**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.

מהנדס מערכת: מירנא אבו חליל

|  |  |  |
| --- | --- | --- |
| **שם חבר הצוות** | **משימות שהוקצו** | **משימות שהושלמו** |
| יניב שתיל | צילום סרטון | צילום סרטון |
| עדן קנטור | הוספת צ'אט-בוט, הכנת מסמך וורד | הוספת צ'אט-בוט |
| דני לבובסקי | הוספת גרפים | הוספת גרפים |
| מירנא אבו חליל | כתיבת תיק משתמש | כתיבת תיק משתמש |
| דיאנא חוג'יראת | כתיבת תיק מתכנת | כתיבת תיק מתכנת |
| שי ליברמן | כתיבת KPI, אתגרים, מבחני קבלה | כתיבת KPI, אתגרים, מבחני קבלה |

|  |  |  |  |
| --- | --- | --- | --- |
| Acceptance Tests | | | |
| Precondition | Expected Result | Description | Test Name |
| Terms “Hello” or “World” are in the database | User will be transferred to result page displaying the webpages in which terms ‘Hello’ OR ‘World’ appears | 1.Click on search bar  2.insert query: “Hello world” in text area  3.click dropdown list “Search Type”  4. choose “OR” filter option  5. click search button | Search functionality, using filter OR, with results |
| There is no page with both “Hello” and “World” in the database | User will be transferred to result page displaying no webpages in which terms ‘Hello’ AND ‘World’ appears | 1.Click on search bar  2.insert query: “Hello world” in text area  3.click dropdown list “Search Type”  4. choose “AND” filter option  5. click search button | Search functionality, using filter AND, with no results |
|  | Alert will appear in red notifying user that the search bar is empty. | 1.Click on search bar  2. Leave text area empty  3. Click “Badger Search” | Search with empty string |
| The database contains numerous terms and their respective frequencies | After clicking “Statistics page” user will be moved to the respective page which will display a graph for “Top 10 Terms Frequencies”. Upon choosing a different title, the graph will update. After choosing “Doughnut” the graph will update a 2nd time. | 1. Click “Statistics page” 2. Click dropdown list under “Statistics Dashboard” 3. Choose “Top 10 searched queries” 4. Choose bar type “Doughnut” | Graph display |
| The term “magic” appears in the database | After clicking “admin page”, a text area for password will appear. Inputting the correct password will transfer user to the admin page. After choosing an index and deleting it, the index will no longer appear in the database and the dropdown list | 1. From the main page, click “Admin page” 2. Enter password into text area below search bar. Password is “123456”. Click Submit 3. From “Select Index” choose index “magic” 4. Click “Delete index” | Admin page, index deletion |
| The term “magic” appears in the database and the term has DocIDs assigned to it | After clicking “admin page”, a text area for password will appear. Inputting the correct password will transfer user to the admin page. After choosing an index, choosing a DocID and deleting it, the DocID will no longer appear in the database and the “Select DocID” for the “magic” index | 1. From the main page, click “Admin page” 2. Enter password into text area below search bar. Password is “123456”. Click Submit 3. From “Select Index” choose index “magic” 4. From “Select DocID” choose the first URL 5. Click “Delete Document” | Admin page, DocID deletion |
| The term “magic” appears in the database | After clicking “Chat assistant” button from main page, user will be moved to the page with the chatbot. After asking the question, the chatbot will return the title of the pages containing the word ‘magic’ | 1. Click on “Chat Assistant” from the main page 2. Enter “Which pages contain magic?” into text area and click enter | Using chatbot, successful answer |
| Chatbot only provides proper answer for the correct input as it states in the beginning of the dialog | After clicking “Chat assistant” button from main page, user will be moved to the page with the chatbot. After asking the question, the chatbot will return a message “I’m not sure about that” indicating an incorrect input was given | 1. Click on “Chat Assistant” from the main page 2. Enter “What pages contain magic?” into text area and click enter | Using chatbot, failed answer |
| The music player’s playlist is not empty | The music player will play whichever song is currently selected | 1. On the main page, click “Play” on the music player | Use music player |

***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.

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

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

**שירותי MicroServices שבהם עשינו שימוש בפרויקט:**

* + - 1. שירות crawling לשליפת מידע מאתר Oracle והכנסתו למסד הנתונים.

**יתרון:** השירות מאפשר איסוף אוטומטי של נתונים מאתר Oracle בצורה יעילה ורציפה, תוך שמירה על מבנה מודולרי שמאפשר עדכון ושיפור האלגוריתם מבלי להשפיע על שאר המערכת. שירות זה משפר את יכולת הרחבת המערכת לאתרים נוספים בעתיד.

* + - 1. שירות חיפוש במסד הנתונים לפי שאילתה והחזרת תשובה מתאימה למשתמש.

**יתרון:** השירות ממוקד בחיפוש מידע מדויק לפי שאילתות מוגדרות, מה שמפחית את העומס על שאר השירותים ומספק תשובות מותאמות אישית בזמן קצר. זה מאפשר הפרדה ברורה בין שכבת הלוגיקה לשכבת הגישה לנתונים.

* + - 1. שירות שליפת הנתונים ממסד הנתונים והצגתם כגרפים לפי מטריקות שונות.

**יתרון:** השירות מאפשר ויזואליזציה אינטואיטיבית של נתונים, מה שמקל על ניתוח תובנות וקבלת החלטות. יתרון נוסף הוא שניתן לשפר את התצוגות ללא השפעה על שאר השירותים במערכת.

* + - 1. שירות ממשק אדמין שמאפשר לעדכן אינדקסים או למחוק אותם ממסד הנתונים.

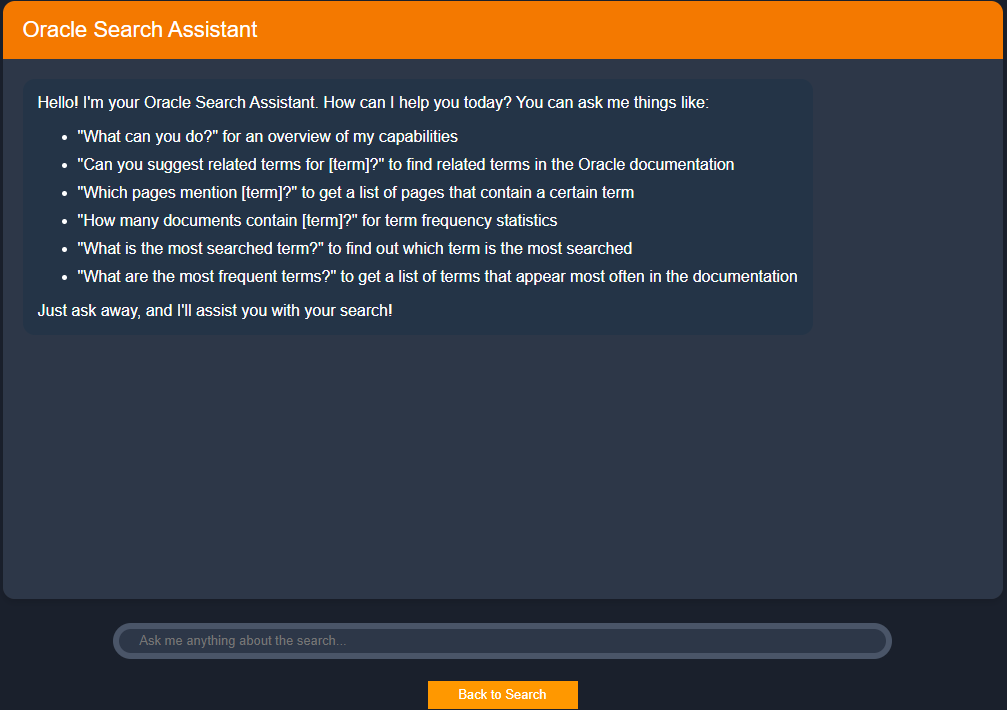
**יתרון:** השירות נותן שליטה גמישה למנהלי המערכת על מסד הנתונים (כגון עדכון או מחיקה של אינדקסים), תוך הגנה על ליבת המערכת מפני שינויים לא מבוקרים. הפרדה זו מגבירה את האבטחה ואת היכולת להתאים את המערכת לצרכים משתנים.

* + - 1. שירות Chatbot הכולל מידע ממסד הנתונים לשאלות המשתמש.

**יתרון:** שירות זה מספק חוויית משתמש משופרת עם גישה מיידית לנתונים רלוונטיים. הוא יכול לעבוד במקביל לשירותים אחרים, ובכך להקל על העומס ולשפר את הזמינות והתגובה המהירה לשאלות המשתמשים.

1. צטבוט שישולב במערכת (כפי שלמדתם בתרגול 8), הכולל מידע אשר מגיע מה - DB שלכם, ועונה בצורה אינטליגנטית לשאלות המשתמש. (15 נקודות).

**Oracle Search Assistant – צ'אט בוט:**



הצ'אט בוט שמימשנו בפרויקט הוא HardCoded, והוא מספק ממשק אינטראקטיבי למשתמשים, אלו הם היכולות שהצ'אט בוט מציע:

1. **שאלות כלליות על המערכת:**

שאלות כמו "hi", "what can you do?", ו-"thanks" יקבלו תגובות קבועות מראש (hardcoded), שמציגות למשתמש את יכולות הצ'אטבוט ומספקות חווית משתמש נעימה.

דוגמה:

"What can you do?" יקבל תשובה מפורטת על היכולות של הצ'אטבוט.

2. **מידע על תדירות מופעי מונחים:**

שאלת המונח הנפוץ ביותר:

שאילתות כמו: "What is the most searched term?" יחזירו את המונח עם תדירות ההופעה הגבוהה ביותר במערכת.

תשובה: המונח עם כמות ההופעות הגדולה ביותר ומספר ההופעות.

שאילתות על ספירת מסמכים:

דוגמה: "How many documents contain the term Oracle?"

תשובה: מספר המסמכים הכוללים את המונח המבוקש.

3. **הצגת עמודים המכילים מונחים:**

שאילתות המצביעות על עמודים:

דוגמה: "Which pages mention the term cloud?"

תשובה: עד 3 עמודים שמכילים את המונח המבוקש, כולל שם העמוד ותדירות ההופעה. אם יש יותר מ-3 עמודים, תופיע הערה על מספר העמודים הנוספים.

4. **סטטיסטיקות על מונחים:**

שאילתות על המונחים הנפוצים ביותר:

דוגמה: "What are the most frequent terms?"

תשובה: רשימה של חמשת המונחים הנפוצים ביותר במסד הנתונים, כולל מספר ההופעות שלהם.

5. **הצעת מונחים קשורים:**

שאילתות על מונחים קשורים:

דוגמה: "Can you suggest related terms for database?"

תשובה: עד 3 מונחים קשורים שמופיעים יחד עם המונח המבוקש במסמכים, כולל מספר המסמכים שבהם הם מופיעים יחד.

1. מהם ה -KPI הרלוונטיים בפרויקט שלכם? הסבירו? (10 נקודות)

**Performance KPIs (ביצועים גבוהים):**

- Average Response Time מדידת הזמן הממוצע מרגע שליחת בקשה ועד קבלת תשובה.  
**לוגיקה נוכחית**: תוצאות הזמן מוצגות כבר במסך פלט השאילתה, מה שמאפשר ניתוח ישיר של זמני התגובה.

- SLA Compliance Rate מעקב אחר % הבקשות שמסתיימות תוך 2 שניות.

**מדד עתידי בקוד:** ניתן להוסיף מדידת זמן לכל בקשה ולשמור את התוצאות ב-log או ב-DB, תוך השוואה אוטומטית מול 2 שניות. ניתן להציג אחוז העמידה ב-SLA כגרף או טבלה.

**Scalability KPIs: (הרחבת יכולות):**

- Response Time vs Load Ratio בדיקה כיצד זמני התגובה משתנים כשהעומס גדל.

**מדד עתידי בקוד:** לשלב מדמה עומסים (load testing tool כמו JMeter) יחד עם מדידת זמן תגובה לכל בקשה. הקוד יוכל לכלול לוגים שמודדים את מספר הבקשות הפעילות ואת זמני התגובה לכל עומס.

- Scale-Up Recovery Time כמה זמן לוקח למערכת להתייצב אחרי הוספת משאבים.

**מדד עתידי בקוד:** לוג שמודד את הזמן מרגע זיהוי עומס ועד לייצוב המערכת לאחר הוספת משאבים. ניתן לשלב טריגרים שמתריעים מתי נוספו משאבים ומתי ה-load ירד לרמה יציבה.

**Accessibility KPIs (ממשק משתמש ידידותי):**

- Average Clicks per Task מדידת כמות הפעולות הנדרשות להשלמת משימה.

**מדד עתידי בקוד:** להטמיע מעקב אחרי אינטראקציות משתמש (clicks) באמצעות אירועי UI.

- Time to First Success כמה זמן לוקח למשתמש חדש להשלים משימה בסיסית.

**מדד עתידי בקוד**: הוספת מעקב אחר משתמשים חדשים.

**System Availability KPIs (זמינות המערכת):**

- System Uptime מדידת אחוז הזמן שהמערכת זמינה (יעד: 99.9%).

**מדד עתידי בקוד:** להוסיף heartbeat service שמבצע בדיקות זמינות תקופתיות ומדווח ל-logging system את אחוזי הזמן שבהם המערכת זמינה.

- Mean Time to Recovery זמן ממוצע להתאוששות מתקלה (יעד: פחות מ-5 דקות).

**מדד עתידי בקוד**: לרשום לוגים של זמן גילוי התקלה וזמן פתרון התקלה. ניתן לחשב ממוצע לתקופות זמן מוגדרות ולשמור היסטוריית התאוששויות.

**Throughput KPIs (קיבולת / קצב ביצוע):**

- Peak Requests per Second כמות הבקשות המקסימלית שהמערכת מסוגלת לטפל בה.

**מדד עתידי בקוד**: לשלב בדיקות עומס עם רישום בקשות ל-DB כדי למדוד את הקצה העליון של הטיפול בבקשות.

- Request Rejection Rate שיעור הבקשות שנדחות כשהמערכת מגיעה לקצה היכולת.

**מדד עתידי בקוד**: לעקוב אחר בקשות שנדחות ולשמור את הנתונים ב-log או ב-DB לצורך ניתוח רמות עומס קריטיות.

1. שקיפות אלגוריתמית – כיצד הבהרתם למשתמשים את האלגוריתמים והנתונים הנאספים בקוד שלכם? (5 נקודות)

על פי עקרונות השקיפות האלגוריתמית (Algorithmic Transparency), פעלנו כדי להבטיח שהמשתמשים יבינו את אופן פעולתה של המערכת והנתונים הנאספים. להלן הפעולות שננקטו:

1. **הסבר ברור למשתמשים על אופן פעולת המערכת:**

במסך החיפוש, הוצגה הודעה המפרטת את השאילתא שבוצעה ואת מספר התוצאות שנמצאו (לדוגמה: "Your search appeared 71 times in the following links").

למשתמשים הוסבר כיצד ניתן לשנות את סוג החיפוש (AND/OR) ולבצע התאמות בהתאם לצרכיהם.

2. **הנגשת אלגוריתמי החיפוש והסטטיסטיקות:**

במסך הסטטיסטיקות, תדירות המילים ("Frequency") מוצגת בבירור ליד כל תוצאה. בנוסף, ניתנה אפשרות להצגת הנתונים באמצעות גרף (Bars, Line, או Doughnut) כדי להמחיש את המידע בצורה ויזואלית.

המשתמש יכול לראות את הקשר בין השאילתא לתוצאות הסטטיסטיות המוצגות.

3. **משוב אינפורמטיבי ומיידי:**

המערכת נותנת משוב ברור בכל שלב, כולל הודעות שגיאה במקרה של שאילתות ריקות או בעיות טכניות.

הודעות אלו מספקות הסבר על הבעיה ומציעות פתרון, כמו הזנת שאילתא חדשה.

4. **שליטה בנתונים הנאספים:**

הנתונים שנאספים קשורים לשאילתות החיפוש בלבד ואינם כוללים מידע אישי של המשתמש.

למנהל המערכת קיימת אפשרות לעדכן או להסיר אינדקסים באופן ידני במסך ה-Admin, תוך שקיפות מלאה על השינויים שבוצעו.

5. **מניעת חשש משגיאות או אי-הבנה:**

בעת כשלון בביצוע חיפוש (למשל, אם המונח שהוזן אינו מופיע במאגר), מוצגת הודעה המפרטת כי לא נמצאו תוצאות ומזמינה את המשתמש לנסות שוב.

התצוגה של סטטיסטיקות ותדירויות ברורה ואינה משאירה מקום לפרשנויות שגויות.

**סיכום:**

באמצעות מתן משוב ברור, הצגת תוצאות וסטטיסטיקות בצורה שקופה ונגישה, ומתן שליטה למשתמש על חוויית השימוש, יישמנו את עקרון השקיפות האלגוריתמית במערכת. בכך, הבטחנו שהמשתמשים יבינו את תפקוד המערכת ואת הנתונים המוצגים להם, ויוכלו לפעול בביטחון מלא.

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

פערי ידע בנוגע לפיצ'ר – על מנת לשלב את פיצ'ר נגן המדיה היינו צריכים להשקיע זמן בלהבין איך ניתן לעשות זאת

עבודה עם Firebase – היה צורך להבין את השימושים של פונקציות המכניסות מידע למסד הנתונים כמו put, post. בנוסף היה צריך להבין כיצד לשלוף את הנתונים בהינתן האופן בו בנינו את מסד הנתונים. בהתחלה כשהיינו מכניסים או מוחקים מידע ממסד הנתונים המחיקה לא הייתה מלאה או שהאינדקס היה נכתב מחדש עם המחיקה למקום הלא נכון.

סינון וסידור אינדקסים – נוספו המון אינדקסים למסד הנתונים שאינם מתאימים למטרה שלו, כגון אינדקסים של מספרים, ג'יבריש, מונחים חסרי משמעות וכד'. היינו צריכים להבין כיצד לסנן אותם כדי שלא יופיעו מחדש.

עבודה בקולאב – מכיוון שכל הקוד נמצא במקום אחד היינו צריכים לגלול המון פעמים בין חלקי קוד על מנת להגיע לחלק הרלוונטי שרצינו לעדכן. בנוסף עבודה במקביל הצריכה תיאום כדי לוודא שהעבודה של אחד לא נדרסת על ידי האחר.

***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.

**ניתן למצוא בקובץ App Developer Guide שמצורף ב-zip.**

1. יש לבנות תיק למשתמש , הכולל הסבר כללי על המערכת , פירוט מסכים, מעברים בין מסכים והסבר על טעויות אפשרויות.

**ניתן למצוא בקובץ App User Guide שמצורף ב-zip.**

1. התייחסו בתיק המשתמש לשקיפות אלגוריתמית – כיצד הבהרתם למשתמשים את האלגוריתמים והנתונים הנאספים בקוד שלכם?

הבהרת האלגוריתמים והנתונים הנאספים לקוד שלנו הושגה באמצעות שקיפות המובנית במספר אלמנטים בתיק המשתמש:

1. **הוראות מפורשות לכלי הניהול**:

בדף הניהול מפורטת התנהגות המערכת, כולל שימוש בסיסמא לצורך גישה לפונקציות קריטיות, כמו ניהול אינדקסים וניהול הפלייליסט במוזיקה. ההסברים מספקים למשתמשים הבנה על פעולותיהם והשפעתם על המידע המאוחסן.

2. **פאנל סטטיסטיקה להנגשת נתונים**:

עמוד הסטטיסטיקה מציג למשתמשים גרפים ודוחות על הנתונים שנאספו, כמו "Top 10 Terms Frequencies" ו-"Top 10 Searched Queries". המשתמשים יכולים לבחור את סוג התצוגה (גרף עמודות, קו, או דונאט) כדי להבין טוב יותר את הנתונים הנאספים.

3. **שקיפות בצ'אט-בוט:**

הצ'אט-בוט משמש כאמצעי חכם לחיפוש וניתוח מסמכים הקשורים ל-Oracle. הוא כולל פאנל הוראות ברור, ומסביר כיצד יש להשתמש במערכת. אם יש שאלה לא ברורה, הבוט מבקש מהמשתמש לנסח מחדש את בקשתו.

4. **שימוש בצבעים ובמשוב מידי:**

פעולות כמו הוספה (ירוק) ומחיקה (אדום) מסומנות בצבעים כדי להבהיר את כוונת הפעולה. בנוסף, הודעות על שגיאות, כמו סיסמא שגויה או שדות ריקים, מספקות משוב מיידי המדריך את המשתמש לפעולה תקינה.

5. **שיפור הבנה באמצעות טיפים למשתמש:**

בתיק המשתמש שולבו טיפים לחיפוש יעיל, כגון שימוש במילות מפתח מדויקות או בדיקת נתונים קרובים במקרה של תוצאות לא מספקות.

1. 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.

**קישור למחברת ב- Collab –**

<https://colab.research.google.com/drive/1Bl0xHpy2kf3zsoc0C-7xuwSpTWJ0ODSc#scrollTo=kQ22Gci6XAkV>

**קישור ל-GitHub -** <https://github.com/muffindonor/cloud_computing_project.git>

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!

A badger looking through a magnifying glass

Description automatically generated

**App User Guide**

**Index**

[The Primary Page 14](#_Toc188575545)

[Main Interface Overview 14](#_Toc188575546)

[Main Components 14](#_Toc188575547)

[Instructions for the Navigation Buttons 15](#_Toc188575548)

[Admin Page 16](#_Toc188575549)

[Admin Dashboard Overview 16](#_Toc188575550)

[Main components before browsing the admin page 16](#_Toc188575551)

[Main Components 16](#_Toc188575552)

[Statistics Page 18](#_Toc188575553)

[Statistics Interface Overview 18](#_Toc188575554)

[Main Components 18](#_Toc188575555)

[Chat Assistant 19](#_Toc188575556)

[Chat Assistant Interface Overview 19](#_Toc188575557)

[Main Components 19](#_Toc188575558)

[Search Result Page 20](#_Toc188575559)

[Search Results Page Interface Overview: 20](#_Toc188575560)

[Main Components 20](#_Toc188575561)

[Design Notes 21](#_Toc188575562)

[Important Notes 21](#_Toc188575563)

## The Primary Page

**A screenshot of a computer

Description automatically generated**

### Main Interface Overview

This interface acts as the primary platform for browsing the application, giving users the ability to see statistics data, execute operational assignments, do searches, and play music. Quick access buttons for the "Admin Page" and "Statistics Page," a small music player, and a prominent search bar with possibilities for refining queries with an option of "AND" or "OR" for Boolean filtering are all included in its user-friendly, simplified design.

### Main Components

**Top Left Navigation:**

Admin Page Button: Links to the admin page to provides access for editing indexes and music player.

Statistics Page Button: Links to statistical graphs based on data extracted from the database.

Chat Assistant Button: Links to a chatbot allowing for real-time chat for guidance and support.

**Center Area:**

Logo: Displays a badger with a magnifying glass, emphasizing the application's search focus.

Search Bar: Enables users to input search queries, accompanied by a dropdown list to select between "AND" and "OR" operators to apply Boolean filtering to the search results.

Search Button: A bold orange button labeled "Badger Search", used to start the search process.

**Top Right Corner:**

Music Player: Includes playback controls to listen to songs in the background while using the search engine.

### Instructions for the Navigation Buttons

**Admin page Button**: Accesses the administrative panel.

Use: Click "Admin Page" to enter the Admin Page. Users will be required to insert a password prior to the page updating. Once the correct password is inserted, the user will be able to manage indexes and edit the music player’s playlist.

**Badger Search Button**: Initiates a search for specific queries.

Use: Click "Badger Search", after writing a query you want to search for, to perform a search. User will be moved to the results page displaying the query results.

**Chatbot Assistant Button**: Provides access to the chatbot.

Use: Click "Chat Assistant" to move to a page dedicated for conversing with the Chat Assistant.

**Statistic Page Button**: Provides access to the Statistic Page.

Use: Click "Statistics Page" to move to the dedicated page displaying various graphs depicting different data statistics.

## 

## **Admin Page**

A screenshot of a computer

Description automatically generated **A screenshot of a computer

Description automatically generated**

Admin Dashboard Overview

The Admin Dashboard allows to manage the indexes stored in the database and music player’s playlist. This interface is for the use of admins only and requires a password to access.

### Main components before browsing the admin page

(After clicking button Admin page at the Main Page)

Center area of the main page at bottom:

Password Required Bar: Enables users to input password.

Submit Button: A bold orange button labeled "Submit" is used to check the password. If the password is correct, it opens the Admin Page. If the password is incorrect, an alert is displayed notifying the user of the incorrect password.

### Main Components

**Tab Navigation:**

Index Management Tab: Displays tools for handling indexes and their documents.

Music Management Tab: Display tools for editing music player’s playlist.

**Index Management Section:**

Select Index Dropdown: Allows to choose an index from the database to edit.

Select DocID Dropdown: Provides the ability to select specific documents within the chosen index. Only becomes available once an index is chosen

Enter New Document URL Bar: Enables adding a new DocID by inputting the URL of a new document to be added to the selected index. Only becomes available once an index is chosen.

**Action Buttons:**

Add Document: A bold green button. Adds the entered document URL to the chosen index’s list of DocIDs in the database. Only becomes available once an index is chosen.

Delete Document: A bold red button. Removes the selected document from the index. Only becomes available once an index is chosen.

Delete Index: A bold red button. Deletes the currently selected index and all related data from the database. Only becomes available once an index is chosen.

**Music Management Section:**

Select Song Dropdown: Allows administrators to choose a song from the playlist.

Enter New Song URL Bar: Enables inputting a URL of a new song to be added to the selected songs.

**Action Buttons:**

Add Song: A bold green button. Adds the URL of a song entered to the ‘New Song URL bar’ to the media player’s playlist.

Delete Song: A bold red button. Removes the selected song from the media player’s playlist.

**Navigation Button:**

Back to Search: Returns the user to the main interface page.

## Statistics Page

A screenshot of a graph

Description automatically generated

### Statistics Interface Overview

The Statistics Interface to provides users with insights into search data, indexes and their frequency in Oracle’ webpages. It allows for the visualization of various metrics in different formats.

### Main Components

**Top Section**:

**Dropdown list**: Allows the user to choose which metric they’d like to view. The default metric is ‘Top 10 Terms’ frequencies”, choosing a different metric will update the graph accordingly. There are several analysis options for the user to choose from:

Top 10 Terms Frequencies.

Top 10 Terms with Most URLs.

Top 10 Searched Queries.

**Graph Type Selector**: Enables users to choose between different visualization formats, color-coded and labeled to improve user’s understanding of the graphs. Default graph is presented as “Bars”, by clicking on a different option the graph’s presentation will be updated accordingly. The presentation options are:

Bar Chart.

Line Chart.

Doughnut Chart.

**Navigation Button**:

**Back to Search**: Returns the user to the main search interface.

**Music Player (Top Right)**:

Features playback controls for added fun.

## **Chat Assistant**

**A screenshot of a computer

Description automatically generated**

### Chat Assistant Interface Overview

The chat assistant Interface is designed to provide users with an interactive tool to search for and analyze terms in Oracle-related documentation, the interface is simple and user-friendly, allowing users to communicate with the bot through a text area.

### Main Components

**1.Top Section**:

The interface has a bright orange header labeled “Oracle Search Assistant”, ensuring clear purpose.

Instructional Panel Located centrally, this panel suggestions control on the features and capabilities of the assistant.

**2.Ask me Bar:**

Located at the bottom, enables user to input the question he wants, then press enter, so chat assistant will give the user the relevant information, and if it is not clear for the chat it will ask the user to write more clearly. Answers must adhere to the description provided by the chatbot.

**3.Navigation Button**:

**Back to Search**: Returns the user to the main search interface.

## ****A blue and white background Description automatically generated with medium confidence**Search Result Page**

## Search Results Page Interface Overview:

**After clicking "Badger Search" on the main interface page (with input in Search Bar, if the user didn't put input it will display an alert)** t**he user is moved the Search Results Page which is designed to display the outcomes of a user’s search query, providing a summary of results and links to navigate through the results from the Oracle.**

### **Main Components**

**1.Result Summary**:

The page displays:

Message summarizing the results.

Total number of relevant results found.

List of visited links is provided based on the search query, and each link is with its frequency of appearance, shown on the right.

**When the user clicks on link, a new page will open.**

**2.Top Right Corner:**

Music Player: includes playback controls to listen to songs in the background during using the website.

**3.Navigation Button**:

**Back to Search**: Returns the user to the main search interface.

## Design Notes

1.Color-Coding: Green is used for ‘adding’ actions, indicating creation or positive changes. Red is used for ‘deletion’ actions, signaling potential risks. Dark theme option in the website to reduce eye exposing to light and enhance focus.

2.Centerd Content**:** The panel and input bar are centered to maintain user focus and guide the interaction process.

3.Clear Instructions:The chatbot assistant’s capabilities are explicitly outlined, ensuring users know how to interact with the interface effectively. Text areas all include a message to help the user understand their purpose.

4.Real\_Feedback**:** The chatbot assistant’s replies immediately to a user’s input and will alert them if their input is incorrect.Alerts **for invalid or empty queries help the user understand the system and guide them towards valid inputs.**

## Important Notes

To access admin page, use the password ‘123456’

**Tips for Effective Searching:**

Use specific keywords rather than broad terms to increase the relevance of your search results.

If you don't find what you're looking for on the first attempt, try using synonyms or related terms.

If you enter incorrect or incomplete data, such as a wrong missing information, the system may not behave as expected. And it will alert of something wrong.

**Interacting with the Chatbot:**

Start by reading the initial message from the chatbot. It will often greet you and ask how it can assist.

Type your question or request in the message input field.

Be clear and specific in your messages to ensure the chatbot can provide accurate assistance.

After typing your message, press **Enter** to submit it.

The chatbot will respond based on the information it has, and you can continue the conversation as needed.



**APP DEVELOPMENT GUIDE**

Index

[Introduction 24](#_Toc188574169)

[Overview 24](#_Toc188574170)

[Purpose 24](#_Toc188574171)

[Main Files 24](#_Toc188574172)

[Installation 24](#_Toc188574173)

[Imports 24](#_Toc188574174)

[Data Fetch 25](#_Toc188574175)

[CSS 25](#_Toc188574176)

[JSON File 26](#_Toc188574177)

[URLs 26](#_Toc188574178)

[Firebase URL 26](#_Toc188574179)

[Web Page URL (Dynamic) 27](#_Toc188574180)

[Song URL (GitHub Hosted MP3s) 27](#_Toc188574181)

[Logo URL (GitHub Hosted PNG) 27](#_Toc188574182)

[Key Functions 27](#_Toc188574183)

[Search Index Management 27](#_Toc188574184)

[Fetching Pages for Indexing 27](#_Toc188574185)

[Search Execution 27](#_Toc188574186)

[Managing Search Queries 28](#_Toc188574187)

[Validation of Terms 28](#_Toc188574188)

[Merging Index Data 28](#_Toc188574189)

[Managing URLs 28](#_Toc188574190)

[Admin Dashboard Functions 28](#_Toc188574191)

[Chatbot Responses 28](#_Toc188574192)

[Statistical Analysis 29](#_Toc188574193)

[User Interface Management 29](#_Toc188574194)

[Music Player Functionality 29](#_Toc188574195)

[Interesting Design Patterns/Code Templates 29](#_Toc188574196)

[Use of ipywidgets 29](#_Toc188574197)

[Use of CSS for Interface Design 30](#_Toc188574198)

[Music Player 31](#_Toc188574199)

[Chatbot Page 31](#_Toc188574200)

[Statistics Page 32](#_Toc188574201)

[Admin Page 32](#_Toc188574202)

[Results Page 33](#_Toc188574203)

# Introduction

## Overview

The system includes a user-friendly interface with search capabilities, providing URLs to Oracle pages, statistics dashboards, a chatbot assistant, and a music player to enhance user experience. An admin interface enables efficient management of indexed data and music playlists.

## Purpose

This project is a search engine application designed to index, search, and evaluate web content. It allows users to perform searches with logical operators (AND/OR), view detailed results, and access analytics such as term frequencies and popular queries. The application integrates Firebase for data storage, ensuring dynamic updates.

# Main Files

## Installation

The "Installation" section outlines the commands needed to set up the required libraries and packages. This step is essential to organize the development environment correctly and ensure that all dependencies are installed for the application to work seamlessly.

For instance:

A close-up of a white background

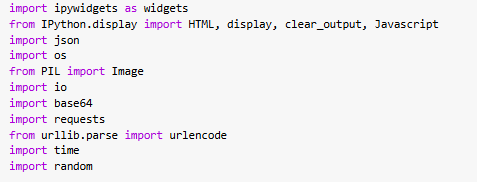
Description automatically generated

These commands install libraries for creating interacting with Firebase (firebase).

## Imports

The Imports section specifies all the libraries and modules included in the project. Each import is necessary to access specific features that the application relies on to perform its tasks effectively.

For instance:



These imports bring in tools for creating widgets, displaying outputs, plotting data, and interacting with APIs and databases.

## Data Fetch

This section focuses on fetching data from external resources. In this project, it primarily deals with obtaining data from Firebase. This functionality is critical as it supplies the application with the essential information required for analysis, processing, and display.

For instance:

A screenshot of a computer program

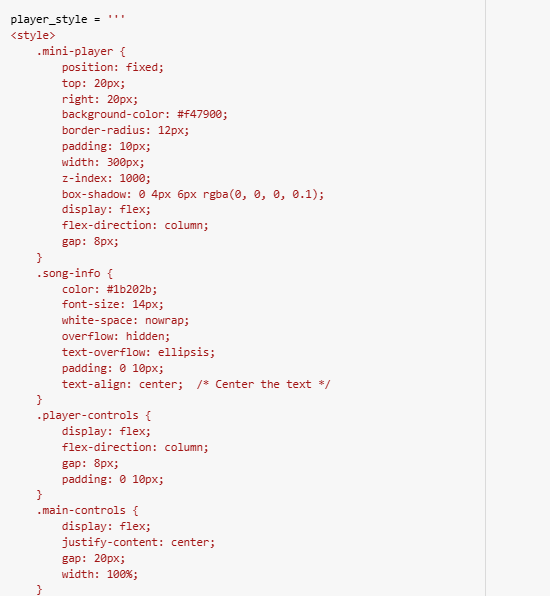
Description automatically generated

This method retrieves and processes the most frequently saved queries from Firebase.

## CSS

This section includes styles that edits the application's user interface. These styles are essential for creating a uniform and aesthetically pleasing design, ensuring that the UI components are visually appealing and user-friendly.

For instance:



## JSON File

The JSON File section focuses on working with and presenting JSON data files. These files are used to organize and maintain structured data, enabling efficient handling, interaction, and fitting of the data based on user actions or inputs.

For instance: we used to store the list of songs for the music player.

A close-up of a white sheet

Description automatically generated

# URLs

## Firebase URL

This URL connects the app to the Firebase Realtime Database. It is used for storing, retrieving, and updating indexed data, search queries.



The URL initializes the Firebase connection to interact with the Database.

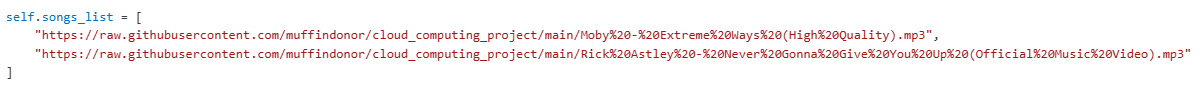
## Web Page URL (Dynamic)

This URL isfetched based on the base\_url parameter provided to the fetch\_pages() function. It represents the web pages that app crawls and indexes for content. Links are extracted from the HTML <a> tags on the base page using BeautifulSoup.



## Song URL (GitHub Hosted MP3s)

This URL point to MP3 files hosted on GitHub and are used in the music player feature IN the application.



## Logo URL (GitHub Hosted PNG)

This URL points to an image file (logo) hosted on GitHub. It is loaded and displayed in the app's user interface.



# Key Functions

## Search Index Management

**def build\_index(self) - (SearchEngine class)**

The function creates and updates the word location index for efficient search.

It clears the current index and rebuilds it from fetched pages. It processes words, validates terms, and updates the index in Firebase.

## Fetching Pages for Indexing

**def fetch\_pages(self, base\_url, num\_pages=5) - SearchEngine class**

This function Crawl web pages to extract content for indexing.

Without this, the system wouldn't have any data to build the search index.

It fetches and processes up to `num\_pages` from the given `base\_url`. Extracts visible text content for indexing.

## Search Execution

**def search\_query(self, query, search\_type='AND') - SearchLogic class**

This function is the core of the search engine, returning meaningful results based on user's queries.

It processes the query, applies AND/OR logic, and ranks results by relevance.

## Managing Search Queries

**def save\_query(self, query) - SearchLogic class**

This function saves, update, and analyze user search queries.

It saves user queries to Firebase, incrementing their count if they already exist.

**def fetch\_top\_queries(self, limit=10) - SearchLogic class**

This function Retrieves the most frequently searched queries.

## Validation of Terms

**def is\_valid\_term(self, term) - SearchEngine class**

This function ensures that only valid terms are included in the search index.

It validates terms based on rules:

1. Term must be longer than 2 characters.

2. Term must not contain numbers or special characters.

3. Term must only contain English letters.

## Merging Index Data

**def merge\_index\_data(self, existing\_data, new\_data) - SearchEngine class**

This function combines existing data from Firebase with new data from fetched pages.

It merges existing and new index data, ensuring no duplicate document IDs.

## Managing URLs

**def update\_index(self, term, added\_urls, removed\_urls, current\_docids)**

This function adds or removes URLs dynamically in the search index.

It updates the index with new or removed URLs.

## Admin Dashboard Functions

Provide admin with tools to manage the system.

**def delete\_index(self, term)**

This function deletes an entire index term and its associated data from Firebase.

**def refresh\_index(self)**

This function synchronizes local and Firebase index data.

## Chatbot Responses

Responds to user queries through conversation and provides helpful insights.

**def get\_most\_frequent\_term(self, groups)**

This function returns the term with the highest frequency in the indexed data.

**def get\_term\_pages(self, term)**

This function lists up to three pages containing the given term.

**def suggest\_search\_term(self, term)**

This function suggests related terms based on co-occurrence in documents.

**get\_term\_document\_count(self, term)**

This function count documents containing a specific term.

**get\_top\_terms(self, groups)**

This function list top 5 most frequent terms

## Statistical Analysis

Provides insights about the most frequent terms, URLs, and queries.

**def fetch\_top\_indexes(self)**

This function retrieves the terms with the highest total frequency.

**def fetch\_top\_index\_URL(self)**

This function retrieves the terms with the highest number of associated URLs.

**def fetch\_top\_queries(self, limit=10)**

This function Retrieves the most frequently searched queries.

## User Interface Management

Facilitates interaction between users and the system.

**def show\_search\_screen(self)**

This function displays the main search interface.

**def show\_results\_screen(self, query, results)**

This function displays search results based on user queries.

**def show\_admin\_page(self)**

This function displays the admin dashboard for managing the system.

## Music Player Functionality

Enhances user experience with a built-in music player.

**def load\_songs(self)**

This function loads a playlist of songs from a JSON file.

**def create\_mini\_player(self)**

This function generates a widget for the music player.

# Interesting Design Patterns/Code Templates

## Use of ipywidgets

Use of ipywidgets: The code extensively uses ipywidgets to create interactive elements in the notebook interface.

For instance: The code makes wide use of ipywidgets to build interactive components for the notebook interface. These widgets allow dynamic user interactions, making the application more engaging and user-friendly. For instance:



This code creates a Back button styled with a warning theme that navigates to the search screen when clicked and a text input field for user queries with a placeholder and custom width. Both widgets enhance interactivity in the user interface.

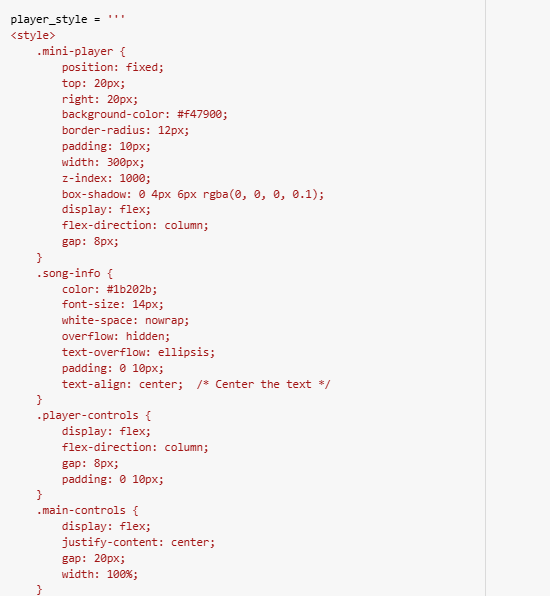
## Use of CSS for Interface Design

It is used to enhance the appearance of the web application, ensuring the interface is visually appealing and easy to navigate.

**Explanation:**

* CSS is used to design the structure and appearance of web elements.
* It helps control features like colors, fonts, spacing, and alignment of components in a consistent way.
* Common approaches include creating reusable style definitions, using responsive techniques to adapt to different screen sizes, and tools like colored backgrounds or styled components for dynamic designs.

For instance:



## Music Player

Enhances user experience with a simple music player that allows playback of a pre-defined playlist.

It Loads songs from a JSON file and displays them in a mini-player format.

Allows users to shuffle and play songs interactively.

A screenshot of a computer program

Description automatically generated

## Chatbot Page

The chatbot provides conversational assistance for users, helping them understand search trends, related terms, and system functionality. It implements pre-defined patterns to respond to user queries. Handles tasks like showing frequent terms, suggesting related terms, and explaining system capabilities.

A computer screen shot of text

Description automatically generated

## Statistics Page

The statistics page visualizes the most frequent terms, URLs, and user queries in a dashboard format.

It fetches data and, displays charts showing term frequency, URL usage, and query trends.

A screenshot of a computer program

Description automatically generated

## Admin Page

The admin page allows administrators to manage the index and music playlist, including adding, updating, or deleting terms and their associated URLs.

It provides dropdowns to select indexes and associated DocIDs. Includes input fields for adding new URLs. Allows deletion of specific DocIDs or entire indexes.

A screenshot of a computer program

Description automatically generated

## Results Page

The results page displays the search query results, including relevant links and their metadata.

It uses search\_query results to display matching documents. Shows the number of results, their frequency, and titles. Provides a back button to return to the search screen.

A screenshot of a computer program

Description automatically generated