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

**תרגיל בית 3**- **עבודה בצוותי העבודה**

מועד הגשה: 26.1.25

המשימה בתרגיל זה: סיום בניית המערכת, ביצוע קוד ריוויו לצוות אחר

איטרצית פיתוח מס 2

שימו לב: למטלה זו שלושה חלקים

**חלק ראשון.**עליכם להגדיר את בעלי התפקידים לאיטרציה זו (יש להחליף מתרגיל בית 2) יש לרשום מי מהנדס.ת המערכת, האחראי.ת בתרגיל זה.

|  |  |  |  |
| --- | --- | --- | --- |
| **שם חבר הצוות** | **משימות שהוקצו** | **משימות שהושלמו** | בדיקת קבלה- acceptance test האם הבדיקה עברה? |
| שי | בניית צ'אט בוט אינטראקטיבי שכולל מידע מהDB | הכל | האם הצ'אט בוט אינטראקטיבי ועונה לנתונים שנשמרו בDB? עברה. |
| רוני | הוספת מסך להצגת רשימת האינדקסים | הכל | האם המסך מציג את רשימת האינדקסים בצורה נכונה וללא שגיאות? עברה. |
| טל | מילוי משימות 1 ו-2 בחלק 2 של הפרויקט | הכל | האם המערכת עובדת בצורה תקינה לאחר הוספת הפתרון? עברה. |
| פבלו | בדיקות קוד (Code Review) | הכל | האם כל שורות הקוד עומדות בסטנדרטים ונטולות שגיאות? עברה. |
| מעיין | תיקונים אחרונים למערכת, כולל UI ועיצוב | הכל | האם העיצוב ברור ונוח לשימוש לפי דרישות המערכת? עברה. |
| ינון | יצירת מצגת וסרטון מעלית | הכל | האם התוכן מעביר את תכונות ותוכן המערכת? עברה. |

***חלק שני: בניית המערכת*** (80 נקודות)

עליכם לסיים את בניית המערכת.

בפרט, יש לוודא קיום של כלל האלמנטים הבאים:

1. כל הפיצ'רים שהתחלתם לממש בתרגיל בית 2 - יש לוודא שכל המסכים תקינים ועובדים כראוי, **בקולאב**,ולעבוד עם firebase לצורך שמירת נתונים .אין להשאיר קישורים או כפתורים למסכים שלא קיימים.- הושלם.

יש לוודא שניתן להריץ את הפרויקט מהמחברת בלבד , ללא כל צורך בהטענה של קבצים נוספים או יציאה למערכת חיצונית (כגון FLASK). (30 נקודות) -הושלם.

1. שימוש ב -microservices (תרגול 7)- עליכם לעשות שימוש לפחות בשניים. פרטו באילו שירותים עשיתם שימוש, ומה היתרון בשימוש בשירותים אלו.(10 נקודות) –

מומש בחלק זה חמישה סרביסים:

**PageFetcher** – אחראי לאיסוף נתונים מ-URL.

**DataExtractor** – אחראי על חילוץ נתונים רלוונטיים מתוך תוכן HTML.

**IndexBuilder** – בונה אינדקס המבוסס על הנתונים שהופקו.

**QueryProcessor** – מעבד שאילתות משתמש ומספק תוצאות חיפוש, כולל הרחבת שאילתות עם מילים נרדפות ותיקון אוטומטי.

 **FirebaseService** – מטפל בשירותי העלאה ואחסון נתונים ב-Firebase.

**תרונות השימוש במיקרוסרביסים:**

* **הפרדה בין אחריות**: כל קלאס מטפל בתחום אחריות שונה (PageFetcher לאוסף נתונים, DataExtractor לאחזור נתונים). זה מפחית את התלות בין הקוד ומאפשר תחזוקה קלה יותר.
* **גמישות**: ניתן לשפר או להחליף כל מיקרוסרביס בנפרד, כל עוד ה-API הפנימי שלו נשמר.
* **יכולת הרחבה**: במקרה של צורך בהוספת פונקציות נוספות, ניתן לעשות זאת על ידי יצירת שירותים חדשים, מבלי לפגוע בשירותים הקיימים.
* **פשטות הבדיקות**: קל יותר לבדוק כל שירות בנפרד כיחידה עצמאית. לדוגמה, ניתן לבדוק את PageFetcher עם קלטים מדומים, ללא תלות ב-DataExtractor.

השימוש בקלאסים אלו מדגיש את השימוש בגישת ה-microservices המודרנית.

1. צטבוט שישולב במערכת (כפי שלמדתם בתרגול 8), הכולל מידע אשר מגיע מה - DB שלכם, ועונה בצורה אינטליגנטית לשאלות המשתמש. (15 נקודות). -הושלם.
2. מהם ה -KPI הרלוונטיים בפרויקט שלכם? הסבירו? (10 נקודות)

**ה-KPI הרלוונטיים לפרויקט הם:**

1. זמן תגובה של השירותים:  
   מדד לזמן התגובה הממוצע של השירותים במערכת, כמו חיפוש אינדקסים, טעינת נתונים מה-DB, ותגובות מהצ'אט בוט.
2. זמינות המערכת:  
   מדד לזמינות הכוללת של השירותים, המשפיע על זמינות המערכת למשתמשים.
3. דיוק המידע בצ'אט בוט:  
   כמות התשובות המדויקות שהצ'אט בוט מספק ביחס לשאלות שנשאלו, לצורך שיפור חוויית המשתמש.
4. שיעור טעויות במערכת:  
   כמות השגיאות שהמשתמשים נתקלים בהן בעת השימוש במערכת, לדוגמה שגיאות בקשות לשרת או בקריאות נתונים.
5. ביצועי מסד הנתונים:  
   מהירות הגישה לנתונים מה-DB עבור פעולות שונות כמו שליפת אינדקסים או עדכון נתונים.
6. שימושיות ממשק המשתמש:  
   שביעות רצון המשתמשים מהממשק ומנגישות הפונקציות המרכזיות של המערכת.

1. שקיפות אלגוריתמית – כיצד הבהרתם למשתמשים את האלגוריתמים והנתונים הנאספים בקוד שלכם? (5 נקודות)  
   **הודעות מערכת**: הצגת הסברים על תיקון והרחבת שאילתות, למשל " האם התכוונת ל smartphone ? "

**ניהול מילים נרדפות**: ממשק לעריכת מילון המילים הנרדפות לצפייה ושליטה בתוצאות החיפוש

**תיעוד נתונים**: הצגת תוצאות מורחבות ומותאמות באופן ברור למשתמש. **סטטיסטיקות חזותיות**: עמוד המציג מונחים נפוצים ו- Cloud Word לתובנות על עיבוד המידע.

1. אתגרים שעלו במהלך העבודה , וכיצד התמודדתם איתם (לדוגמא- DB גדול מדי, יצירת ויזואליזציה, מבנה הקוד וכו) (10 נקודות)

אתגרים שעלו במהלך העבודה וכיצד התמודדנו:

* שימוש ב-GUI דרך Google Colab לא היה נוח: במהלך העבודה, נתקלנו בקושי לשלב ממשק משתמש גרפי (GUI) בצורה נוחה בתוך Google Colab, שאינה מותאמת לתמיכה מלאה בווידג'טים אינטראקטיביים.

פתרון: השתמשנו בDash -, פלטפורמה מתקדמת המאפשרת יצירת ממשקים גרפיים אינטראקטיביים בצורה אינטואיטיבית, ותומכת גם בריצה בתוך Google Colab. DASH סיפקה פונקציות מוכנות רבות לUI מה שהפך את העבודה על ממשק המשתמש ליעילה ונוחה יותר, תוך שימוש מלא במגבלות וביכולות של Colab.

* חסימת זחלנים על ידי אליבאבא :ניסיונות לגשת לתוכן האתר של Alibaba באמצעות זחלנים (web scrapers) נחסמו על ידי מנגנוני האבטחה של האתר.   
  פתרון: במקום לנסות לעקוף את המנגנונים, בחרנו להוריד את קבצי ה-HTML של הדפים באופן ידני ולעבוד עליהם באופן מקומי. שיטה זו אפשרה לנו לבצע ניתוח של המידע הרצוי מבלי להפר את כללי השימוש של האתר.

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

בחלק זה תכינו מסמכים המתארים את הפרויקט שלכם:

1. יש לבנות תיק למתכנת הכולל את שמות כל הקבצים המרכזיים, פונקציות מרכזיות, קטעי קוד/תבניות עיצוב מעניינים שהשתמשתם בהם.
2. יש לבנות תיק למשתמש , הכולל הסבר כללי על המערכת , פירוט מסכים, מעברים בין מסכים והסבר על טעויות אפשרויות.
3. התייחסו בתיק המשתמש לשקיפות אלגוריתמית – כיצד הבהרתם למשתמשים את האלגוריתמים והנתונים הנאספים בקוד שלכם?
4. 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.

חלק שלישי בדפים הבאים:

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

בהצלחה!

**תיק למתכנת**

**קבצים מרכזיים:**

* **Cloud\_With\_MicroServices.ipynb**

**פונקציות מרכזיות:**

**fetch\_pages(start, end)**הסבר:פונקציה זו מבצעת קריאה דינמית לדפים באתר אינטרנט ומחזיר את תוכן ה-HTML של כל דף בפורמט BeautifulSoup ומטפלת בשגיאות בעת כישלון קריאה, כולל טיפול בשגיאות HTTP.

def fetch\_pages(self, start=1, end=33):

        pages\_content = {}

        for i in range(start, end + 1):

            url = self.base\_url.format(i)

            try:

                response = requests.get(url)

                if response.status\_code == 200:

                    pages\_content[f"page\_{i}"] = BeautifulSoup(response.text, "html.parser")

                else:

                    print(f"Failed to fetch page {i}: Status code {response.status\_code}")

            except Exception as e:

                print(f"Error fetching page {i}: {str(e)}")

        return pages\_content

**extract\_data(page)**הסבר: פונקציה זו שולפת נתונים מתוך תוכן HTML של דף אינטרנט ומחזיר מידע מובנה על פריטים.

    def extract\_data(self, page):

        if page is None:

            return []

        products = []

        for product in page.select(".fy23-search-card.m-gallery-product-item-v2.J-search-card-wrapper.fy23-list-card.searchx-offer-item"):

            title\_element = product.select\_one(".search-card-e-title a")

            span\_element = product.select\_one(".search-card-e-title span")

            price\_element = product.select\_one(".search-card-e-price-main")

            image\_element = product.select\_one(".search-card-e-slider div a img")

            rating\_element = product.select\_one(".search-card-e-review")

            if title\_element and span\_element and price\_element and image\_element:

                product\_data = {

                    "title": span\_element.get\_text(strip=False),

                    "url": "https://" + title\_element.get('href')[2:],

                    "price": price\_element.get\_text().split("-")[0].strip(),

                    "image": "https://" + image\_element.get('src')[2:],

                    "rating": rating\_element.get\_text(strip=False) if rating\_element else "NA"

                }

                products.append(product\_data)

        return products

**remove\_exact\_duplicates(self, products)**הסבר: פונקציה זו מסירה כפילויות מדויקות מתוך רשימת המוצרים על בסיס כותרת וקישור.

def remove\_exact\_duplicates(self, products):

        seen = set()

        unique\_products = []

        for product in products:

            identifier = (product['title'], product['url'])

            if identifier not in seen:

                seen.add(identifier)

                unique\_products.append(product)

        return unique\_products

**build\_index(database)**הסבר: פונקציה זו בונה אינדקס לחיפוש מהיר של פריטים, כולל הפעלת סטמינג והסרת מילים מיותרות, מחלקת את הנתונים לפי מונחי חיפוש ייחודיים ומאפשרת התאמה מהירה.

    def build\_index(self, database):

        temp\_index = {}

        for doc\_id, product in enumerate(database):

            if 'title' not in product or not product['title']:

                print(f"Skipping product at index {doc\_id} due to missing title.")

                continue

            content = product['title']

            words = set(re.findall(r'\w+', content.lower()))

            for word in words:

                if word not in temp\_index:

                    temp\_index[word] = {"DocIDs": [], "details": []}

                temp\_index[word]["DocIDs"].append(doc\_id)

                temp\_index[word]["details"].append({

                    "title": product["title"],

                    "url": product["url"],

                    "price": product["price"],

                    "image": product["image"],

                    "rating": product["rating"],

                    "DocID": doc\_id

                })

        temp\_index = self.remove\_stop\_words(temp\_index)

        temp\_index = self.apply\_stemming(temp\_index)

        transformed\_index = [{"term": term, "DocIDs": data["DocIDs"], "details": data["details"]} for term, data in temp\_index.items()]

        return transformed\_index

**search\_index(query)**הסבר: פונקציה זו מבצעת חיפוש באינדקס המבוסס על שאילתה, כולל הרחבה עם מילים נרדפות ומחזירה תוצאות ממויינות לפי התאמות למונחי החיפוש

    def search\_index(self, query):

        query\_words = self.expand\_query\_with\_synonyms(query)

        stemmed\_query\_words = [self.stemmer.stem(word) for word in query\_words]

        results = {}

        for word in stemmed\_query\_words:

            for entry in self.index:

                if entry["term"] == word:

                    for detail in entry["details"]:

                        title = detail["title"].lower()

                        if word in title:

                            unique\_key = (detail["title"], detail["url"])

                            if unique\_key not in results:

                                results[unique\_key] = {

                                    "title": detail["title"],

                                    "url": detail["url"],

                                    "price": detail["price"],

                                    "image": detail["image"],

                                    "rating": detail["rating"],

                                    "match\_count": 0

                                }

                            results[unique\_key]["match\_count"] += 1

                    break

        sorted\_results = dict(sorted(results.items(), key=lambda item: item[1]["match\_count"], reverse=True))

        return sorted\_results

**upload\_data(path, data)**הסבר: פונקציה זו מעלה נתונים למסד הנתונים Firebase ומאפשרת גיבוי וניהול של נתוני הפרויקט בענן.

    def upload\_data(self, path, data):

        result = self.conn.post(path, data)

        print(result)

        return result

**תיק למשתמש:**

**הסבר כללי על המערכת:**

מערכת מיועדת למשוך נתונים מהאתר "Alibaba", לעבד אותם ולספק ממשק משתמש אינטראקטיבי מבוסס Dash בGoogle Colab, המערכת מציעה חוויית משתמש אינטואיטיבית הכוללת חיפוש מותאם אישית, עיבוד נתונים, הצגת סטטיסטיקות וניהול אינדקסים. הפלטפורמה תומכת במעברים בין מסכים בצורה חלקה ומאפשרת עבודה יעילה ונוחה.

**פירוט מסכים:**

**1. מסך חיפוש – Search Page**:

תיאור מסך:

מסך זה מאפשר למשתמש להזין שאילתת חיפוש בתיבת חיפוש.  
- שימוש באינדקס פנימי המאפשר חיפוש חכם ומותאם אישית למשתמש.  
- התאמה אוטומטית של השאילתה במקרה של שגיאות הקלדה (autocorrect).מעברים בין מסכים:  
- לחיצה על כפתור Search תעביר למסך תוצאות החיפוש.  
-לחיצה על Statistic Page תעביר למסך הסטטיסטיקות.

-לחיצה על Chat Bot תעביר למסך הצ'אט בוט.  
- לחיצה על Manager Page תעביר למסך ניהול אינדקסים.

**2. מסך תוצאות החיפוש – Results Page:**תיאור מסך:  
מסך זה מציג את תוצאות החיפוש לפי השאילתה שהוזנה.  
- מוצגת רשימה הכוללת: כותרת המוצר, מחיר, דירוג, קישור למוצר באתר Alibaba ותמונה.  
- כפתור Load More שיטען תוצאות נוספות.מעברים בין מסכים:

- לחיצה על Search Pageתעביר למסך החיפוש.  
-לחיצה על Statistic Page תעביר למסך הסטטיסטיקות.

-לחיצה על Chat Bot תעביר למסך הצ'אט בוט.  
- לחיצה על Manager Page תעביר למסך ניהול אינדקסים.

**3. מסך סטטיסטיקות - Statistic Page:**תיאור מסך:  
מסך זה מציג גרף הממחיש את הקשר בין מחיר המוצר לדירוג שלו.  
מעברים בין מסכים:  
- לחיצה על כפתור Back to Search Page תעביר למסך החיפוש.

- לחיצה על כפתור Back To Manager Page תעביר למסך ניהול אינדקסים.

**4. מסך צ'אט בוט – Chat Bot:**תיאור מסך:  
מסך המאפשר למשתמש לשאול שאלות ולקבל תשובות מצ'אט בוט המבוסס על הנתונים שנאספו.  
- המערכת תומכת בשאילתות שונות כמו חיפוש מידע כללי על מוצרים או סטטיסטיקות שנאספו.

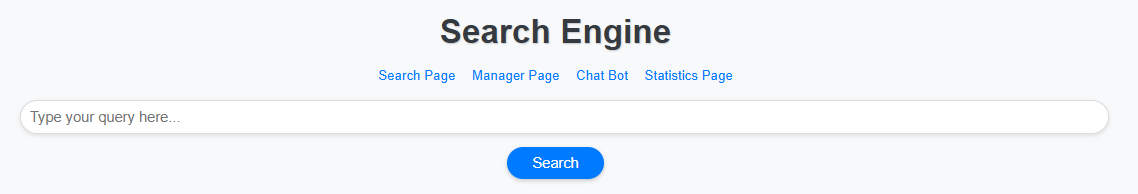
מעברים בין מסכים:

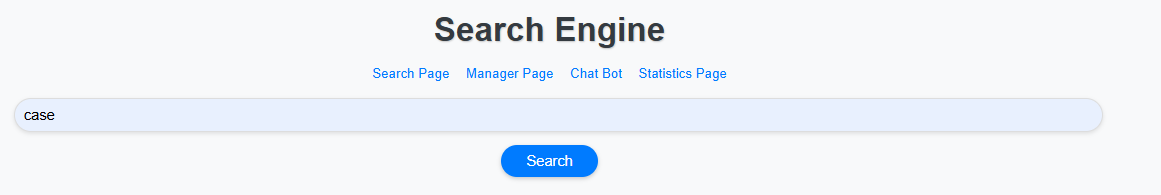
- לחיצה על Search Pageתעביר למסך החיפוש.  
-לחיצה על Statistic Page תעביר למסך הסטטיסטיקות.

-לחיצה על Chat Bot תעביר למסך הצ'אט בוט.  
- לחיצה על Manager Page תעביר למסך ניהול אינדקסים.  
- לחיצה על כפתור Send תעביר תשלח לצ'אט שאלה שנרשמה.

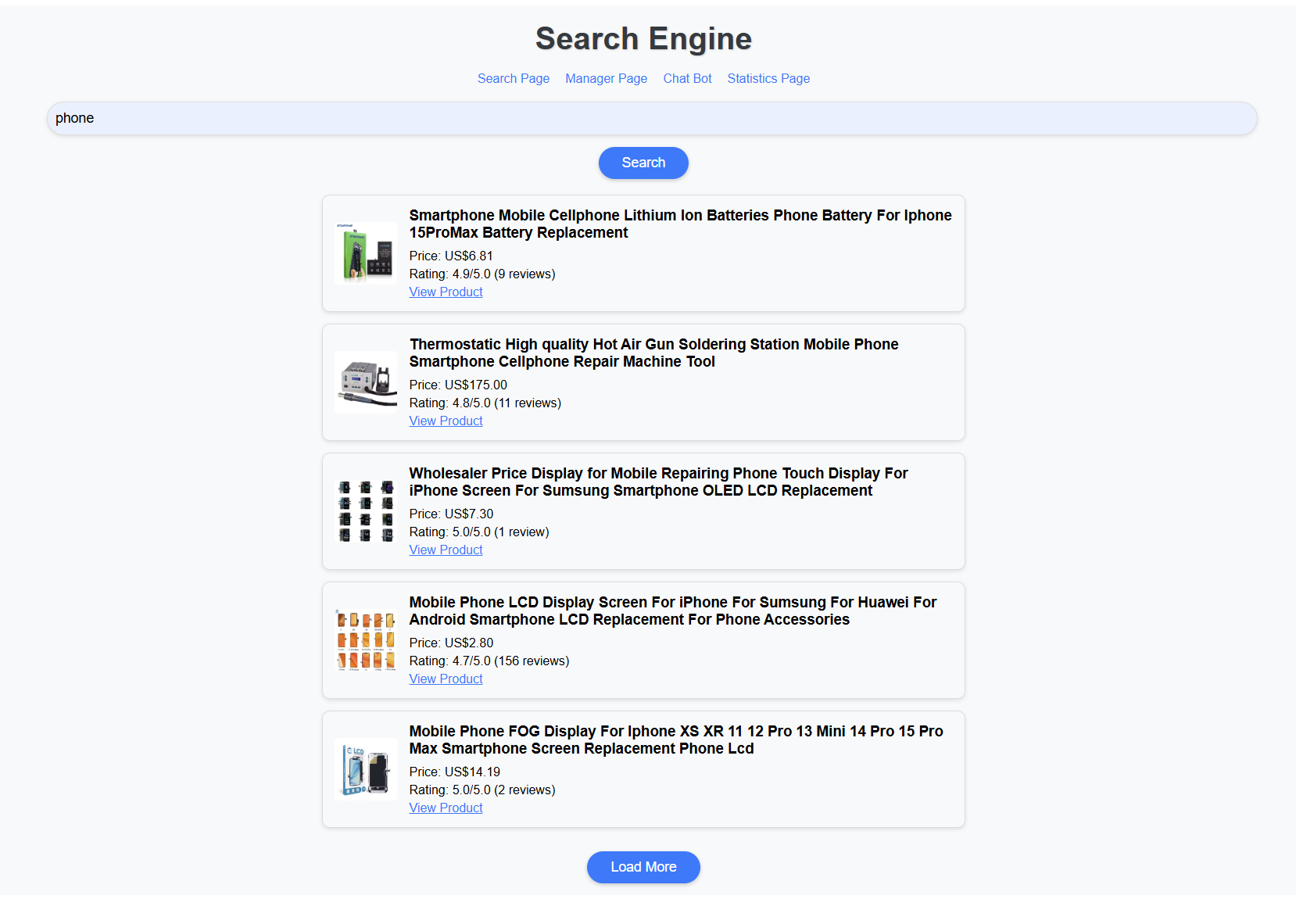
**5. מסך ניהול אינדקסים: Manager Page -**תיאור מסך:  
מסך המיועד לניהול אינדקסים פנימיים של המערכת, כולל אפשרות הוספה, מחיקה ועדכון של אינדקסים.  
מעברים בין מסכים:  
- לחיצה על כפתור Back to Search Page תעביר למסך החיפוש.

**1. מסך חיפוש – Search Page**:





**2. מסך תוצאות החיפוש – Results Page:**



תמונה שמכילה טקסט, צילום מסך, גופן, קו

התיאור נוצר באופן אוטומטי

**3. מסך סטטיסטיקות - Statistic Page:**

A screen shot of a graph

Description automatically generated

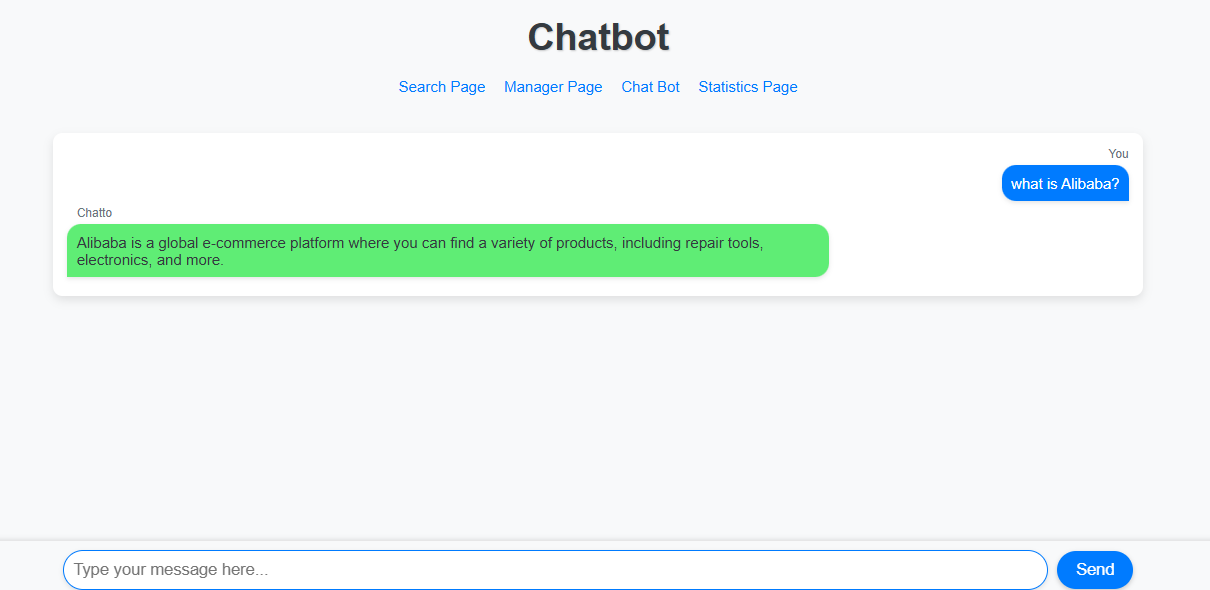
תמונה שמכילה טקסט, צילום מסך, גופן, קו

התיאור נוצר באופן אוטומטי

**4. מסך צ'אט בוט – Chat Bot:**

תמונה שמכילה טקסט, צילום מסך, גופן, קו

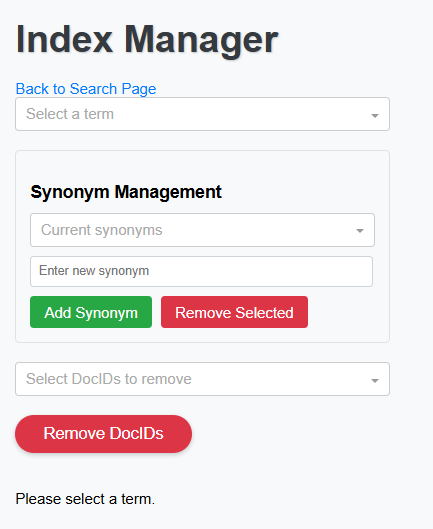
התיאור נוצר באופן אוטומטי

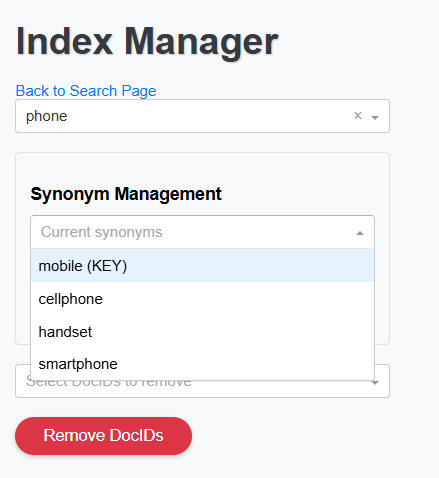


תמונה שמכילה טקסט, צילום מסך, גופן, קו

התיאור נוצר באופן אוטומטי

**5. מסך ניהול אינדקסים: Manager Page -**





**שגיאות אפשריות:**

**1. מחיקת אינדקסים:**

- מחיקת אינדקס שהוא KEY במילון המילים הנרדפות תגרום למחיקת כל המילים הנרדפות הקשורות אליו.

לדוגמה: אם נמחק את המילה mobileשמוגדרת כ KEYבמילון, כל המילים הנרדפות של mobile כמו phoneימחקו גם הן. **2. אינטראקציה עם צ'אט בוט:**

ייתכן שמשתמש יזין שאלה שאינה נתמכת או שאינה מחזירה תוצאה רלוונטית.  
  
3. בפונקציה **apply\_stemming:**

השימוש בפונקציה זו עלול להמיר מילים לצורה סטנדרטית באופן שגוי או לא צפוי, מה שיגרום לכך שבמסך ניהול האינדקסים, מילה מסוימת לא תימצאבשורת החיפוש:  
Select a term.לדוגמה: אם המשתמש מחפש את המילה ,mobile ייתכן שהפונקציה תמיר אותה לmobil וכך לא יתקבלו תוצאות של מילים נרדפות עבור מילת החיפוש המקורית.

**שקיפות אלגוריתמית:**  
  
**הודעות מערכת**: הצגת הסברים על תיקון והרחבת שאילתות, למשל " האם התכוונת ל smartphone ? "

**ניהול מילים נרדפות**: ממשק לעריכת מילון המילים הנרדפות לצפייה ושליטה בתוצאות החיפוש

**תיעוד נתונים**: הצגת תוצאות מורחבות ומותאמות באופן ברור למשתמש.

**סטטיסטיקות חזותיות**: עמוד המציג מונחים נפוצים ו- Cloud Word לתובנות על עיבוד המידע.