מערכת "יום הערכה" — מדריך הפעלה

מדריך זה כתוב בשפה פשוטה, צעד אחר צעד. אין צורך בידע בתכנות.
המערכת מיועדת ל-40 נבחנים במקביל: כל נבחן עובר 4 פרקי מבחן וריאיון אחד, לאורך 5 סבבים.

תוכן העניינים

1. בדיקה מהירה על המחשב שלך
2. העלאה לאוויר — שתי דרכים
3. עריכת שאלות (בנק התוכן)
4. מסך המנהל — ניהול היום
5. חזרה גנרלית לפני היום עצמו
6. סוף היום — הורדת תשובות ובדיקת AI
7. פתרון תקלות נפוצות

---

1. בדיקה מהירה על המחשב שלך

כך תוכל/י ללחוץ על המערכת ולראות אותה עובדת, עוד לפני שמעלים אותה לאוויר.

הדרך הקלה (דאבל-קליק):

1. בתיקיית app, לחץ/י דאבל-קליק על הקובץ הפעל שרת.command.
2. ייפתח חלון שחור (טרמינל) עם הכיתוב: *"מערכת יום הערכה פועלת"*. אל תסגור/י אותו — כל עוד הוא פתוח, המערכת פועלת.
3. פתח/י דפדפן (Chrome) והיכנס/י לכתובת: http://localhost:3000
4. כדי לעצור, סגור/י את החלון השחור.

בממשק אחד: הכתובת פותחת כברירת מחדל את מסך הנבחן. בפינה העליונה יש כפתור "כניסת מנהל" — לחיצה הכול
פותחת את מסך המנהל (סיסמת ברירת מחדל לבדיקה: admin), ומשם חוזרים בכל רגע ב-"חזרה למסך הנבחן". אין עליו
צורך לזכור כתובות שונות.

אם הדאבל-קליק לא עובד (דרך ידנית):

1. פתח/י את היישום Terminal (טרמינל).
2. גרור/י לתוכו את התיקייה app — ליד המילה cd — ולחץ/י Enter. (או הקלד/י cd ואז גרור/י את התיקייה.)
3. בפעם הראשונה בלבד הקלד/י והריצ/י: npm install
4. הריצ/י: npm start
5. היכנס/י בדפדפן לכתובת שלמעלה.

> הנתונים בבדיקה המקומית נשמרים בקובץ בתוך app/data/. אפשר למחוק אותו בכל רגע כדי להתחיל נקי.

---

2. העלאה לאוויר — שתי דרכים

יש שתי דרכים להריץ את המערכת ביום עצמו. בחר/י לפי המצב שלך.

דרך א' — רשת מקומית (מומלץ כשכולם באותו מקום)

הדרך הפשוטה והאמינה ביותר ליום אחד באולם אחד — בלי אינטרנט, בלי חשבונות, בלי תשלום.

1. מריצים את המערכת על מחשב אחד באולם (דאבל-קליק על הפעל שרת.command, כמו בסעיף 1).
2. מוודאים שכל 40 המחשבים מחוברים לאותה רשת Wi-Fi כמו מחשב השרת.
3. מגלים את כתובת הרשת של מחשב השרת: בהגדרות ה-Wi-Fi של המק מופיעה כתובת כמו 10.100.102.6. (החלון השחור בהפעלה גם יכול להציג אותה.)
4. כל נבחן נכנס בדפדפן לכתובת: http://<הכתובת-של-השרת>:3000 — לדוגמה http://10.100.102.6:3000.
5. המנהל נכנס מאותו מקום דרך כפתור "כניסת מנהל" בפינה.

> טיפ: בדקו את הכתובת ממחשב אחר לפני היום עצמו. אם משהו לא מתחבר — בדרך כלל זה חומת-אש או רשת אורחים שמבודדת מכשירים; השתמשו ברשת אחת רגילה.

דרך ב' — ענן (Render), כשהנבחנים במקומות שונים

נשתמש בשירות Render. פתיחת החשבון והזנת פרטי התשלום נעשים על ידך.

שלב א' — להעלות את הקוד ל-GitHub (פעם אחת):

1. פתח/י חשבון חינם ב-github.com.
2. צור/י מאגר (Repository) חדש, לדוגמה בשם yom-haaracha.
3. העלה/י אליו את תוכן התיקייה app (הקבצים שבתוכה). אפשר לגרור אותם דרך כפתור "Add file → Upload files" באתר GitHub.

- חשוב: להעלות את תוכן app בלבד (server.js, package.json, התיקיות /content/, public/, lib וכו') — לא את קבצי ה-PDF שמסביב.

שלב ב' — לחבר ל-Render:

1. פתח/י חשבון ב-render.com (אפשר להתחבר עם חשבון GitHub).
2. לחצ/י New → Blueprint, ובחר/י את המאגר שיצרת. Render יזהה לבד את קובץ ההגדרות render.yaml ויציע להקים שירות בשם yom-haaracha.
3. במסך שנפתח, תחת Environment Variables, הגדר/י את EXAMINER_PASSWORD — סיסמת בוחן משלך (לא admin!). זו הסיסמה שתפתח את מסך הבוחן.
4. לחצ/י Apply / Deploy. אחרי כמה דקות תופיע כתובת אינטרנט (למשל https://yom-haaracha.onrender.com).
5. בדוק/י שהכתובת נפתחת. זו הכתובת שתחלק/י לנבחנים ביום עצמו.

הערות חשובות:

- בחר/י בתוכנית Starter (מכונה קטנה שדולקת תמיד), לא ב-Free — התוכנית החינמית נכבית מעצמה אחרי חוסר פעילות ותאבד את הטיימר. זו עלות חודשית קטנה.
- קובץ ההגדרות כולל דיסק קבוע כך שהתשובות לא ייעלמו גם אם המכונה תופעל מחדש.
- מסך הבוחן נמצא תמיד בכתובת שלך עם /examiner בסוף.

---

3. עריכת שאלות (בנק התוכן)

כל שאלה היא נתונים, לא קוד — אפשר לערוך בלי לגעת במנגנון.

- כל פרק הוא קובץ נפרד בתיקייה app/content/ (בסיומת .json).
- כדי לערוך: פתח/י את הקובץ בעורך טקסט פשוט (כמו TextEdit במצב טקסט רגיל), שנה/י את הנוסח, ושמור/י.
- כדי להוסיף פרק: העתק/י קובץ קיים, שנה/י את השם ואת התוכן. השתמש/י בקובץ קיים כתבנית מחייבת.
- נוסחאות נכתבות בפורמט KaTeX בין הסימנים \(... \) (בשורה) או \[... \] (מודגש).

לדוגמה: \(\{f(x)=\dfrac{3x}{x^{2}+1}\). שים/י לב לשמור על הקו הנטוי ההפוך \ — בלעדיו הנוסחה תישבר.

- לשבר עם מונחים בעברית (בלי לוכסן) השתמש/י ב: [[frac:מסה|נפח]] — יופיע כשבר עם קו אמיתי.

בדיקת תקינות אוטומטית ("שומר הסף"):

המערכת בודקת שכל שאלת הוראה מנוסחת במילים בלבד (בלי "צייר", "נסח משוואה" וכו') ושכל נוסחה תקינה.

כדי לבדוק את כל הבנק ידנית. הריצ/י בטרמינל (בתור app):

```
npm run check-content
```

פרק עם שגיאה לא יעלה לאוויר עד שיתוקן. אפשר לראות את הדוח גם במסך הבוחן, בכפתור "תקינות בנק התוכן".

אחרי עריכה מקומית — העלה/י את הקבצים המעודכנים ל-GitHub, ו-Render יעדכן את האתר אוטומטית.

---

4. מסך המנהל — ניהול היום

העליונה של המסך יש כפתור "כניסת מנהל" — לחיצה עליו והזנת סיסמת הבוחן פותחות את מסך הניהול. (אפשר בפינה

גם להיכנס ישירות לכתובת שמסתיימת ב-/examiner.)

פתיחת משתמשים מראש (מומלץ לפני היום): בחלק "ניהול נבחנים" אפשר לפתוח לנבחנים משתמשים מראש —

- נבחן יחיד: שם + קוד (ואם רוצים גם סבב ריאיון ומקצועות).
- רשימה שלמה: מדביקים את כל הנבחנים בבת אחת, שורה לכל אחד בפורמט שם, קוד. אפשר להוסיף עמודה שלישית לסבב הריאיון: שם, קוד, סבב.

ביום עצמו הנבחן פשוט מזין את השם והקוד שקבעת. אם לא בחרת לו מקצועות מראש — הוא בוחר בעצמו בכניסה כי בחירת המקצועות היא חלק מההערכה). זה גם המקום להוסיף כמה נבחנים כדי לבדוק את המערכת (וכפתור (מומלץ,

× מסיר נבחן בדיקה).

תכנון הריאיונות (מי מתראיין בכל סבב)

יש 5 סבבים, ובכל סבב חלק מהנבחנים יוצאים לריאיון והשאר עושים פרק. בחלק "תכנון ריאיונות" במסך המנהל:

- קיבולת לכל סבב: לכל אחד מ-5 הסבבים יש שדה "קיבולת" = כמה מראיינים (מקומות ריאיון) יש באותו סבב. אפשר לשנות בכל רגע (בד"כ 8–10).

מי משובץ, לפי שם: כל כרטיס סבב מציג את שמות הנבחנים שמשובצים לריאיון באותו סבב, ואת המונה "משובצים

- / קיבולת". אם עברת את הקיבולת — הכרטיס נצבע אדום עם "חריגה" (השינוי מתקבל, זו רק אזהרה).
- קביעת השיבוץ: בטבלת הנבחנים למטה, בעמודה "ריאיון בסבב", בוחרים לכל נבחן את הסבב שלו (מראש או בלייב).
- "אזן אוטומטית (למי שאין סבב)": משבץ בלחיצה את הנבחנים שעדיין אין להם סבב, לסבבים עם מקום פנוי. "אזן הכל מחדש" מחלק את כולם מחדש שווה בשווה.
- מונה "נבחנים ללא שיבוץ" עוזר לוודא שכולם שובצו.

> חשוב לזכור: מספר מקומות הריאיון בכל הסבבים יחד צריך להספיק לכל הנבחנים (למשל 50 נבחנים → צריך בערך 10 מראיינים בכל סבב). עדיף לתכנן את הריאיונות לפני תחילת המבחן; שינוי בלייב בונה מחדש את הלוח של אותו נבחן.

- שחרור קוד סבב: לחיצה על "סבב 1" (וכן הלאה) פותחת לכל הנבחנים את המשבצת הבאה שלהם.

המערכת מנתבת כל נבחן אוטומטית לפרק הבא שלו, או לריאיון — לפי הלוח.

- בקרות לכל הנבחנים בבת אחת (מתחת לכפתורי הסבבים):
- ☐ השהה את כולם / ▸ המשך לכולם — עוצר ומחדש את הטיימר לכולם, למשל להפסקה כללית.
- סיים את המבחן (אדום) — סוגר את המבחן לכל הנבחנים לפני הזמן; כולם עוברים למסך "המבחן הסתיים".

להשתמש רק בסוף. אם נלחץ בטעות — יופיע כפתור "החזר לפעילות".

- טבלת הנבחנים: מציגה בזמן אמת איפה כל נבחן, כמה זמן נותר לו, כמה תשובות שמר, והתראות (אם עבר לטאב אחר או ניסה להדביק).
- עקיפות ידניות לכל נבחן (הכפתורים הקטנים בשורה):
- +2ד׳ — הוספת שתי דקות לטיימר.
- ☐ / ▸ — השהיה והמשך (למשל בזמן שהנבחן בריאיון).
- ↺ — איפוס המשבצת (מתחיל את הטיימר מחדש).
- סיים — סיום מוקדם של הפרק לאותו נבחן (עובר להמתנה לסבב הבא).
- הורד תשובות (Excel) — כפתור למעלה. שומר קובץ אקסל קריא עם כל התשובות (שם, מקצוע, שאלה, התשובה שנתנו, ולרב-ברירה גם אם צדקו). זה הקובץ לצפייה, לבדיקה ידנית או לשליחה. מומלץ ללחוץ עליו כמה פעמים במהלך היום כרשת ביטחון.
- JSON — כפתור נוסף למעלה; מוריד את אותם נתונים בפורמט המתאים לבדיקת ה-AI (סעיף 6). האוטומטית.

מה הנבחן רואה

- בראש המסך מופיעים שם הנבחן וכפתור "יציאה", כדי שתמיד יהיה ברור מי מחובר.
- בתחתית כל פרק יש כפתור "הגש פרק" — הנבחן לוחץ עליו כשסיים, ואז עובר להמתנה לסבב הבא (אפשר גם פשוט לחכות שהזמן ייגמר או שהבוחן יסיים).

חזרה אחרי ניתוק: אם הנבחן סוגר את הדפדפן וחוזר לאותו מחשב — הוא ממשיך אוטומטית. ממחשב אחר — הוא

- מזין שוב את אותו שם ואותו קוד וחוזר בדיוק לאותה נקודה. הכול נשמר בשרת, לא במחשב של הנבחן.

---

5. חזרה גנרלית לפני היום עצמו

לפני היום האמיתי, כדאי להריץ סימולציה שבודקת שהכול עובד ומתאושש מתקלות.

כשהמערכת פועלת (מקומית או ב-Render), הריצ/י בטרמינל (בתור app):

```
npm run rehearsal
```

הסקריפט ירשום 40 נבחנים מדומים, יעבור על 5 הסבבים, ישמור תשובות, ואז ינתק נבחן אחד ויחבר אותו מחדש כדי לוודא שהוא חוזר בדיוק לאותה נקודה. בסוף יוצג סיכום "עברו / נכשלו".

לבדיקה מול האתר בענן:

```
BASE=https://[redacted].onrender.com EXAMINER_PASSWORD=[redacted] npm run rehearsal
```

שבירה ידנית מומלצת: באמצע פרק, סגור/י את הדפדפן או נתק/י את האינטרנט לרגע, ואז היכנס/י שוב עם אותו שם וקוד — הנבחן צריך לחזור לאותו פרק, לאותה שאלה, ולאותו זמן.

בדיקת

לפני היום האמיתי — מחק/י את קובץ הנתונים כדי להתחיל נקי (ראה/י סעיף 7).

---

6. סוף היום — הורדת תשובות ובדיקת AI

שלב א' — הורדת התשובות (הדרך הפשוטה):
במסך המנהל לחצ/י על "הורד תשובות (Excel)". יישמר קובץ אקסל בשם assessment-answers.xlsx — שורה לכל תשובה, עם השם, המקצוע, נוסח השאלה, התשובה שהנבחן נתן, ולשאלות רב-ברירה גם עמודה "נכון". אפשר לפתוח, לקרוא, ולשלוח לבודקים. זה כל מה שצריך אם בוחרים לבדוק ולתת ציונים ידנית.

שלב ב' (רשות) — בדיקת AI אוטומטית:
זהו רק אם רוצים שהמחשב ייתן ציונים אוטומטית במקום בדיקה ידנית. הבדיקה קוראת את התשובות ומפיקה לכל ציון לכל שאלה, ציון לכל מקצוע, ציון מרוכב 0–100 (משוקלל לפי קושי — מתמטיקה 5 יח"ל שוקלת יותר), דירוג נבחן:
ואחוזון בתוך הקבוצה, והמלצה מילולית. לצורך זה מורידים תחילה את הגיבוי בכפתור "JSON".

- בלי מפתח API (מצב הדגמה): הריצ/י npm run grade. תקבל/י דוחות לדוגמה שמראים בדיוק את המבנה. ציוני שאלות ההוראה במצב זה הם הדגמה בלבד.
- עם מפתח API (בדיקה אמיתית):

השג/י מפתח מ-console.anthropic.com (זו "סיסמה" שמאפשרת לתוכנה לשלוח את התשובות ל-Claude ולקבל
1.
ציונים).
2. בתיקיית app, העתק/י את הקובץ config.example.json לשם config.local.json, והדבק/י בו את המפתח.
3. הריצ/י:

```
npm run grade
```

4. הדוחות ייכתבו לתיקיית /app/exports: קובץ report-all.json (כל הנבחנים, מדורגים) וקובץ נפרד לכל נבחן.

> רק שמירת התשובות חייבת להיות מיידית (וכך זה עובד). הבדיקה עצמה רצה בנוחות בסוף היום.

---

7. פתרון תקלות נפוצות

- "הכתובת http://localhost:3000 לא נפתחת" — ודא/י שהחלון השחור (הטרמינל) עדיין פתוח ומריץ את השרת.
- "נבחן יצא/התנתק" — הוא פשוט נכנס שוב עם אותו שם וקוד, וחוזר בדיוק לאותה נקודה. שום דבר לא אבד.
- "נבחן תקוע" — במסך הבוחן, השתמש/י בכפתורי העקיפה (הוספת זמן, איפוס משבצת).
- נוסחה מופיעה כטקסט מוזר עם \frac — כנראה נמחק בטעות הקו הנטוי \ בעת עריכה. הריצ/י npm run

.כדי לאתר את הבעיה check-content

להתחיל נקי (למחוק את כל הנתונים): כשהשרת כבוי, מחק/י את הקבצים בתוך /app/data (הקבצים בסיומת .db).

-

בפעם הבאה שתפעיל/י, המערכת תתחיל ריקה.

- לשנות את משך הפרק (ברירת מחדל 20 דקות): אפשר להגדיר משתנה סביבה SLOT_DURATION_SEC

(בשניות). ב-Render מוסיפים אותו תחת Environment Variables.

בהצלחה! לכל שאלה, הקבצים החשובים הם: server.js (המנוע), content/ (השאלות), public/ (המסכים).