

🔑 הערכה עצמית – מטלה מספר 5 - אפשרות מספר 3

אפשרות מספר 3

מזהה קבוצה:

30af9fe9-f034-4e99-972f-319b8171fb1d

תעודות זהות:

206591596

205965437

קישור לריפו:

<https://github.com/hilibar1/3997-ollama-rag>

הציון העצמי שלי: 96 / 100

◆ תיעוד הפרויקט (PRD + ארכיטקטורה) – 20/20

- PRD מקיף (docs/PRD.md): מסמך דרישות מוצר מפורט המכסה את כל הניסויים (1-4), יעדים, מדדים ומטרות פדגוגיות
- תיעוד ארכיטקטורה מקיף (docs/architecture_experiments_1_2.md): תיעוד טכני מעמיק עם דיאגרמות, החלטות עיצוב ותיאורי רכיבים
- מדריכי ניסוי ספציפיים: מדריכים מפורטים נפרדים לניסוי 3 (אינטגרציה ענן) וניסוי 4 (שליפה קונטקסטואלית ודירוג מחדש)
- תיעוד קונפיגורציה (docs/configuration.md)

◆ README ותיעוד קוד – 14/15

- README מקיף: מובנה היטב עם סעיפים ברורים להתקנה, שימוש ופתרון בעיות
- הוראות שלב-אחר-שלב: הוראות התקנה מפורטות
- דוגמאות שימוש: מספר דוגמאות המציגות כיצד להריץ ניסויים שונים
- תיעוד קוד: docstrings עקביים לכל הפונקציות והמחלקות
- תיעוד נוסף עבור הניסויים תחת docs

◆ מבנה הפרויקט ואיכות הקוד – 14/15

- עיצוב מודולרי: רכיבי RAG מופרדים בצורה נקייה
- הפרדת אחריות: לכל מודול אחריות יחידה ומוגדרת היטב
- שמות עקביים: עקיבה אחר PEP 8
- עקרון DRY, פונקציונליות משותפת הוצאה ל-utilities
- חלק קטן מהקוד האינטגרטיבי יכול להיות מפורק לתת-פונקציות כדי לשפר קריאות, כדוגמת קוד הניסויים עצמו, אבל בחרנו להשאיר אותו פחות מודולרי על מנת לשפר את הקריאות שלו (מאחר ומדובר בניסוי שרץ כסקריפט).

◆ קונפיגורציה ואבטחה – 9/10

- תצורת YAML עם דריסות משתני סביבה ותמיכה בארגומנטים של CLI
- כל התצורה הרגישה (מפתחות URLs, API) משתמשת במשתני סביבה. שימוש ב-`env.example`.
- `gitignore` מקיף

◆ בדיקות ואיכות – 14/15

- סוויטת בדיקות מקיפה: 97 בדיקות יחידה המכסות את כל רכיבי ה-RAG הליבתיים
- כיסוי גבוה עבור מודול ה-RAG הליבתי
- אסטרטגיית בדיקות ממוקדת: מדידת כיסוי מתמקדת ב-`/src/rag` (ליבת ה-RAG), למעט סקריפטי אינטגרציה
- בדיקת מקרי קצה: בדיקות מכסות תנאי שגיאה, קלטים ריקים, קבצים חסרים, פרמטרים לא חוקיים
- שימוש נכון ב-mocks: ליבוד יחידות תחת בדיקה
- ארגון בדיקות: מבנה הבדיקות משקף את מבנה המקור

◆ מחקר וניתוח תוצאות – 15/15

- תוצאות ניסויים: 15 קבצי תוצאות ב-`/results/experiments`
- דוחות השוואה: `results/comparison/compare_*.md`
- סיכום: `results/SUMMARY.md` המרכז את כל הממצאים
- ויזואליזציות: תיקיית `/results/visualizations`
- ניסויים שיטתיים: 15 ריצות ניסוי מלאות על פני כל 4 סוגי הניסויים
 - וריאציות פרמטרים מרובות:
 - ניסוי 2: בדיקת גודל LLM, מודלי embedding, פרמטרי chunking
 - ניסוי 3: השוואה מקומי מול ענן (מצבים A, B, C עם OpenAI)
 - ניסוי 4: בדיקת שליפה קונטקסטואלית ודירוג מחדש
 - תוצאות מפורטות: ניסוי כולל מדדים ב-JSON + ניתוח Markdown
- דוחות השוואה: 4 מסמכי השוואה המנתחים tradeoffs
- מדדים במעקב: זמן אינדוקס, latency שאילתה, שימוש בזיכרון, דיוק תשובה, עלויות טוקנים

◆ חוויית משתמש והרחבה – 10/10

- CLI אינטואיטיבי: ממשק שורת פקודה ברור עם ברירות מחדל הגיוניות
- ארגומנטים גמישים: תמיכה בדריסת כל פרמטר תצורה דרך CLI
- logging התקדמות: משוב בזמן אמת במהלך אינדוקס, שליפה ויצירה
- הודעות שגיאה ברורות: הודעות שגיאה מועילות עם הצעות לתיקון
- ארכיטקטורה מודולרית: קל להרחבה עם רכיבים חדשים
- עיצוב מוכן פלאגינים: רכיבי RAG chunker, retriever, reranker ניתנים להחלפה

סיכום כללי

הפרויקט מציג רמה גבוהה בכל היבטי הפיתוח: תיעוד, ארכיטקטורה, איכות קוד, בדיקות, ניסויים אמפיריים וניתוח נתונים. הוא משלב בין הנדסת תוכנה מקצועית לבין מחקר אקדמי מבוסס-נתונים.

1. **שילוב בין עומק הנדסי לעומק מחקרי** – המערכת בנויה בצורה מודולרית, נקייה ומורחבת, תוך ביצוע ניסויים מתוכננים היטב וניתוחים סטטיסטיים שמציגים הבנה עמוקה של ההשפעות של כל פרמטר.
2. **הקפדה על תיעוד** – ה-PRD, מסמך הארכיטקטורה וה-README מספקים תמונה מלאה וברורה של כל מרכיבי המערכת. כל החלטה מוסברת, וכל רכיב מתועד ברמה שמזכירה מסמכי מוצר ותכנון בתעשייה.
3. **איכות קוד גבוהה ועקבית** – המבנה המודולרי, שמירה על סטנדרטים אחידים, והפרדה חדה בין שכבות הופכים את המערכת לקלה לתחזוקה ולהרחבה – מאפיין חשוב בפרויקטים אמיתיים.
4. **בדיקות תוכנה ברמה מקצועית** – מספר גבוה של בדיקות, כיסוי גבוה על החלקים הקריטיים ושימוש נכון במוקדים מצביעים על בגרות הנדסית והבנה של חשיבות בדיקות במערכות מבוצרות ורבות-רכיבים.
5. **חווייה למשתמש והרחבה עתידית** – CLI נוח, יכולת להחליף מודלים בקלות, קונפיגורציה גמישה ורכיבים הניתנים להחלפה.

במהלך התרגיל למדנו כיצד להשתמש בrag, להריץ ניסויים, ולהשוות בין שיטות שונות, כגון:

- בניית יישום RAG בסיסי מלא
- ביצוע ניסויים שיטתיים עם שינוי פרמטרים
- השוואת פריסה מקומית מול ענן
- יישום והערכת אסטרטגיות RAG מתקדמות (שליפה קונטקסטואלית, דירוג מחדש)
- ניתוח תוצאות עם טבלאות והשוואות
- הסקת מסקנות עצמאיות המבוססות על ראיות אמפיריות

הצהרת יושר אקדמי (Academic Integrity Declaration)

אני מצהיר/ה בזאת ש:

ההערכה העצמית שלי היא כנה ואמיתית

בדקתי את העבודה מול כל הקריטריונים לפני קביעת הציון

אני מודע/ת שציון עצמי גבוה יוביל לבדיקה דקדקנית יותר

אני מקבל/ת את העובדה שהציון הסופי עשוי להיות שונה מהציון העצמי

העבודה היא פרי עבודתי/נו (של הקבוצה) ואני/ו אחראים/ם לכל תוכנה

