**פרויקט גמר – סימסטר א':**

**שם הפרויקט: PAWsitive Life**

**שמות אנשי הצוות:**

שם: אור אביכזר אלמליח שם: נועה דנון

ת.ז: 318659455 ת.ז: 207229303

**מטלה 2 – Detailed Requirements:**

**דרישות פונקציונליות:**

* **רישום משתמשים וניהול פרופיל:**
  + משתמשים יכולים להירשם עם שם משתמש וסיסמא.
  + תמיכה ביצירת פרופיל עבור כלב הכולל: שם, גיל, מס' שבב, משקל, פרטי קשר של הוטרינר וכו'.
  + תיעוד מידע רפואי: חיסונים, סירוס/עיקור ומידע רקע.
* **תזכורות וטיפול שוטף:**
  + שליחת תזכורות לקביעת תורים לחיסוני שגרה.
  + תזכורות לרכישת מזון בהתאם להיסטוריית הקניות. (?)
* **מעקב רפואי והתפתחותי:**
  + מעקב אחר שלבי ההתפתחות מגורות ועד בגרות .
  + הצגת מידע רפואי רלוונטי על כל שלב בחיי הכלב.
* **ניהול משאבים ומידע:**
  + הצגת השוואת מחירים עבור ביטוחי כלבים
  + המלצות למזון לכלבים עם חישוב לפי צריכה משוערת ומחירים משוקללים.
  + איתור וטרינרים וגינות כלבים קרובות על מפה אינטראקטיבית.
* **אינטראקציה חברתית:**
  + צ'אט / פורום לבעלי כלבים לשיתוף מידע והתייעצות.
  + ארגון אירועי כלבים מקומיים (?).
  + מעגל "חברים קרובים":
    - המשתמש יוכל לבחור כלבים אחרים במערכת ולהוסיף אותם למעגל "חברים קרובים".
    - אם המשתמש מעדכן שהוא נמצא בגינת כלבים מצוין, המערכת תשלח התראה למיקום זה רק לבעלי הכלבים שנמצאים במעגל החברים הקרובים [ותציע להם להצטרף].
  + קישור בין בעלי כלבים לקביעת פגישות חברתיות.
* **בינה מלאכותית:**
  + שאיבת מידע רפואי והתפתחותי בהתאם לנתוני הכלב.
  + ניתוח מידע רפואי ושיווקי לצורך התאמה אישית למשתמש (?)

**דרישות לא פונקציונליות:**

* **זמינות (Availability):**
  + המערכת תהיה זמינה לפחות **90%** מהזמן במהלך בדיקות או מצגות.
  + זמינות זו מספקת עבור פרויקט אקדמי ואינה דורשת תשתיות מתקדמות.
* **ביצועים (Performance):**
  + זמן תגובה מרבי לפעולות קריטיות (כגון הצגת נתוני פרופיל): **עד 5-8 שניות.**
  + פעולות שאינן קריטיות (כגון עדכון פרטים): **עד 10 שניות**.
* **אבטחת מידע (Security):**
  + אבטחה בסיסית של נתוני המשתמשים תתבצע באמצעות כלי ) Firebaseכגון Authentication).
  + תקשורת בסיסית תשתמש ב-HTTPS אם מתאפשר (אך אינה חובה בשל האופי האקדמי של הפרויקט).
* **סקלאביליות (Scalability):**
  + המערכת תתמוך בעד **50 משתמשים מדומים** בו-זמנית לצורך בדיקות והדגמות.
  + אין צורך בתמיכה באלפי משתמשים בשלב זה.
* **נגישות (Accessibility):**
  + תמיכה בשפה העברית עם יישור מימין לשמאל (RTL).
  + רמות בסיסיות של נגישות, כגון שימוש בטקסט ברור ולחצנים ברורים למשתמשים.
* **אמינות (Reliability):**
  + במקרים של קריסה, המערכת תוכל לשחזר נתונים ידנית במידת הצורך (למשל, על ידי שחזור בסיסי ב-Firebase).
* **חוויית משתמש (User Experience):**
  + ממשק משתמש פשוט ואינטואיטיבי, עם זמני טעינה של עד **10 שניות** למסכים עיקריים.
  + עיצוב בסיסי, רספונסיבי במידת האפשר, המותאם לטלפונים חכמים סטנדרטיים.

**מקרי שימוש:**

* **מקרה שימוש 1 – רישום משתמש חדש:**
  + תנאים מוקדמים: המערכת זמינה (זמינות לפחות 90% מהזמן) וניתן ליצור חשבון חדש.
  + תנאים לאחר: המשתמש רשום במערכת ויכול להוסיף פרטי כלב.
  + זרימה בסיסית:
    - 1 – המשתמש פותח את האפליקציה ובוחר באפשרות "הרשמה".
    - 2 – המשתמש מזין פרטים: שם, סיסמא, אימייל.
    - 3 – המערכת יוצרת חשבון ומציגה עמוד לפרטי הכלב.
  + זרימות חלופיות:
    - אם שם המשתמש תפוס, המערכת תציג הודעה ותאפשר לבחור שם אחר.
* **מקרה שימוש 2 – הוספת כלב לפרופיל המשתמש:**
  + תנאים מוקדמים: המשתמש רשום במער.
  + תנאים לאחר: פרטי הכלב נוספו לפרופיל המשתמש.
  + זרימה בסיסית:
    - 1 – המשתמש נכנס לפרופיל האישי שלו.
    - 2 – המשתמש בוחר באפשרות "הוסף כלב".
    - 3 – המערכת מבקשת מהמשתמש להזין פרטים: שם הכלב, גיל, מספר שבב, משקל, פרטי הווטרינר ועוד.
    - 4 – המשתמש מאשר את הפרטים.
    - 5 – המערכת שומרת את המידע ומעדכנת את פרופיל המשתמש.
  + זרימות חלופיות:
    - אם אחד השדות לא מולא (משדות החובה), המערכת מציגה הודעה ומבקשת השלמה.
* **מקרה שימוש 3 – תזכורת לחיסון:**
  + תנאים מוקדמים: פרטי הכלב, כולל תאריכי חיסונים קודמים, הוזנו למערכת.
  + תנאים לאחר: המשתמש מקבל התראה על חיסון קרוב.
  + זרימה בסיסית:
    - 1 – המשתמש מזין תאריך חיסון קודם.
    - 2 – המערכת מחשבת את תאריך החיסון הבא ושולחת תזכורת.
  + זרימות חלופיות:
    - המשתמש דוחה תזכורת, והמערכת מעדכנת את התאריך בהתאם.
* **מקרה שימוש 4 – מעגל חברים קרובים:**
  + תנאים מוקדמים: המשתמש רשום במערכת ויש לו פרופיל עם כלב.
  + תנאים לאחר: כלב נוסף למעגל "החברים הקרובים".
  + זרימה בסיסית:
    - 1 – המשתמש נכנס לאזור "חברים קרובים".
    - 2 – המשתמש מחפש כלב לפי שם הכלב, שם הבעלים או פרטים נוספים.
    - 3 – המשתמש בוחר להוסיף את הכלב מעגל החברים הקרובים.
    - 4 – המערכת מציגה הודעה המאשרת שהכלב נוסף.
  + זרימות חלופיות:
    - אם הכלב כבר במעגל, המערכת מציגה הודעה מתאימה.
* **מקרה שימוש 5 – איתור וטרינר או גינת כלבים בקרבת מקום:**
  + תנאים מוקדמים: המשתמש נתן הרשאה למיקום.
  + תנאים לאחר: המשתמש רואה מפת מיקום עם תוצאות רלוונטיות.
  + זרימה בסיסית:
    - 1 – המשתמש נכנס לאזור "מפה".
    - 2 – המשתמש בוחר באפשרות "וטרינר" או "גינת כלבים".
    - 3 – המערכת מזהה את מיקום המשתמש ומציגה תוצאות רלוונטיות.
    - 4 – המשתמש בוחר נקודה במפה ומקבל פרטים נוספים, כגון כתובת ומספר טלפון.
  + זרימות חלופיות:
    - אם אין תוצאות באזור, המערכת מציגה הודעה מתאימה.
* **מקרה שימוש 6 – הצטרפות לקבוצת רכישה:**
  + תנאים מוקדמים: המשתמש רשום במערכת.
  + תנאים לאחר: המשתמש הצטרף לקבוצת רכישה.
  + זרימה בסיסית:
    - 1 – המשתמש נכנס לאזור "קבוצות רכישה".
    - 2 – המשתמש בוחר קבוצת רכישה לפי קטגוריה (מזון, חטיפים, ציוד וכו').
    - 3 – המשתמש מצטרף לקבוצה ומזין את הפרטים הדרושים (כמו כמות פריטים רצויה).
    - 4 – המערכת מעדכנת את המשתמש עם פרטי התשלום והאיסוף.
  + זרימות חלופיות:
    - אם אין קבוצות פעילות, המערכת מציעה למשתמש ליצור קבוצה חדשה.
* **מקרה שימוש 7 – שליחת התראה על מיקום בגינת כלבים:**
  + תנאים מוקדמים: המשתמש הגדיר מעגל חברים קרובים.
  + תנאים לאחר: נשלחה התראה לכלבים במעגל החברים הקרובים.
  + זרימה בסיסית:
    - 1 – המשתמש מעדכן את מיקומו בגינת כלבים.
    - 2 – המערכת מזהה את המיקום ומציגה אפשרות לשלוח התראה.
    - 3 – המשתמש מאשר את השליחה.
    - 4 – המערכת שולחת הודעה לכל החברים הקרובים עם שם הגינה והמיקום שלה.
  + זרימות חלופיות:
    - אם אין חברים קרובים, המערכת מציגה הודעה מתאימה.
* **מקרה שימוש 8 – מעקב אחר שלבי התפתחות:**
  + תנאים מוקדמים: המשתמש הוסיף פרטי כלב לפרופיל.
  + תנאים לאחר: המשתמש מקבל מידע על שלב ההתפתחות הנוכחי של הכלב.
  + זרימה בסיסית:
    - 1 – המשתמש מזין נתונים עדכניים על הכלב, כגון גיל ומשקל.
    - 2 – המערכת משתמשת באלגוריתם בינה מלאכותית לניתוח שלב ההתפתחות (גור, צעיר, בוגר וכו') ושאיבת מידע רלוונטי.
    - 3 – המערכת מציגה מידע רלוונטי, כגון אילוף מומלץ ותהליכים פיזיולוגיים.
  + זרימות חלופיות:
    - אם נתונים חסרים, המערכת מבקשת השלמה מהמשתמש

**סיפורי משתמשים:**

* **סיפור משתמש 1: רישום משתמש חדש:**
  + כתיאור המשתמש: אני אדם שרק עכשיו אימץ כלב ראשון ואין לי הרבה ידע על גידול כלבים.
  + מטרה: אני רוצה להירשם לאפליקציה כדי לקבל תזכורות לחיסונים, מידע על סוגי מזון וכלים להתמודדות עם קשיים כמו אילוף.
  + קריטריוני קבלה:
    - המשתמש מצליח להירשם באמצעות שם משתמש וסיסמא.
    - לאחר ההרשמה, המשתמש יכול להוסיף את פרטי הכלב.
    - המערכת שולחת תזכורות לטיפולים הקרובים ומספקת מידע על גידול כלבים.
* **סיפור משתמש 2: הוספת כלב לפרופיל:**
  + כתיאור משתמש: אני בעלים של שני כלבים ורוצה לעקוב אחר המידע הרפואי והחיסונים שלהם.
  + מטרה: אני רוצה להוסיף את הכלב החדש שלי לפרופיל כדי לנהל את הנתונים שלו באפליקציה.
  + קריטריוני קבלה:
    - המשתמש יכול להזין פרטים כמו שם הכלב, גיל, משקל ומספר שבב.
    - המערכת מאפשרת לנהל את הנתונים הרפואיים של כל כלב בנפרד.
    - המשתמש רואה תזכורות מותאמות לכל כלב בנפרד.
* **סיפור משתמש 3: תזכורת לחיסונים:**
  + כתיאור משתמש: אני בעלים עסוק ולעיתים קרובות שוכח לקבוע תורים לחיסונים השנתיים של הכלב שלי.
  + מטרה: אני רוצה שהאפליקציה תזכיר לי מתי לקבוע תור לחיסון הבא.
  + קריטריוני קבלה:
    - המשתמש מקבל התראה לפחות שבוע לפני התאריך המיועד לחיסון.
    - המערכת מאפשרת למשתמש לדחות את התזכורת ולחשב מחדש את התאריך.
    - התזכורת כוללת פרטים כמו סוג החיסון ותאריך מומלץ לקביעת התור.
* **סיפור משתמש 4: קישור לגינת כלבים:**
  + כתיאור משתמש: אני בעלים חברתי שאוהב להיפגש עם חברים בגינת הכלבים.
  + מטרה: אני רוצה לעדכן את החברים הקרובים שלי כשאני מגיע לגינת כלבים כדי שנוכל להיפגש.
  + קריטריוני קבלה:
    - המשתמש יכול להוסיף כלבים למעגל החברים הקרובים.
    - המערכת מאפשרת לעדכן מיקום בגינת כלבים ולשלוח התראה אוטומטית לחברים.
    - ההתראה כוללת את שם הגינה והמיקום שלה.
* **סיפור משתמש 5: השתתפות בקבוצת רכישה:**
  + כתיאור משתמש: אני רוצה לחסוך כסף ברכישת מזון וציוד לכלבים.
  + מטרה: אני רוצה להצטרף לקבוצת רכישה כדי לרכוש מזון במחיר מוזל.
  + קריטריוני קבלה:
    - המשתמש בוחר קטגוריה (מזון, חטיפים וכו') ומצטרף לקבוצת רכישה פעילה.
    - המערכת מעדכנת את המשתמש בפרטי התשלום והאיסוף.
    - המשתמש מקבל הודעה כשהקבוצה מגיעה למינימום משתתפים.
* **סיפור משתמש 6: מעקב אחר שלבי התפתחות:**
  + כתיאור משתמש: אני מגדל גור כלבים ורוצה לדעת מה מצפה לי בשלב הבא בגידול שלו.
  + מטרה: אני רוצה לקבל מידע על שלב ההתפתחות הנוכחי של הכלב, כולל טיפים לאילוף וטיפול.
  + קריטריוני קבלה:
    - המשתמש מזין נתונים כמו גיל ומשקל הכלב.
    - המערכת משתמשת בבינה מלאכותית ומציגה את שלב ההתפתחות הנוכחי ואת המידע הרלוונטי.
    - המשתמש מקבל טיפים מותאמים אישית כמו אילוף, התנהגות ותזונה.
* **סיפור משתמש 7: צ'אט להתייעצות:**
  + כתיאור משתמש: אני רוצה להתייעץ עם בעלי כלבים אחרים בנוגע לבעיות אילוף.
  + מטרה: אני רוצה לשאול שאלות בצ'אט או בפורום ולהשתתף בדיונים עם משתמשים אחרים.
  + קריטריוני קבלה:
    - המשתמש יכול להעלות שאלה לצ'אט.
    - משתמשים אחרים יכולים להגיב ולהציע פתרונות.
    - המערכת מאפשרת להוסיף קבצים כמו תמונות או סרטונים להמחשת הבעיה.
* **סיפור משתמש 8: איתור וטרינרים וגינות כלבים:**
  + כתיאור משתמש: אני עברתי לאזור חדש ולא מכיר את הווטרינרים וגינות הכלבים הקרובים.
  + מטרה: אני רוצה למצוא וטרינרים וגינות כלבים באזור שלי.
  + קריטריוני קבלה:
    - המשתמש מאפשר לאפליקציה לגשת למיקום שלו.
    - המערכת מציגה מפה אינטראקטיבית עם מיקומי וטרינרים וגינות כלבים.
    - המפה כוללת פרטים כמו שעות פתיחה ופרטי קשר.
* **סיפור משתמש 9: שיפור החוויה היומיומית:**
  + כתיאור משתמש: אני רוצה לוודא שאני מנהל את חיי הכלב שלי בצורה יעילה ובריאה.
  + מטרה: אני רוצה שהאפליקציה תעזור לי להישאר מעודכן בכל מה שקשור לכלב שלי, מטיפול רפואי ועד מפגשים חברתיים.
  + קריטריוני קבלה:
    - המערכת שולחת תזכורות חשובות באופן קבוע.
    - המשתמש יכול לנהל את כל הפרטים של הכלב שלו במקום אחד.
    - המערכת מספקת מידע מועיל ותמיכה בהתנהלות היומיומית.

**תיעדוף והערכת המאמץ עבור כל דרישה:**

|  |  |  |  |
| --- | --- | --- | --- |
| **פונקציונליות** | **תיעדוף** | **הערכת מאמץ (שעות)** | **הסבר** |
| רישום משתמשים וניהול פרופיל | Must Have | 10 | בסיס חיוני למערכת – רישום משתמשים וניהול פרטי הכלב. |
| תזכורות לטיפולים וחיסונים | Must Have | 12 | חיוני למעקב רפואי ותפעול אפליקציה. |
| הצגת מידע רפואי לכלב באמצעות בינה מלאכותית | Should Have | 20 | מוסיף ערך אך אפשר לדחות לאחר השלמת הפונקציות הקריטיות. |
| מעגל חברים קרובים | Should Have | 10 | פונקציה חברתית שמוסיפה ערך אך אינה קריטית לפעילות הבסיסית |
| מפת גינות כלבים ווטרינרים | Should Have | 15 | פונקציה מועילה למשתמש, אך דורשת יותר משאבים לפיתוח ואינטגרציה. |
| קבוצות רכישה | Nice to Have | 18 | פונקציה מועילה חברתית, אך דורשת עיצוב ותכנון נוסף. |
| צ'אט בין בעלי כלבים | Nice to Have | 25 | פונקציה מורכבת ולא הכרחית, במיוחד בפרויקט קטן. |

**תכנון לפי תיעדוף:**

* שלב שלישי – פונקציות מורכבות יותר:
  + מפת גינות כלבים ווטרינרים.
  + קבוצות רכישה.
* שלב רביעי (אחרון) – פונקציות מתקדמות:
  + צ'אט.
* שלב ראשון – פונקציות קריטיות:
  + רישום משתמשים וניהול פרופיל.
  + תזכורות לטיפולים וחיסונים.
* שלב שני – פונקציות מועילות:
  + הצגת מידע רפואי לכלב.
  + מעגל חברים קרובים.

**דרישות אדריכליות:**

1. ארכיטקטורת מערכת:

* המערכת תשתמש בארכיטקטורה מבוססת **MVC**, המתאימה לפיתוח מהיר וניהול קוד ברור בפרויקטים קטנים.
* Firebase ישמש כמסד נתונים ראשי לניהול נתונים ולסנכרון בזמן אמת.

2. סקלאביליות:

* תמיכה בעד **50 משתמשים מדומים בו-זמנית** לצורך בדיקות והדגמות.
* Firebase תספק יכולת בסיסית לניהול נתונים.

3. ביצועים:

* המערכת תספק זמן תגובה של עד 5-8 **שניות לפעולות קריטיות**, בהתחשב במגבלות משאבים ותשתיות בפרויקט קטן.
* תמיכה בכמות משתמשים מינימלית, **עד 10 משתמשים פעילים בו-זמנית** בהדגמות.

4. אמינות וזמינות:

* יעד זמינות של **90% במהלך בדיקות ומצגות**.
* מנגנוני גיבוי ידניים באמצעות Firebase.

5. אבטחה:

* שימוש ב-**Firebase Authentication** לאימות בסיסי.
* שימוש ב-**HTTPS** יתבצע אם מתאפשר, אך אינו דרישה מחייבת בשל האופי האקדמי של הפרויקט. (?)

6. ארכיטקטורת נתונים:

* שימוש ב-**Firebase Realtime Database** לניהול נתונים.
* הנתונים יותאמו למבנה פשוט של פרופילי משתמשים ופרטי כלבים בלבד. (?)

7. אינטגרציה:

* המערכת תשתלב ברמה **בסיסית** עם **Google Maps API** לצורך הצגת מיקומי וטרינרים וגינות כלבים.
* אינטגרציה עם שירותי **AI** כמו OpenAI תהיה סימבולית בלבד, ללא יישום מורכב. (?)

8. חוויית משתמש:

* תמיכה **בעיצוב בסיסי ורספונסיבי** באפליקציה, מותאם לטלפונים חכמים נפוצים.
* תצוגה אינטואיטיבית של פרופילים, תזכורות ומידע נוסף.

9. מעקב ולוגים:

* שימוש ב-**Firebase Crashlytics** או כלי דומה למעקב בסיסי אחרי שגיאות.
* אין צורך במעקב מתקדם אחר ביצועים או ניטור בזמן אמת. (?)

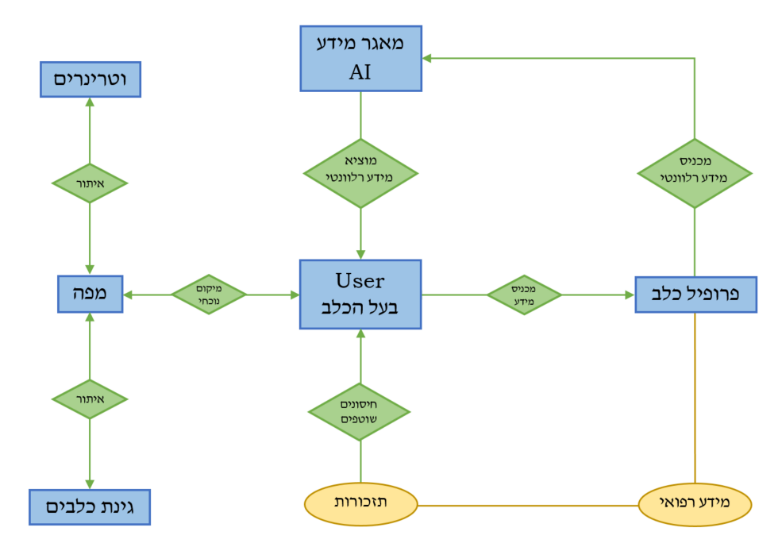
10. תאימות:

* המערכת תהיה מותאמת למכשירי **אנדרואיד בגרסה 8.0 ומעלה**, התואמת לרוב המכשירים.
* נגישות בסיסית, עם טקסט ברור ולחצנים נוחים לשימוש.

11. טכנולוגיות:

* פיתוח האפליקציה יתבצע ב-**Java/Kotlin** עם Android Studio.
* Firebase תספק את כל שירותי ה-Backend, כולל אחסון ואימות משתמשים, ללא שימוש בשירותי ענן מורכבים נוספים.

**דיאגרמת USE-CASE:**



**דרישות טכנולוגיות:**

*1. שפות תכנות ומסגרות עבודה:*

**Frontend ו-Backend:**

* שפת פיתוח: **Java/Kotlin** לפיתוח אפליקציית אנדרואיד.
* מסגרות עבודה נבחרות: **Android SDK** ו-**Firebase SDK**.

**שיקולים:**

* Java/Kotlin מתאימות במיוחד לפיתוח אפליקציות אנדרואיד, בעלות תמיכה נרחבת בקהילה ותיעוד מצוין.

*2. מסדי נתונים:*

**Firebase Realtime Database:**

* שמירת נתונים בזמן אמת, כמו פרופילי משתמשים ופרטי כלבים.
* אין דרישה ליכולות מתקדמות של **Offline Sync**, אך נתונים יישמרו עם אפשרות לגיבוי ידני.

**דרישות:**

* יכולת אחסון של נתונים מובנים, כמו פרופילי משתמשים, פרטי כלבים ותזכורות.
* **Auto-Scaling** לא נדרש, אך Firebase יתמוך בעומסים קטנים לצורך בדיקות. (?)

*3. טכנולוגיות Frontend:*

**תשתית פיתוח:**

* שימוש ב-**XML** לעיצוב ממשק המשתמש.
* תמיכה בעיצוב רספונסיבי וממשק ידידותי למשתמש.

**שיקולים:**

* תצוגה נוחה וברורה של פרופילים, תזכורות ומאמרים מותאמים אישית.

*4. טכנולוגיות Backend:*

**Firebase Functions (לא חובה):**

* אם נדרש עיבוד לוגיקה עסקית, ניתן לשלב ברמה בסיסית.

**API חיצוניים:**

* שימוש ב-**Google Maps API** להצגת מיקומים, ללא אינטגרציה עמוקה.
* שילוב **סימבולי** עם שירותי **AI**, כמו OpenAI, אם הזמן מאפשר.

*5. שירותי ענן:*

**Firebase:**

* שירותי Backend הכוללים אחסון, אימות משתמשים, והתראות.
* מתאים במיוחד לפרויקטים התחלתיים, עם אפשרות להתרחבות בעתיד.

**Google Cloud:**

* לניהול שירותי Firebase ולשילוב עם Maps API.

*6. כלי פיתוח:*

**סביבת פיתוח:**

* **Android Studio:** סביבת פיתוח מובילה לפיתוח אפליקציות אנדרואיד.

**ניהול גרסאות:**

* שימוש ב-**Git** עם **GitHub** לניהול גרסאות ושיתוף פעולה בצוות.

**בדיקות:**

* **JUnit** לבדיקות יחידה (Unit Tests).
* **Firebase Test Lab** לבדיקות מכשירים (Device Testing).

*7. כלים לאבטחת מידע:*

**פרוטוקולים:**

* **HTTPS** להצפנת התקשורת בין האפליקציה לשרתים.

**אימות:**

* **Firebase Authentication** יספק אבטחת גישה בסיסית.

**הגנה על מסדי נתונים:**

* חוקי אבטחה ב-**Firebase** (Security Rules) למניעת גישה לא מורשית.

**ניתוח חולשות:**

* שילוב כלים כמו Snyk או OWASP ZAP לאיתור חולשות בקוד.

*8. תאימות וסטנדרטים:*

**תקנים:**

* דגש על נגישות בסיסית (טקסט ברור ולחצנים נגישים).
* אין דרישה לעמידה מלאה בתקני GDPR, אך המערכת תשמור על פרטיות בסיסית.

**תמיכה במכשירים:**

* תאימות למכשירי אנדרואיד בגרסה 8.0 ומעלה.

*9. סטנדרטים לתיעוד:*

**קוד:**

* תיעוד ברור בקוד (Comments) תוך שימוש בתבניות אחידות.

**API:**

* אין צורך בתיעוד מתקדם כמו Swagger; תיעוד ידני במידת הצורך. (?)

**ארכיטקטורה:**

* יצירת דיאגרמות UML לתיאור המערכת והרכיבים.

*10. בדיקות ואבטחת איכות:*

**מסגרת בדיקות:**

* JUnit לבדיקות אוטומטיות של לוגיקה עסקית.

**כלי ביצועים:**

* אין צורך ב-Firebase Performance Monitoring; תהליכי בדיקה ידניים יספיקו. (?)

**שיטות עבודה:**

* דגש על בדיקות ידניות במהלך פיתוח, תוך התמקדות בפונקציות הקריטיות.