

## 🔪 הערכה עצמית – מטלה מספר 3 - פיתוח מכונת טיורינג באמצעות סוכני AI פנימיים של CLI

מזהה קבוצה:

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

תעודות זהות:

206591596

205965437

קישור לריפון:

[/https://github.com/Almog16/3997-turing-machine](https://github.com/Almog16/3997-turing-machine)

הציון העצמי שלי: 94 / 100

### ♦ תיעוד פרויקט (PRD + ארכיטקטורה) – 20/20

- הפרויקט מכיל קובץ [architecture.md](#) ו-`adri`.
- הפרויקט מכיל קובץ [PRD.md](#) על דרישות הפרויקט
- [README](#) המרכזי של הפרויקט מכיל הסבר מפורט על התקנה, ומבנה הפרויקט. כמו כן הוא מכיל הסבר על `models cost`.
- תחת `docs` תוכלו למצוא עוד חומרים רלוונטים על הפרויקט.
- תחת `assets` תוכלו למצוא תמונות וסרטוני הפעלה.

### ♦ README ותיעוד קוד – 14/15

- README מקיף ומדויק עם הוראות הפעלה צעד-אחר-צעד, בדיקות, ניפוי שגיאות וצילומי מסך.
- תמיכה בשני מצבי הרצה, עם הסברים עבור שניהם.
- קוד פייתון מתועד עם `docstring`.

### ♦ מבנה פרויקט ואיכות קוד – 14/15

- מבנה הפרויקט פשוט וקל להבנה, במידת הצורך יש תיעוד מפורט על התקנה ושימוש.
- תחת `claude` תוכלו למצוא את כל ה-`mdm` הרלוונטים עבור ה-`agents`, כמו כן גם `commands` שנתמכים בעד שימוש ב-`claude code`.

### ♦ קונפיגורציה ואבטחה – 9/10

- תמיכה בקובץ `env.example` ושימוש נכון בקוד.

### ♦ בדיקות ואיכות (14/15) – Testing & QA

- סקריפט קל לשימוש לבדיקת rates שונים ביחד לcosin similary. הסקריפט מחשב מרחקים ומייצר גרף השוואתי בין error rates, כך שבציר ה-y תוכלו למצוא את cosine similarity ובציר ה-x את error rate בהתאמה.
- הוספת unit tests עבור test\_similarity and test\_error\_sweep עם coverage 98%.

### ♦ מחקר וניתוח – 13/15

- סקריפט קל לשימוש לבדיקת rates שונים ביחד לcosin similary. הסקריפט מחשב מרחקים ומייצר גרף השוואתי בין error rates, כך שבציר ה-y תוכלו למצוא את cosine similarity ובציר ה-x את error rate בהתאמה.
- מחברת analysis.py.

### ♦ ממשק משתמש והרחבה (10/10) – UI/UX & Extensibility

- תומך בשני ממשקי הרצה -
  - ממשק אינטרקטיבי דרך Claude Code עם הוראות התקנה ברורות.
  - הרצת script bash ע"י headless option של Claude.

### 🗨️ נימוק כללי

הפרויקט פותח מתוך מחויבות מלאה לעמידה בכל דרישות דף ההנחיות ובסטנדרטים של פרויקט תעשייתי. לאורך תהליך העבודה, הושקעו כ-1-2 ימי פיתוח אינטנסיביים שכללו אפיון, תכנון ארכיטקטוני, כתיבת קוד, בניית תשתית CLI, ניסויים אמפיריים, הפקת גרפים, וכתיבת תיעוד מקיף וברור.

במהלך הפיתוח העמקנו בשימוש ב-Claude Code וביכולות המתקדמות שלו — סוכנים (Agents), מיומנויות (Skills), ופקודות (Commands) — ובנינו תהליך מלא המדגים שרשרת תרגום רב-שלבית המבוססת על סוכנים עצמאיים. בנוסף, פותחו תרחישי הרצה מגוונים: מצב אינטראקטיבי דרך Claude Code ומצב Headless להרצה אוטומטית, לצד Bash scripts שמאפשרים חוויית שימוש מקצועית וגמישה.

העבודה על הפרויקט גם חידדה הבנה מעמיקה בעיבוד שפה טבעית, תרגום רב-שלבי, וניתוח מבוסס Embeddings, כולל חישובי Cosine Similarity וניהול ניסויים על רמות שונות של שגיאות כתיב. כל אלו תועדו בצורה סדורה בקבצי README, PRD, ADR וקבצי ארכיטקטורה.

לצד ההצלחה, זוהתה גם נקודה לשיפור עתידי: שימוש במודל מתקדם ואחיד יותר לצורך חישוב ה-Embeddings עשוי לשפר את עקביות ואיכות הבדיקות.

### 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

22.11.25

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