שאלה 3- טפסים, לוגיקה, ו-DOM עם JavaScript בסיסי + מתקדם

סה"כ: 60 נקודות | סעיף א: 20 נקודות | סעיף ב: 40 נקודות

הנחיות כלליות (לשני הסעיפים):

- **יש** להשתמש אך ורק ב-HTML, CSS בסיסי ו- JavaScript.
- יש להקפיד על מבנה קוד תקני, שמות משתנים ברורים ופונקציות מופרדות.
- יש להקפיד על עיצוב רספונסיבי מלא- אין לאפשר גלילה אופקית באף רזולוציה.
- יש להשתמש בתיקיית images/ לשמירת תמונות מקומיות (אין לייבא תמונות מ-URL).
- **אין** להשתמש בספריות חיצוניות או ב-frameworks (כגון jQuery, Bootstrap).
- אין להשתמש ב-AI או בקוד שלא נכתב על-ידכם.
- אין להשתמש ב-Emoji , אנימציות, רכיבים מוכנים או תווים דקורטיביים- יש להשתמש רק בתגיות HTML מתאימות.
- הקפידו על אחידות בגובה/רוחב של כרטיסים, ריווחים (margin, padding) ופריסה.
- התוכן הוא **לבחירתכם** - כל עוד הוא אמין, מקורי, ומשרת את מטרת השאלה.

<u>סעיף א - טופס עם ולידציה ועדכון DOM מיידי (20 נקודות)</u>

בנו טופס HTML לפי ההקשר של הגרסה שקיבלתם.

הטופס יכיל לפחות **6 שדות שונים** שמתאימים לנושא הגרסה.

מבין השדות:

- לפחות שדה חובה אחד לפחות
- שדה חובה מסוג email
- לפחות **שדה אחד עם תנאי לוגי ייחודי**

(למשל: מספר בתחום מסוים, שדה שתלוי בערך של שדה אחר, תאריך עתידי בלבד וכו')

<u>דרישות פונקציונליות:</u>

כאשר לוחצים על כפתור השליחה:

- יש לבצע ולידציה לכל השדות הרלוונטיים.
- אם קיימות שגיאות- יש להציג אותן בצורה ברורה ונגישה בתוך הדף (אין להשתמש ב-alert).
- אם הטופס תקין- יש להציג הודעת הצלחה מותאמת בעמוד.

<u>דרישות תצוגה:</u>

- אין לשמור את הנתונים בפועל, הפעולה משפיעה רק על ה-DOM (ולא על storage).
- יש להקפיד על מבנה רספונסיבי מסודר, עם פריסה נוחה לקריאה.
- הוולידציה וההודעות יבוצעו ב-JavaScript בלבד (ללא HTML5 validation מובנה).

הטופס הוא נקודת פתיחה- חשבו היטב אילו שדות מתאימים לקונטקסט של הגרסה שקיבלתם, ואיך לגרום לו להיראות ולהרגיש אמין, אינטואיטיבי ומקצועי.

<u>סעיף ב - הרחבת הטופס לאפליקציה שלמה עם localStorage בטעינה (40 נקודות)</u>

בסעיף זה תרחיבו את המערכת שבניתם כך שתאפשר ניהול מתקדם של הנתונים.

פונקציונליות נדרשת:

1. **שמירה של פריטים חדשים:** כל שליחת טופס תיצור אובייקט חדש ותשמור אותו בתוך localStorage. האובייקט יוצג מיידית בעמוד.
2. **הצגת נתונים קיימים:** בעת טעינת הדף, יוצגו כל הנתונים השמורים.
3. **UI נדרש:** ממשק נוח, רספונסיבי ונגיש במיוחד כאשר יש כמות גדולה של נתונים.
4. **עדכון בזמן אמת:** כל פעולה (הוספה, מחיקה, שינוי) תעדכן גם את ה-DOM וגם את ה-localStorage.

פירוט **פונקציות נדרשות בקוד (ניתן להוסיף עוד בנוסף לאלו):**

- saveItem() - שמירה
- loadItems() - טעינה עם פתיחת העמוד
- renderItems() - רינדור התצוגה ל-DOM
- deleteItem(id) - מחיקה
- updateItem(id, changes) - עדכון שדה מסוים בפריט

בנוסף, עליכם להוסיף פונקציונליות מתקדמת אחת לפחות,
לפי ההקשר של הגרסה שקיבלתם. לדוגמה:

- סינון פריטים לפי שדה מסוים (קטגוריה / רגש / סטטוס / סוג תקלה)
- חישוב נתון כולל (למשל: סכום, ממוצע, אחוזים, התפלגות)
- סטטיסטיקות לפי סוגים (כמה פריטים מכל קטגוריה)
- שליטה על סדר ההצגה (למשל: מיון לפי תאריך או לפי עוצמה)
- סיווג צבעוני של פריטים לפי ערכים (רגשות, רמות קושי וכו')

הפעולה המתקדמת צריכה להיות חלק אינטגרלי מהממשק ולא רק פונקציה "בצד".
היא צריכה להשפיע על חוויית השימוש.

דרישות טכניות לסעיף ב:

- אין להשתמש ב-Cookies, sessionStorage או URL params.
- הנתונים ישמרו כ-**מערך של אובייקטים.**
- יש להקפיד על שימוש ב-event listeners תקינים, קוד קריא ותחזיק.
- יש לכתוב קוד מודולרי:
  - פונקציות נפרדות לכל פעולה: שמירה, טעינה, רינדור, מחיקה וכו'.
  - הפרדה בין הלוגיקה העסקית לבין עדכון ה-DOM.

טיפים:

- וידאו שהקוד נשאר קריא גם כשיש הרבה נתונים.
- כתבו קוד שניתן להרחיב בקלות.
- ביצעו בדיקה עצמית: רעננו את הדף- האם המידע עדיין מופיע?
- אל תעמיסו את המסך- אם יש הרבה מידע, חשבו איך לארגן אותו.

<u>גרסה 1 - מערכת בקשת לחשים אישית (למתחילים באקדמיה לקוסמות)</u>

הקשר: מערכת עבור סטודנטים באקדמיה לקוסמות, המבקשים ללמוד לחש חדש בהתאם לפרופיל האישי שלהם.

סעיף א - טופס בקשה ללחש

<u>דוגמאות לשדות אפשריים:</u>

- שם מלא
- כתובת אימייל
- סוג הלחש המבוקש (התקפי / הגנתי / חיזוק)
- רמת הידע הקיימת (1-10)
- האם קיים שרביט (כן/לא)
- מספר ניסיונות קודמים בלחש
- חתימה דיגיטלית של מדריך
- יום לידה אסטרולוגי

סעיף ב - ניהול בקשות לחש

<u>פונקציונליות חובה:</u>

- שמירת כל בקשה ב-localStorage
- הצגת כל הבקשות בעת טעינת הדף
- אפשרות למחיקת בקשה
- אפשרות לשינוי סטטוס הבקשה: "ממתינה", "מאושרת", "נדחתה"

<u>פונקציונליות מתקדמת לבחירה:</u>

- סינון לפי סוג לחש
- הצגת אחוזי הצלחה כלליים (לפי ניסיונות קודמים)
- סטטיסטיקה: כמה בקשות נשלחו לכל סוג לחש

גרסה 2 - מערכת פיקוח על תיבות דואר מדברות

הקשר: מערכת דיווח על שימוש לקוי בתיבות דואר חכמות ומתלוננות.

סעיף א - טופס דיווח

דוגמאות לשדות אפשריים:

- מזהה תיבה
- צבע התיבה
- טון דיבור (רגזן / ממלכתי / נעים)
- כמות הדואר שהוזנה
- האם התיבה נסגרה
- סוג המשתמש (אדם / חתול / פיה)
- כתובת אימייל לאיתור

סעיף ב - לוח תלונות

פונקציונליות חובה:

- שמירה של כל תלונה ב-localStorage
- הצגה של כל התלונות ברגע הטעינה
- מחיקת תלונה קיימת
- שינוי סטטוס: "תיבה רגועה" / "דורשת טיפול"

פונקציונליות מתקדמת לבחירה:

- סינון לפי רמת עצבנות
- ספירה: כמה תלונות לכל סוג משתמש
- חישוב ממוצע כמות דואר שנכנסה לפי צבע תיבה

<u>גרסה 3 - טופס מועמדות לפסטיבל "חידון שירים לבעלי כנפיים"</u>

הקשר: מערכת הרשמה לתחרות מוזיקלית למעופפים בלבד.

סעיף א - טופס הרשמה

דוגמאות לשדות אפשריים:

- שם פרטי ובמה
- כתובת אימייל
- סוג כנפיים (נוצות / עור / אור)
- סגנון מוזיקלי
- שיר נבחר
- ניסיון קודם בתחרויות
- האם הקול נבדק ע"י נשר מוסמך

סעיף ב - ניהול הרשמות

<u>פונקציונליות חובה:</u>

- שמירה של כל מועמד ב-localStorage
- טעינת כל המועמדים מהאחסון
- מחיקת מועמד
- שינוי סטטוס: "עבר אודישן", "לא עבר"

<u>פונקציונליות מתקדמת לבחירה:</u>

- סינון לפי סגנון שירה
- סינון לפי סוג כנף
- סטטיסטיקה: כמה מועמדים לפי סגנון או סוג כנפיים
- חישוב כמה עברו אודישן

גרסה 4 - דיווח תקלות בתעלות זמן

הקשר: מערכת ממשלתית לדיווח על כשלים במעברי זמן.

סעיף א - טופס דיווח תקלה

שדות אפשריים:

- מספר תעלה

  זמן יעד

  זמן נחיתה בפועל
- תיאור התקלה (רעד, חור בזמן, מפגש עם דינוזאור)

  האם יותר מאדם אחד חצה
- שדה אבטחה נוכחי
- כתובת אימייל לדיווח חוזר

סעיף ב - לוח תקלות זמן

פונקציונליות חובה:

- שמירה של כל תקלה ב-localStorage
- הצגה מלאה עם טעינת העמוד
- מחיקת תקלה
- שינוי סטטוס: "פתוחה", "טופלה"

פונקציונליות מתקדמת לבחירה:

- סינון לפי סוג תקלה
- סינון לפי זמן יעד
- חישוב ממוצע או סטיית זמן בין יעד לנחיתה

גרסה 5 - הצעות לקרקס הרגשות הנודד

הקשר: המשתמשים מגישים רעיונות למופעים חדשים לקרקס רגשי.

סעיף א - טופס הצעת מופע

שדות אפשריים:

- שם האטרקציה
- הרגש המרכזי (פחד / שמחה / בלבול / געגוע)
- האם מצריך השתתפות או רק צפייה
- ציוד נדרש
- דרגת אינטנסיביות (1-5)
- מגבלת גיל
- אימייל יוצר הרעיון

סעיף ב - מערכת ניהול הצעות

פונקציונליות חובה:

- שמירה של כל הצעה ב-localStorage
- הצגת כלל ההצעות בטעינה
- מחיקה של הצעה
- שינוי סטטוס: "אושרה", "נדחתה"

פונקציונליות מתקדמת לבחירה:

- סינון לפי רגש
- הצגה לפי דרגת אינטנסיביות
- ספירת הצעות לפי סוג רגש
- חישוב ממוצע דרגת אינטנסיביות