מסמך אפיון Warranty Keeper -

1. מבוא

Warranty Keeper היא מערכת לניהול אחריות על מוצרים שנרכשו. המערכת מאפשרת למשתמש להעלות מסמך תעודת אחריות, ומערכת AI תנתח את המסמך ותשלוף את הנתונים הרלוונטיים ללא צורך בהזנה ידנית. המשתמש יוכל לעדכן נתונים אלו לפי הצורך.
המערכת כוללת גם תזכורות לפני פקיעת האחריות, הצגת מצב האחריות של המוצרים, חיפוש וסינון מוצרים, תמיכה בריבוי משתמשים ושיתוף אחריות, וכן הצגת המלצות לתיקון או רכישה במקרה של תקלה לאחר פקיעת האחריות.

1.1 מטרות הפרויקט

- לאפשר למשתמשים לנהל אחריות על מוצרים בקלות וביעילות.
- לספק התראות על תוקף האחריות למניעת החמצת הזדמנויות לתיקון במסגרת האחריות.
- לאפשר חיפוש וסינון מוצרים בקלות לפי תוקף האחריות, קטגוריות וכו.'
- לתמוך בשיתוף אחריות בין מספר משתמשים במקרה של מוצרים משותפים.
- להציע המלצות לתיקון או רכישה במידה והאחריות פגה.

1.2 קהל היעד

- אנשים פרטיים הרוכשים מוצרים עם אחריות ורוצים לנהל אותם בצורה מסודרת.
- משפחות או קבוצות שותפים המעוניינים לשתף אחריות על מוצרים.
- עסקים קטנים שרוצים לוודא שמוצריהם מכוסים באחריות תקפה.

אסטרטגיה עסקית

שיתופי פעולה עם חברות מוצרי חשמל

מנהל האפליקציה יוצר מגעים עסקיים עם **חברות מוצרי חשמל גדולות** ,מכיוון שתעודות אחריות הן רלוונטיות במיוחד עבור תחום זה.

- כאשר משתמש מעלה **תעודת אחריות** של חברה שנמצאת בשיתוף פעולה עם המערכת, יוצג **כפתור להארכת האחריות** ולכניסה ישירה לאתר החברה.
- כל כניסה לאתר החברה דרך האפליקציה תתועד, ו**פעם בחודש** יבוצע חישוב של כמות הכניסות דרך האפליקציה.
- על בסיס כמות הכניסות, החברה תשלם **עמלה חודשית** למערכת.Warranty Keeper

שליחת דוח שימוש חודשי לחברות

פעם בחודש תישלח **לחברות השותפות דוח שימוש מפורט** ,הכולל:

- כמות המשתמשים שהעלו תעודות אחריות של החברה.
- מספר הכניסות לאתר החברה דרך המערכת.
- סטטיסטיקות כלליות על מוצרים שהמשתמשים מנהלים תחת האחריות של החברה.

1.3 פונקציונליות עיקרית

- **העלאת מסמך תעודת אחריות** –המערכת תזהה ותשלוף את הנתונים הרלוונטיים באמצעות AI.
- **אפשרות עריכת הנתונים** שהופקו על ידי המשתמש.
- **יצירת רשימה מרכזית** של כל המוצרים והצגת סטטוס האחריות שלהם.
- **שליחת התראות** לפני פקיעת האחריות.
- **חיפוש וסינון מוצרים** לפי קטגוריה, חנות רכישה, ותוקף אחריות.
- **תמיכה בריבוי משתמשים** ושיתוף אחריות על מוצרים.
- **הצגת המלצות לרכישה או תיקון** במקרים בהם האחריות כבר פגה.

1.4 בעיות שהמערכת פותרת

אובדן מסמכים –משתמשים רבים שומרים תעודות אחריות בצורה פיזית או מפוזרות בין מיילים ותיקיות, מה שמוביל לאובדן המסמך בעת הצורך.
קושי באיתור מידע רלוונטי –תעודות אחריות מכילות מידע רב, ולא תמיד ברור מהי תקופת האחריות, אילו חלקים מכוסים, או כיצד לממש את האחריות.
חוסר נוחות בניהול תעודות אחריות מרובות –מוצרים שונים נרכשים מתקופות שונות, וניהול מסודר של כל תעודות האחריות בצורה נגישה הוא אתגר.

1.5 טכנולוגיות בשימוש

- **צד שרת ASP.NET Core**: לניהול ה API- והלוגיקה העסקית.
- **צד לקוח React.js**: לבניית ממשק משתמש דינמי ואינטראקטיבי.
- **מסד נתונים MySQL**: לניהול המידע על המוצרים והאחריות.
- **זיהוי טקסט ממסמכים**: שימוש ב OCR-ו) NLP-למשל (Tesseract + OpenAI API לשליפת נתונים ממסמכים.
- **מערכת התראות**: שילוב של Firebase Cloud Messaging או Twilio לשליחת תזכורות והתראות.

2. מצורף בנפרד

3.אפיון תמציתי

3.1רשימת פונקציות המערכת

לוגיקה עסקית	פלט	פרמטרים	Route	שם הפונקציה
מזהה אוטומטית את הנתונים הרלוונטיים מתעודת האחריות	{ fileId: string, extractedData: object }	Body: { file: File }	POST /api/upload	**העלאת מסמך**
מציג את כל המסמכים ששייכים למשתמש	{ documents: Document[] }	Headers: { Authorization: Bearer token }	GET /api/documents	**קבלת רשימת המסמכים**

לוגיקה עסקית	פלט	פרמטרים	Route	שם הפונקציה
מאפשר למשתמש לעדכן נתונים במקרה של זיהוי שגוי	{ success: true }	Body: { updatedData: object }	PUT /api/document/{id}	עדכון נתוני מסמך
מוחק מסמך קיים מהמערכת	{ success: true }	Headers: { Authorization: Bearer token }	DELETE /api/document/{id}	מחיקת מסמך
בודק מסמכים שהתוקף שלהם עומד לפוג ושולח התראה למשתמש	{ notifications: Notification[] }	Headers: { Authorization: Bearer token }	GET /api/notifications	התראות על סיום אחריות

3.2אימות והרשאות

– **Authentication** מבוצע באמצעות.JWT Authentication
תפקידים במערכת:
משתמש רגיל –יכול להעלות, לערוך ולמחוק מסמכים משלו.
מנהל מערכת –יכול לנהל משתמשים, לעדכן הגדרות ולצפות בסטטיסטיקות שימוש.

3.3אפליקציית ניהול

הפקת דוחות שימוש (כמות מסמכים שהועלו, תאריכי תפוגה קרובים וכו').

- יצירת משתמשים, מחיקתם ועדכון הרשאות.
- ניהול מכסות אחסון לכל משתמש, עדכון הגדרות אבטחה וניהול הרשאות גישה.

4. מבנה בסיס הנתונים(DB Schema)

טבלת משתמשים:(Users)

- **id (PK)** – מזהה משתמש
- **name** – שם מלא
- **email** – כתובת אימייל
- **password** – סיסמה (מוצפנת)

טבלת מוצרים:(Products)

- **id (PK)** – מזהה מוצר
- **user_id (FK)** – מזהה משתמש הבעלים

- name – שם המוצר
- category – קטגוריה
- purchase_date – תאריך רכישה
- warranty_expiration – תאריך פקיעת אחריות

טבלת מסמכי אחריות:(WarrantyDocuments)

- id (PK) – מזהה מסמך
- product_id (FK) – מזהה מוצר
- file_url – כתובת המסמך

טבלת שותפויות עסקיות(BusinessPartners)

- id (PK) – מזהה חברה
- name – שם החברה
- website_url – כתובת האתר של החברה
- agreement_details – פרטי ההסכם

טבלת כניסות לאתרי שותפים(PartnerWebsiteVisits)

- id (PK) – מזהה כניסה
- user_id (FK) – מזהה משתמש
- company_id (FK) – מזהה החברה
- visit_date – תאריך הכניסה

5. לוח זמנים משוער לפיתוח

- **ספרינט 1 (שבוע 1)**: הקמת בסיס נתונים, יצירת API ראשוני וניהול משתמשים.
- **ספרינט 2 (שבוע 2)**: העלאת מסמכים וזיהוי נתונים באמצעות AI.
- **ספרינט 3 (שבוע 3)**: בניית ממשק משתמש והצגת רשימת המוצרים.
- **ספרינט 4 (שבוע 4)**: פיתוח מערכת התראות ותמיכה בריבוי משתמשים.
- **ספרינט 5 (שבוע 5)**: פיתוח מערכת המלצות לרכישה ותיקון.
- **ספרינט 6 (שבוע 6)**: בדיקות, שיפורים והעלאת המערכת לאוויר.