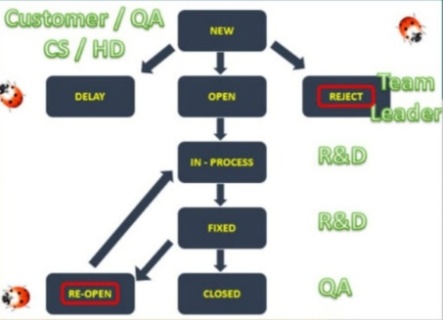
* **בדיקת שפיות –** בדיקה בסיסית ביותר שהאתר עושה את מה שהוא צריך לעשות בהגדרתו, בדיקה פונקציונלית בסיסית שהמערכת עובדת כמו שצריך
* **- Verification** וידוא שהמערכת עומדת בדרישות שהוגדרו. **Validation**- בדיקת התאמה בין התוצאה בפועל לדרישות המשתמש.
* **-Security Testing**בדיקות אבטחה להערכת עמידות המערכת בפני תקיפות, דוגמה: אימות זהות, הרשאות, לבדוק שהאדם הנכון נכנס ומה הוא יכול לראות.
* **Usability Testing**- בדיקות נוחות השימוש וההתמצאות בממשק המשתמש, נגישות לבעלי מוגבלויות.
* **Recovery & Survival Testing**- בדיקות התאוששות המערכת לאחר כשל או קריסה, דוגמה: שמירת מייל בטיוטות גם כשהמחשב כבה.
* **Regression Testing**- בדיקות לווידוא ששינויים בקוד לא פגעו בתפקוד קיים.
* **Performance Testing**- בדיקות עומס, ביצועים ומהירות טעינה, זמני תגובה, מטרתה למדוד ניצול CPU, זיכרון ואיתור צווארי בקבוק. לדוגמה, זמני תגובה של קבלת אישור עבור עסקת מזון בשופרסל. זמן התגובה אמור לקחת כמה שניות אחרת האפליקציה תחשב כמקרטעת.
* **Load Testing**- בדיקות עומסים להערכת ביצועי המערכת תחת כמות משתמשים גבוהה.
* **Interface Testing** - בדיקות ממשקים בין רכיבים שונים במערכת (חיצוניים), התממשקות בין מזג אוויר וטמפ', יש לוודא מי מוסר, ומי מקבל מידע, מה טיב המידע ובאיזה פורמט מגיע מצד לצד.
* **GUI Testing**- בדיקות ממשק משתמש לבדיקת תצוגה וקלות השימוש.
* **Functional Testing**- בדיקות פונקציונאליות להבטחת עמידה בדרישות המערכת, לדוגמה בדיקה שכפתור הוסף חבר בפייסבוק אכן מוסיף.
* **Compatibility Testing**- בדיקות תאימות לפלטפורמות שונות, תאימות של התוכנה לדפדפנים שונים.
* **Accessibility -**בדיקות לווידוא עמידה בדרישות נגישות למשתמשים עם מוגבלויות כמו לקות ראייה, שמיעה או רעד בידיים. נגישות היא חלק מבדיקות שימושיות, ונועדה לוודא שהמערכת תומכת בטכנולוגיות מסייעות, למשל תוכנות הקראה לעיוורים.
* **Static Testing**-בדיקות סטטיות המבוצעות ללא הרצה בפועל של המערכת, לדוגמה, בדיקה שהמסמכים כתובים היטב ואין פערי אפיון או תקלות באפיון ברמת התיעוד של המערכת מבלי שהרצנו אותה עדיין.
* **Acceptance Testing**- בדיקות קבלה לווידוא מוכנות המערכת לשימוש הלקוח, לדוגמה, בדיקת תהליכים עסקיים, אינטראקציות בין התהליכים השונים וכל תהליך עסקי בנפרד (כניסה לחשבון וקנייה בעגלה). Ebay -הרשמה, פרטי כרטיס אשראי, מציאת מוצר, תשלום, קבלת מוצר, חשבונית.
* **-STP- Software Test Plan**בדיקות תוכנית- מסמך תכנון הבדיקות המפרט את הבדיקות המתוכננות ושאינן מתוכננות, סיכונים, טכניקות הבדיקה, משאבי הבדיקה, אבני הדרך ולו"ז ליישום הבדיקות. מסמך זה נכתב בשלבים מוקדמים, במקביל לאפיון. אם הבודקים חסרי ניסיון -זהו סיכון שעלול לפגוע בפרויקט ויש להיערך אליו.
* **STR- Summary Test Results**- סיכום ממצאי הבדיקות - מסמך סיכום של ממצאי כל שלב בבדיקות. מסמך זה יכלול את המלצת צוות הבדיקות לגבי העברת המערכת לשלב הבא כולל הערכת הסיכונים.
* **STD- software Test Description** – תכנון מפורט של הבדיקות- אוסף מסמכי התכנון המפורט של הבדיקות, הכוללים את תרחישי הבדיקות, נתוני הקלט והתוצאות הצפויות.
* **ALM- Application Life cycle Management** - כלי לניהול תהליך של מחזור חיי התוכנה. התחלה- שלב הרעיון, שלב שבו התוכנה נכנסת לשימוש Deployment )) ושלב המוות, שבו אין לתוכנה ערך עסקי. פיתוח התוכנה מתבצע בתהליך שיטתי ולוגי המורכב משלבים מוגדרים היטב שאין לפסוח עליהם.
* **Entry and Exit criteria** –קריטריון כניסה = מה התנאים שניתן בגינם להתחיל את פרויקט הבדיקה. קריטריון זה נועד לוודא שהמערכת ראויה להיבדק לפני שאנו פותחים בפרויקט הבדיקה. למשל אם אני מסוגלת לחפש ולהעלות מוצר ביד 2 אפשר להתחיל להכניס את המערכת לבדיקות. קריטריון יציאה = באיזה תנאים נחליט שאפשר לסיים את הבדיקות. קריטריון זה קובע מהו מדד הסף שאם נעמוד בו נמליץ להעביר את המערכת לשלב הבא.
* **-Boundary Value Analysis (BVA)** בדיקת ערכי קצה- ערכי מינימום ומקסימום של טווח. לדוגמה, בדיקה בעת כניסה למייל /למערכת כאשר ממלאים את שדות: שם, גיל וכו' וכמובן גם במצב ריק.
* **Agile Testing**- בדיקות במסגרת פיתוח זריז וגמיש (Agile).
* **TDD (Test-Driven Development)**- פיתוח מונחה בדיקות שבו הבדיקות נכתבות לפני הקוד.
* **Debugging**- ניפוי כשלים ואיתור תקלות בקוד.
* **Black Box Testing**-בדיקות קופסה שחורה המתמקדות בקלט ובפלט בלבד. לדוגמה, בודק, ללא ידיעתו של המבנים הפנימיים של אתר, בודק את דפי האינטרנט באמצעות דפדפן; מספק פלטים ומאמת את הפלטים נגד התוצאה הצפויה (סוגים: שקילות, גבולות קצה, טבלאות החלטה, מעבר מצבים, מקרי שימוש).
* **White Box Testing**-בדיקות קופסה לבנה המתמקדות בלוגיקה הפנימית של הקוד. לדוגמה, גילוי מקרי קצה ושגיאות חבויות בקוד
* **בדיקת יציבות (Stability Testing)** -היא בדיקה שמטרתה לבדוק עד כמה המערכת מתפקדת לאורך זמן בתנאים שונים, כולל עומסים כבדים ושימוש ממושך, כדי לוודא שהיא לא קורסת או מאטה באופן משמעותי. האם המערכת יכולה להמשיך לפעול כראוי לאורך זמן ללא כשל?
* **Unit Testing**-בדיקות יחידה של רכיבים בודדים בקוד. דרישה יכולה להיבדק במגוון תרחישים. תרחיש יכול לכלול מס' דרישות שנבדקות באמצעותו.
* **Integration Testing**- בדיקות שילוב של מודולים שונים במערכת לדוגמה כאשר אנו מחברים את המקלדת למחשב – תהליך זה נקרא בדיקת יחידה. אך כאשר נרצה לחבר את המקלדת וגם את העכבר – תהליך זה נקרא אינטגרציה/ שילוב.
* **Maintainability Testing**-בדיקות תחזוקתיות להערכת גמישות הקוד לשינויים עתידיים.
* **Risk Management**- ניהול סיכונים בפרויקט והקצאת תגובות מתאימות, להעריך סיכונים אפשריים למוצר או לפרויקט ולהקצות דרכי תגובה לכל סיכון בכדי לצמצם נזק מוערך ממנו.
* **CRUD Testing** -בדיקות יצירה, קריאה, עדכון ומחיקה של נתונים במערכת. לדוגמה, אתר יד -2 בו ניתן ליצור מודעה ( Create), לכתוב בחיפוש ולבדוק אם קיימת האפשרות לקרוא את המודעה( Read), אם ברצוננו לעדכן את המודעה (Update )ולאחר שהמוצר נמכר או שמכל סיבה אחרת ברצוננו למחוק את המודעה מאיזו סיבה שהיא, נבדוק בחיפוש שהמודעה לא מופיעה יותר (Delete)
* **7 עקרונות בדיקה כלליים אוניברסליים:**  
  1. מערכת לעולם לא תהיה **חפה מפשע,** תמיד יהיו בה תקלות  
  2. יש הרבה מערכות מורכבות ואי אפשר לבדוק הכל אז נבדוק רק דברים **חשובים וקריטיים**  
  3. **שילוב בדיקות** צריך לעשות כמה שיותר מוקדם – כדי לעלות כמה שיותר מהר על תקלות  
  4. **תקלות מגיעות ביחד** – מצאתי תקלה באזור מסוים, סביר להניח שיש באותו אזור מקבץ תקלות  
  5. **פרדוקס ההדברה** – אם אבדוק את אותו אזור שוב ושוב בסוף לא אמצא בעיות. בדיקות חוזרות ונשנות על אותו אזור עלולות לאתר פחות תקלות עם הזמן. יש לגוון בין תסריטים ובדיקות על מנת למצוא תקלות חדשות, אחרת לא נמצא תקלות חדשות בהרצות החוזרות.  
  6. **בדיקות תלויות הקשר** – בדיקות נעשות באופן שונה על מערכות שונות בגלל אופיין השונה. בדיקה של אתר אינטרנט שונה מבדיקת משחק או אפליקציה  
  7. **אשליית היעדר תקלות** – גם אם בדקתי את המערכת ולא מצאתי תקלות זה לא אומר שהמערכת תצליח. יכול להיות שהיא עובדת אך לא מתאימה ללקוח
* **-Decision Table** טבלת החלטות לבדיקת מצבים שונים במערכת, ללכוד דרישות שמכילות תנאים לוגיים.
* **Risk-Based Testing (RBT)**- בדיקות המבוססות על הערכת סיכונים, טכניקה טובה להרצה סלקטיבית של מקרי בדיקה על פי הכרחיות הפונקציה הנבדקת.
* **High-Risk Areas**-זיהוי רכיבים קריטיים למערכת.
* **Cross-Browser Testing** -בדיקות תאימות בין דפדפנים שונים.
* **Mobile Testing**- בדיקות ייחודיות לעולם המובייל (רזולוציות, מערכות הפעלה, תקשורת רשת).

**אוטומציה בבדיקות -יתרונות**: חיסכון בזמן, עקביות, ביצוע בדיקות חוזרות במהירות. **חסרונות**: עלויות גבוהות בהקמה, תחזוקה מורכבת, לא מתאים לכל סוגי הבדיקות.

* **Authorization**- הרשאות למשתמשים במערכת, בניגוד ל-**Authentication** שמתמקד בזיהוי משתמשים, כניסה תקינה ולא תקינה למערכת.
* **State Transition Testing** -בדיקות מעבר מצבים לבדיקת זרימת הנתונים, בניית מקרי בדיקה מתוך מעברי המערכת הניתנים למימוש.
* **Stress Testing**-בדיקות עמידות המערכת בעומסים גבוהים, בדיקה של עומס שמורץ לאורך זמן ארוך בכדי לבדוק את עמידות רכיבי המערכת.
* **מחלקות שקילות (Equivalence Partitioning)-**שיטה שבודקת קבוצות שונות של נתונים כדי לוודא שכל הקבוצה פועלת בצורה דומה. במקום לבדוק כל אפשרות בנפרד, בוחרים נציג מכל קבוצה.
* **Exploratory Testing**- בדיקות חקרניות שמתבצעות ללא תסריט קבוע מראש, על בסיס אינטואיציה.
* **מחזור חיי התקלה (Bugs Life Cycle):**NEW-התקלה מדווחת לראשונה על ידי לקוח, צוות QA, צוות שירות לקוחות (CS), או מוקד תמיכה (HD). OPEN-התקלה נפתחת ונבדקת על ידי המנהלים הרלוונטיים או צוותי הפיתוח. POSTPONED/ DELAY- במידה שלא ניתן לטפל בתקלה מיידית, היא מושהית. REJECT-אם התקלה אינה תקפה, המנהל (Team Leader) דוחה אותה חזרה לבודק, מצביע על היכרות חלשה של הבודק את המוצר שכן התקלה שדווחה נדחתה ע"י בכיר. IN-PROCESS-התקלה נמצאת בתהליך טיפול פעיל על ידי צוות הפיתוח (R&D). FIXED-התקלה מתוקנת על ידי צוות הפיתוח. QA-התקלה נבדקת שוב על ידי צוות הבטחת האיכות (QA) לאחר התיקון. CLOSED-במידה שהתקלה נבדקה ואושרה כמתוקנת, היא נסגרת. RE-OPEN- אם התקלה אינה נפתרה באופן מלא, היא נפתחת מחדש לצורך טיפול נוסף.
* **בדיקות אלפא ובטא**. בבדיקות **אלפא** אני מביא את הלקוח אלי למפעל ושולט על האירוע, הוא מבצע את הבדיקות לידי – יתרונות- וודאות, דיסקרטיות. חיסרון- עלות גבוהה. בבדיקות **ביטא** שולחים את המוצר ללקוח לשימוש עצמי ולאחר זמן מה אקבל ממנו פידבק. יתרון- פשוט וקל למימוש. חיסרון- לא דיסקרטי. מתאים בעיקר למשחק מחשב



* 1. בדיקה בין ממשקים API:

מי מוסר, מי מקבל את המידע, מה טיב המידע ובאיזה פורמט

* 1. INTERFACE -מה זה?

תקשורת בין ממשקים חיצוניים

* 1. בדיקת קופסה שחורה מתבססת על:

בדיקות שמבוססות על קלט מול פלט, לא נראה את הקוד

* 1. בדיקת קופסה לבנה מתבססת על:

בדיקות המבוססות על מבנה הקוד הפנימי של התוכנה

* 1. מהו היחס בין דרישות לקוח מתועדות לתרחישי בדיקה שיש לכתוב ולהריץ?

תרחיש בדיקה יכול לכסות מספר דרישות, דרישה אחת יכולה להיבדק במגוון תרחישים

* 1. לאחר קריאת מסמכי אפיון מצאת תיאור של תהליך שמשמעותו אינה ברורה:

יש לפנות לאנליסט לגבי פער באפיון (פער באפיון = אנליסט)

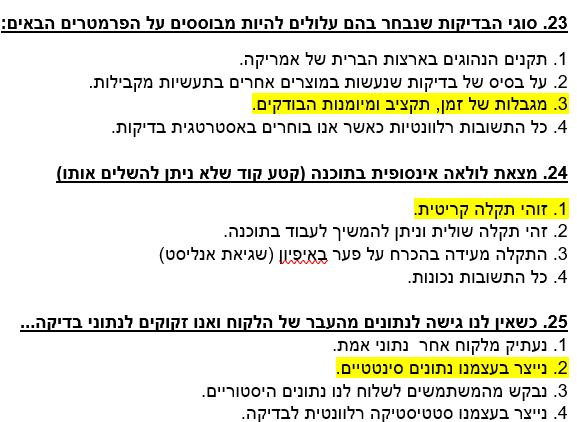
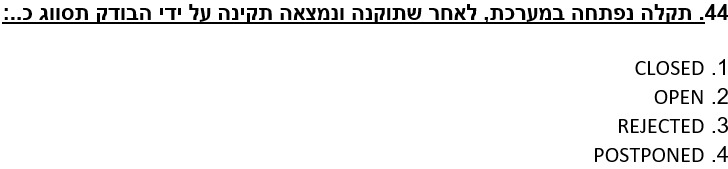
* 1. אם יש שאלה על מצב שההסבר לקוי – שהאפיון כתוב בצורה שגויה (אומרים איך זה ייראה אבל אני רואה שמדובר בתקלה או בעיה וחוסר הבנה. כשאני יודע מה המערכת תעשה, אבל היא תעשה את זה בצורה שגויה = תקלת אפיון):

יש לפתוח תקלה בדבר תקלת אפיון לטיפולו של האנליסט

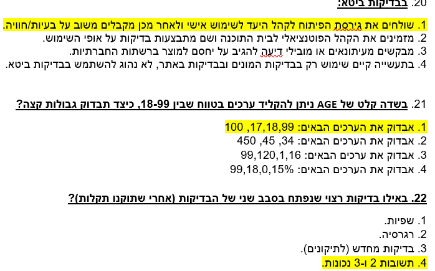
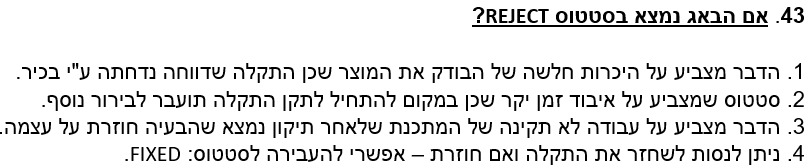
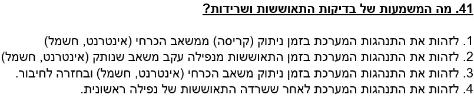
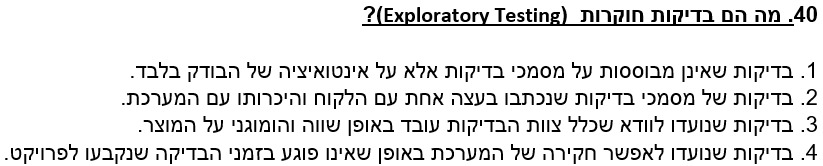
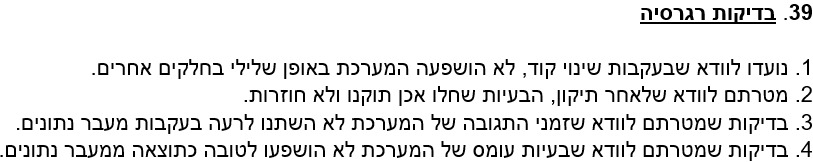
* 1. האם ייתכן מצב של באג ברמת חומרה נמוכה ותיעדוף גבוה?

כן, באג בפונקציה זניחה שהלקוח מתעקש לקבל עליה טיפול חירום לאלתר

* 1. האם ייתכן מצב של באג ברמת חומרה גבוהה עם תיעדוף נמוך?

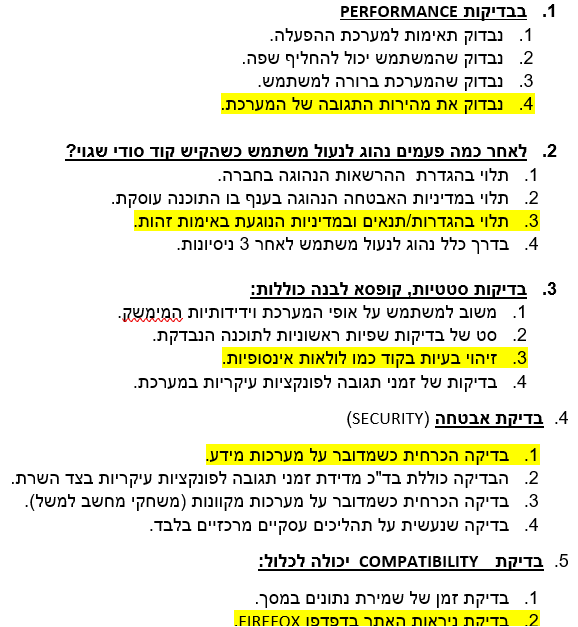
תמונה שמכילה טקסט, צילום מסך, גופן

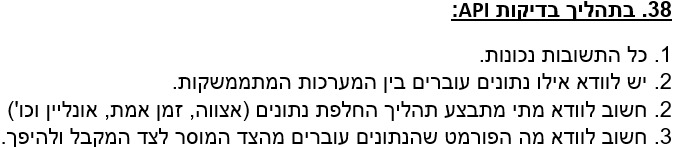
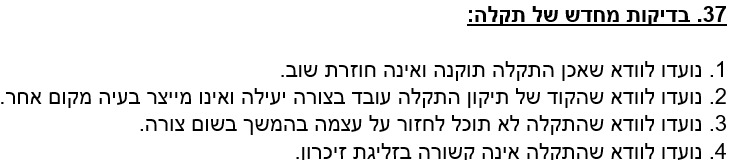
תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.כן, באג שמוביל לקריסה במודול מסוים אך הלקוח אינו משתמש בחלק זה של המערכת

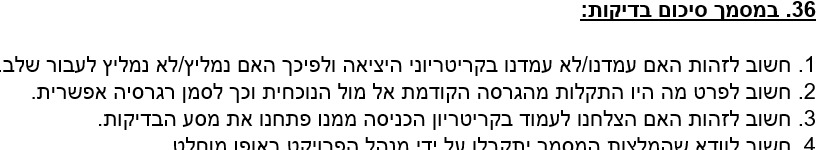
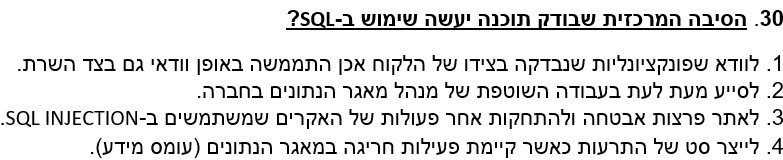
תמונה שמכילה טקסט, צילום מסך, גופן

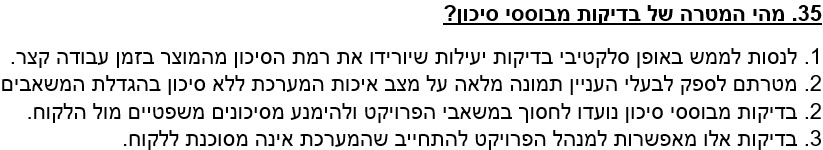
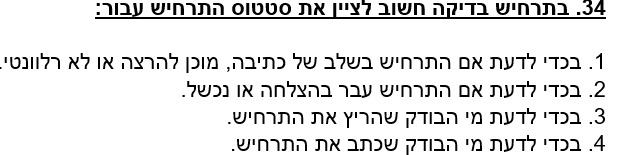
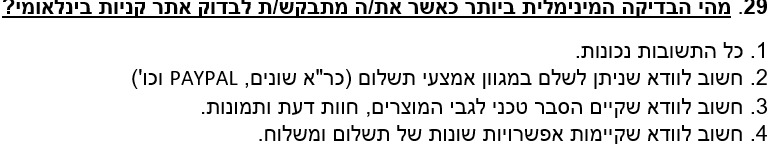
תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.תמונה שמכילה טקסט, צילום מסך, גופן

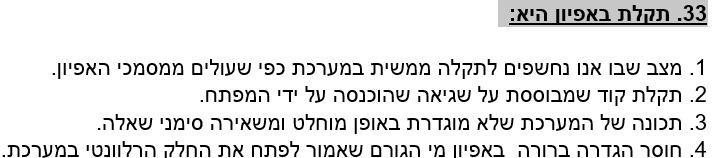
תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.תמונה שמכילה טקסט, גופן, צילום מסך

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

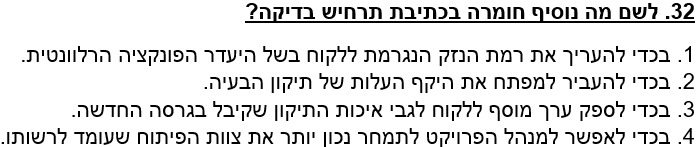
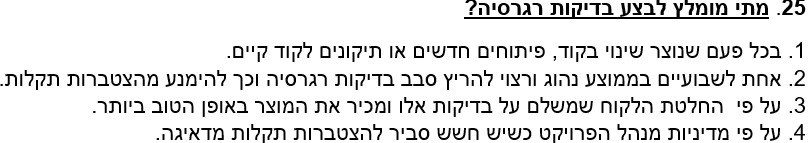
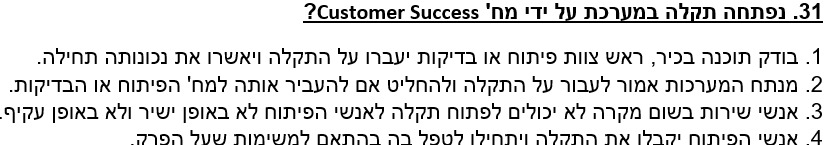
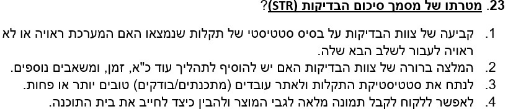
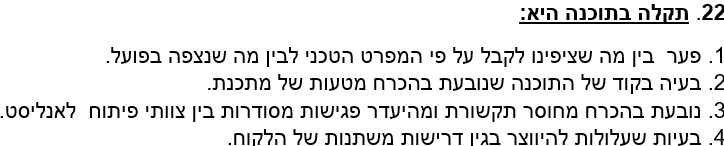
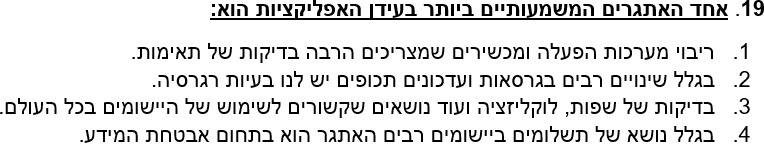
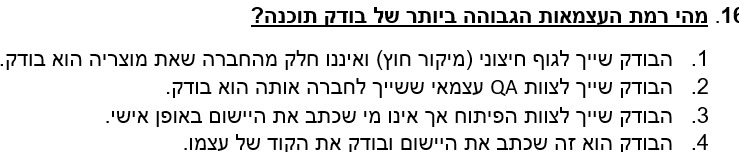
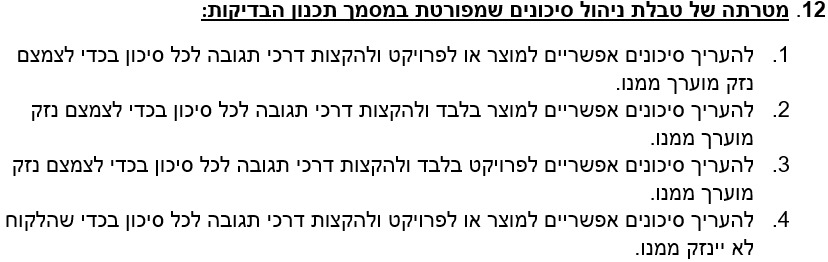
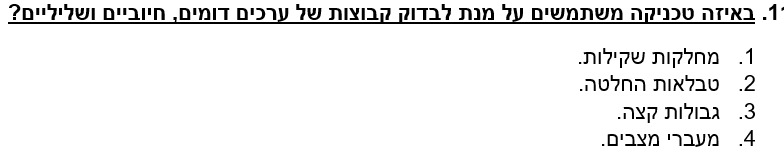
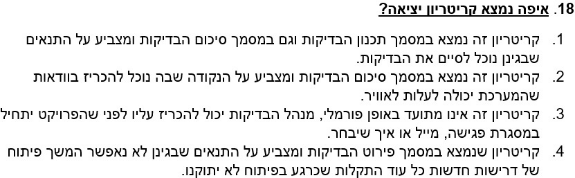
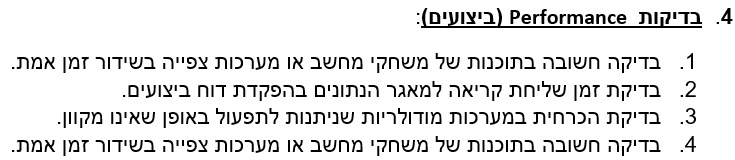
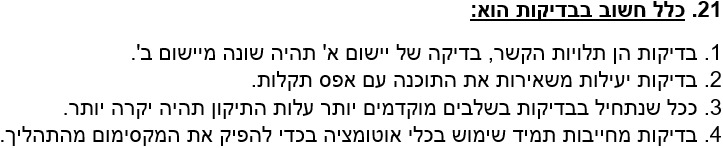
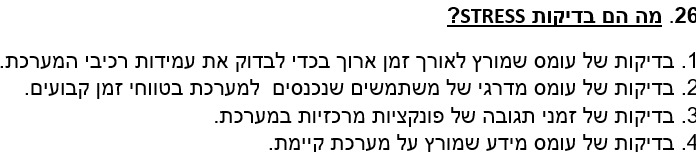


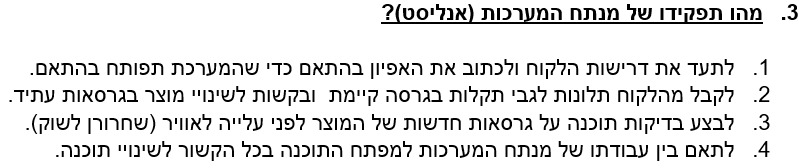
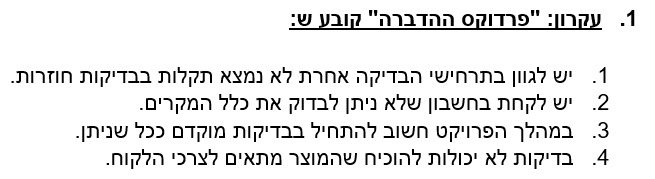
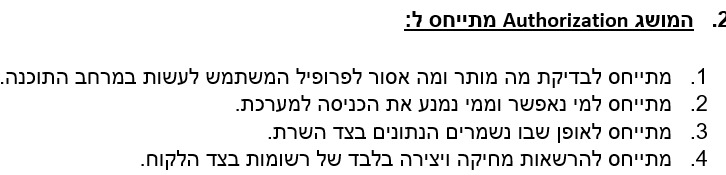






תמונה שמכילה טקסט, גופן, צילום מסך, לבן

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.

תמונה שמכילה טקסט, צילום מסך, גופן, לבן

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.תמונה שמכילה טקסט, גופן, צילום מסך, לבן

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.תמונה שמכילה טקסט, גופן, צילום מסך, לבן

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.תמונה שמכילה טקסט, גופן, צילום מסך, לבן

תוכן שנוצר על-ידי בינה מלאכותית עשוי להיות שגוי.