית מסכים מרכזיים במערכת

*חלק ראשון : בניית מערכת (10 נקודות)*  
  
כיצד נעשתה החלוקה מול הצוות :   
חלוקת המשימות הוקצתה לאחר הניסיון מהתרגיל בית הראשון, סטודנטים חזקים יותר קיבלו משימות קשות יותר, סטודנטים השולטים בבקנד ובמימוש אלגוריתמים קיבלו את משימת מנוע החיפוש ואלו ששולטים בעיצוב קיבלו הכנת המסכים וממשק המשתמש.  
  
האם היה ממשק בין חברי הצוות:   
כן. למרות הקשיים הראשוניים, התקיים ממשק עבודה בין חברי הצוות, כולל שיתוף קבצים, חלוקת אחריות מחדש ושיחות הבהרה. חברי צוות שהפגינו מוכנות לקחת יוזמה סייעו בקידום חלקים שלא בוצעו במועד, מה שסייע להשלמת המטלה בזמן.

המשימות מולאו במלואן על ידי כל חברי הצוות.

|  |  |  |
| --- | --- | --- |
| **איטרציה 1** | | |
| **שם חבר הצוות ותפקיד בתרגיל זה** | **משימות שהוקצו** | **משימות שהושלמו** |
| ניב אורן Scrum master ,Project manager | חלוקת העבודה בין חברי הצוות, מימוש האינדקס, מימוש מנוע החיפוש, איחוד בין הגרסאות השונות, אישור סופי של המסכים | all |
| יוני גרינברג *backend developer* | מימוש האינדקס, מימוש מנוע החיפוש | all |
| אלי רובינוב  *frontend developer,UI* | עיצוב המסכים וכל חווית המשתמש | all |
| עמנואל אלקובי  UI,QA | מילוי קובץ הוורד | all |
| שחר נחום  Product manager | סידור מסך הגרפים | all |
| אדהם עלואן  Frontend developer | הוספת פיצ'ר למערכת | all |

**🔹 רשימת ה־Stop Words שבחרנו:**

'a', 'an', 'the', 'and', 'or', 'but',

'to', 'from', 'of', 'in', 'on', 'at', 'for', 'by', 'with', 'as',

'this', 'that', 'these', 'those', 'it', 'its',

'be', 'is', 'are', 'was', 'were', 'has', 'have', 'had',

'not', 'no', 'yes', 'true', 'false', 'null', 'none'

**🔹 נימוק לבחירת המילים:**

בחרנו במילים אלו משום שמדובר במילים פונקציונליות (מילות עצירה נפוצות) שאינן נושאות משמעות תוכן משמעותית במסמכים. מילים אלו מופיעות כמעט בכל טקסט באנגלית, אך אינן תורמות להבנת הנושא המרכזי או להבחנה בין מסמכים. הסרתן מסייעת לבנות אינדקס יעיל יותר, ממוקד במונחים תוכניים חשובים בלבד.

**🔹 Stemming / Lemmatization:**

במערכת השתמשנו ב־**Stemming**, באמצעות האלגוריתם PorterStemmer מספריית nltk . תהליך זה חותך סיומות נפוצות של מילים ומחזיר את "שורש" המילה (לדוגמה: המילים running, runs, ran כולן יהפכו ל run ).  
שיטה זו פשוטה ומהירה ומספקת תוצאה מספקת עבור מערכת אינדוקס בסיסית כמו שלנו.

*חלק שני:*

*יש להריץ את הקוד במחברת, ולחכות 30 שניות לנסיון האזנה לחיישנים. אם אין מידע מהחיישנים הוא לוקח פייק-דאטה מהפייר-בייס.*

***התייחסות ל־8 כללי הזהב של שניידרמן – איך המערכת שלנו מיישמת אותם:***

**1. עקביות (Consistency):**המערכת שומרת על עיצוב אחיד בין כל המסכים – צבעים קבועים, גופנים אחידים, וכפתורים באותם מקומות. לדוגמה, גם במסך ההתחברות וגם במסך הרישום, השדות מוצגים בצורה דומה.

**2. נגישות למשתמשים שונים (Universal Usability):**המערכת לא מניחה שהמשתמש הוא מומחה – כל כפתור ושדה מוסברים, יש הנחיות ברורות, והעיצוב פשוט להבנה. גם משתמש חדש יכול להבין מה לעשות בלי הסברים מסובכים.

**3. משוב ברור (Informative Feedback):**בכל פעולה שהמשתמש עושה – כמו התחברות, רישום או הורדת קובץ – הוא מקבל הודעה מיידית שמסבירה אם הצליח או נכשל, ולמה. לדוגמה: “Login successful” או “Please enter both username and password”.

**4. סגירה ברורה של תהליך (Design Dialogs to Yield Closure):**אחרי כל פעולה, המשתמש יודע בוודאות שהיא הושלמה. יש הודעת סיום ברורה – למשל אחרי רישום או חיפוש, מופיעה הודעה עם סימון של הצלחה או כישלון.

**5. מניעת שגיאות (Prevent Errors):**המערכת בודקת שהשדות מולאו לפני שליחת טפסים. אי אפשר להירשם או להתחבר עם שדות ריקים, וככה נמנעות טעויות פשוטות.

**6. אפשרות לחזור אחורה (Permit Easy Reversal of Actions):**  
כמעט בכל מסך יש כפתור “Back” שמאפשר למשתמש לחזור בלי להיתקע. אם נכנסת למסך רישום והתחרטת – אין בעיה לחזור.

**7. שליטה בידי המשתמש (Support Internal Locus of Control):**המערכת פועלת רק לפי פעולות של המשתמש – אין דברים שקורים "מעצמם". המשתמש מחליט מתי לחפש, להירשם, להוריד וכו'.

**8**. **הפחתת עומס על הזיכרון (Reduce Short-Term Memory Load):**המשתמש לא צריך לזכור מידע קודם. הכל מופיע על המסך – תוצאות חיפוש, הודעות, שמות משתמשים – וזה מפחית את העומס המנטלי.

|  |  |  |
| --- | --- | --- |
| הערת משוב | האם התבצע שינוי באפליקציה בעקבות ההערה? | נימוק |
| הגרפים | כן | |  | | --- | |  |  |  | | --- | | עדכנו את הגרפים כך שיוצגו בצורה קריאה, ברורה ומסודרת יותר. | |  | |
| |  | | --- | | הוספת לינק לתוצאות חיפוש |  |  | | --- | |  | | כן | בכל תוצאה יש כעת קישור ישיר לעמוד הרלוונטי. |
| עיצוב לא עובד או לא מעניין | כן | שידרגנו את העיצוב: צבעים חדשים, אייקונים, ופריסה נקייה וחדשנית. |
| גרפים לא מופרדים | כן | שופרה ההיררכיה הוויזואלית בין הגרפים ונעשה שימוש בקונטרסט צבעים גבוה. |
| צבעים יותר בהירים | כן | השתמשנו בצבעים יותר נעימים. |
| |  | | --- | | עיצוב דחוס | |  |  |  | | --- | |  | | כן | הגברנו ריווחים בין רכיבים, שיפרנו מרווחים פנימיים בטבלאות וטפסים. |
| |  | | --- | | המערכת מבולגנת |  |  | | --- | |  | | כן | ביצענו איחוד של אזורים שונים במסכים, הסרנו כפילויות ואירועים לא רלוונטיים. |

**שאלון SUS**

קיבלנו ציון של  **61.5 מתוך 100** בשאלון SUS, שזה אמור להיות "בינוני–נמוך", אבל זה לא אומר יותר מדי במקרה שלנו.

רוב המשתמשים מילאו את השאלון **במהירות, בלי לקרוא כמו שצריך**, רק רצו לסיים וללכת.  
חלק מהשאלות היו הפוכות – לפעמים ציון גבוה זה טוב, ולפעמים זה דווקא רע – ואף אחד לא הסביר להם את זה.  
לא הייתה הדרכה, לא הייתה בקרה, ובפועל הם פשוט סימנו משהו וזהו.

מה שכן, **כששאלנו אותם ישירות וקיבלנו פידבק אמיתי – רוב התגובות היו חיוביות**.  
עשינו הרבה שיפורים לפי מה שהם הציעו, והממשק התקדם משמעותית.

**בשורה התחתונה:** הציון הזה לא באמת משקף את החוויה או את השימושיות של המערכת.  
זה סתם מספר שהתבסס על שאלון שמולא בחפיפניקיות.

**3 מדדים להצלחת המערכת**

**שיעור שימוש בפועל (Adoption Rate)**  
כמה משתמשים בפועל התחברו למערכת, ביצעו חיפושים, נרשמו, או השתמשו בלוח הניהול.  
מערכת מצליחה היא כזו שלא רק נבנתה – אלא באמת משתמשים בה.

**מהירות השגת ערך (Time to Value)**  
כמה זמן לוקח למשתמש חדש (נגיד מנהל או משתמש רגיל) להבין איך המערכת עובדת, ולהוציא ממנה פעולה משמעותית – כמו חיפוש, הורדה של קובץ, או צפייה בנתוני חיישנים.  
אם משתמש מצליח להבין ולהשתמש במערכת מהר – זו הצלחה.

**שמירה על משתמשים לאורך זמן (Retention / Engagement)**  
מדד שבודק האם המשתמשים חוזרים להשתמש במערכת שוב, או רק נכנסו פעם אחת ויצאו.  
אם המשתמשים מתחברים שוב ושוב – סימן שהמערכת באמת עוזרת להם.

**המערכת שלנו, "Shablool Search Engine", מבוססת על ארכיטקטורת שכבות בהתאם למבנה שהוצג בהרצאה 7. המערכת מחולקת ל־4 שכבות עיקריות, כאשר כל שכבה אחראית על תחום מסוים, והזרימה מתבצעת מלמעלה למטה.**

תמונה שמכילה טקסט, צילום מסך, תרשים

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

**Layered Architecture Breakdown – Shablool Search Engine**

**1. Presentation Layer**

* תפקיד: ממשק למשתמש (תוצאות, טפסים, לחצנים)
* קוד: search\_screen(), admin\_screen(), register\_screen(), user\_screen()
* טכנולוגיה: ipywidgets, HTML

**2. Application Layer**

* תפקיד: טיפול בפעולות משתמש וחיבור ללוגיקה
* קוד: do\_search(), on\_login\_clicked(), on\_register\_clicked(), button.on\_click(...)

**3. Business Logic Layer**

* תפקיד: לוגיקה עסקית – חיפוש, רישום, ניתוח טקסט
* קוד: search\_mqtt(), firebase\_register(), clean\_and\_tokenize(), סטמינג ו־Stop Words

**4. Data Access Layer**

* תפקיד: חיבור למקורות נתונים חיצוניים
* קוד: FBconn.get(), FBconn.put(), paho.mqtt.client, requests, BeautifulSoup

**פיצ'ר שהוספנו למערכת**

הפיצ'ר שהוספנו למערכת מאפשר למנהל לראות את כל המשתמשים הרשומים ולרשום משתמשים חדשים. רק מנהל יכול לגשת למסך הזה – משתמש רגיל לא רואה אותו בכלל.

בצד של הקוד, הפיצ'ר הזה ממומש בתוך הפונקציה admin\_screen(), ששם מציגים את כל המשתמשים הרשומים מתוך Firebase. הרישום של משתמש חדש נעשה בפונקציה register\_screen(), כשבסוף זה נרשם ל־Firebase דרך firebase\_register().

מה שהמנהל רואה בפועל זה מסך עם טבלה של כל המשתמשים במערכת, כולל התפקיד שלהם כמו Manager או Employee וגם כפתור לרשום משתמש חדש. זה מאפשר ניהול בסיסי של הרשאות מתוך הממשק עצמו.