

תרגיל פרויקט – תיבת תלונות אנונימיות בבסיס צבאי (React + Express API)

רקע

צה"ל בוחן מערכת שמאפשרת לחיילים לשלוח תלונות אנונימיות באופן דיגיטלי. המערכת לא שומרת שם/מזהה משתמש, רק:

- `category` (אוכל / ציוד / פקודות / אחר)
- `message` (טקסט התלונה)
- `createdAt` (תאריך יצירה)

רק מפקדים שמזינים **סיסמת אדמין** יכולים לצפות ברשימת התלונות.

מטרות לימוד

צד לקוח (React)

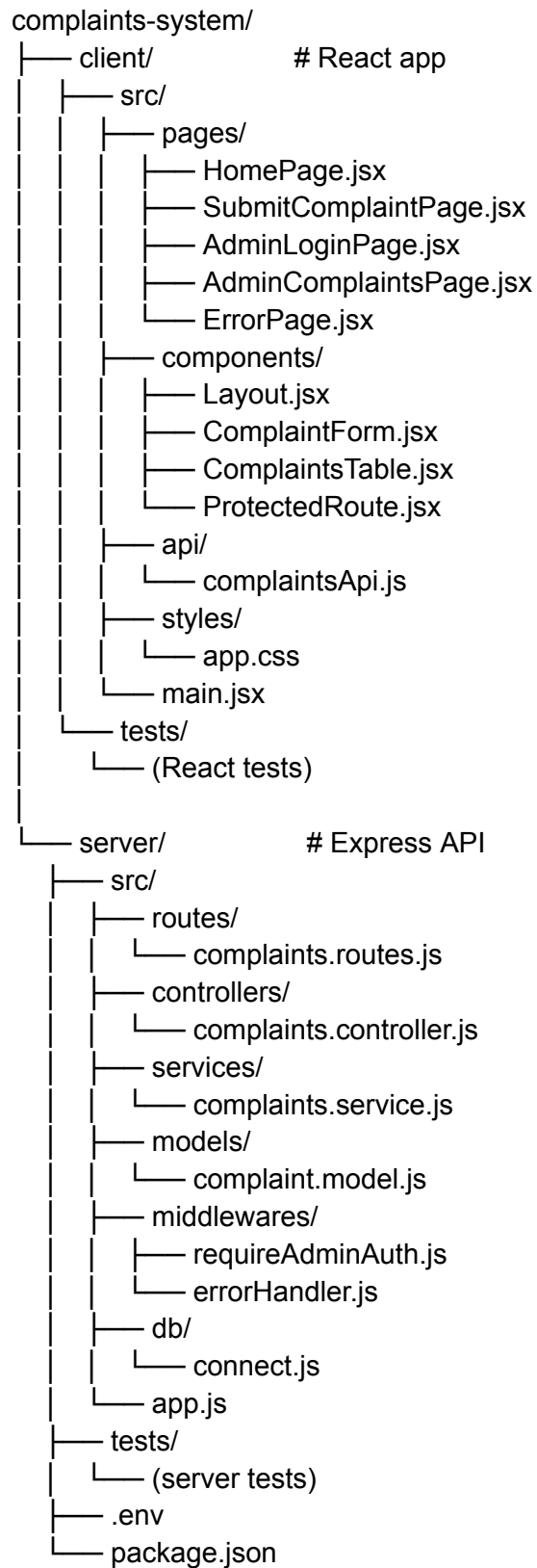
- עבודה עם **React Router** (ניווט בין דפים)
- עבודה עם **טפסים** (controlled components)
- שליחת בקשות לשרת בעזרת `fetch` / `axios`
- ניהול מצב בסיסי (`useState`, `useEffect`)
- בניית UI נקי ורספונסיבי

צד שרת (Express API)

- בניית **REST API** במקום הגשת HTML סטטי
 - שמירה למסד נתונים בענן (Supabase או MongoDB Atlas)
 - הפרדה לארכיטקטורה: `routes` / `controllers` / `services` / `models`
 - בדיקות Unit + Integration לשרת
-

ארכיטקטורה ותיקיות מומלצות

מונוריפו (מומלץ)



דרישות פונקציונליות – צד לקוח (React)

(/) Home Page (1)

מציג:

- הסבר קצר על מטרת המערכת
- כפתור/לינק: "שליחת תלונה" → מעבר ל-`submit/`
- אזור "מפקדים בלבד":
 - שדה סיסמה
 - כפתור "כניסה לאדמין" → מעבר ל-`admin/login/` או ביצוע התחברות ואז ניתוב ל-`admin/`

דגש: ב-React אין "טעינת CSS דרך Express" כמו בפרויקט הישן; ה-UI מוגש ע"י React. השרת עכשיו אחראי ל-API בלבד.

(/submit Submit Complaint Page (2)

כותרת: "שליחת תלונה אנונימית"
טופס (Controlled):

- `select` קטגוריה: אוכל / ציוד / פקודות / אחר
- `textarea` תוכן תלונה (מינימום תווים מומלץ)
- כפתור שליחה
- אחרי הצלחה:
 - הודעת הצלחה
 - איפוס טופס / ניתוב חזרה לדף הבית

(/admin/login Admin Login Page (3)

- שדה סיסמה
- כפתור "התחברות"
- אם הסיסמה שגויה → הודעת שגיאה + אין גישה לדף התלונות

(/admin Admin Complaints Page (4)

- מוצג רק אם המשתמש עבר אימות אדמין
- מציג רשימת תלונות בטבלה/כרטיסים:
 - `category`
 - `message`
 - `createdAt` (פורמט קריא)
- בונוס: פילטר לפי `category` + מונה לכל קטגוריה

(5) הגנה על נתיבים (Client)

- `ProtectedRoute` שמונע גישה ל-`admin/` אם אין אימות

- אם אין אימות → ניתוב ל-`admin/login/`

חשוב: הגנה בצד לקוח בלבד לא מספיקה. חייבים גם הגנה בצד שרת (ראו בהמשך).

דרישות פונקציונליות – צד שרת (Express API)

API Endpoints (מינימום נדרש)

1. `POST /api/complaints`
 - קולט: `{ category, message }`
 - שומר במסד: `{ category, message, createdAt }`
 - מחזיר תשובה של הצלחה (ואפשר גם את האובייקט שנשמר)
2. `POST /api/admin/login`
 - קולט: `{ password }`
 - משווה ל-`ADMIN_PASSWORD` מתוך `env`.
 - אם תקין:
 - מחזיר "אימות הצליח"
 - ונותן מנגנון אימות (אחד מהבאים):
 - **Option A (פשוט):** טוקן קצר חיים (JWT) שמוחזר ללקוח
 - **Option B:** Session cookie (אם עובדים עם cookies)
- אם לא תקין: `401 Unauthorized`
3. `GET /api/complaints` (מוגן)
 - דורש אימות אדמין (middleware)
 - מחזיר רשימת תלונות (ממוינת מהחדש לישן)

כללי אבטחה בסיסיים

- הסיסמה נשמרת רק ב- `env`.
- אין לחשוף URI למסד נתונים בקוד
- אסור לשמור שום מזהה משתמש של החייל (לא IP, לא `userId`, לא "name")

דרישות עיצוב (React UI)

המערכת צריכה להיראות נקייה, אחידה ורספונסיבית:

- רקע עדין (למשל `#f0f0f0`)
- Container מרכזי לבן עם `padding + border-radius + shadow` עדין
- כותרות ממורכזות
- הטופס בטור אחד, עם רווחים ברורים
- כפתורים ברורים (Primary לפעולה הראשית)
- התאמה למובייל: שימוש ביחידות יחסיות + `max-width`

דרישות בדיקות

בדיקות צד שרת

Unit Tests

- בדיקה ללוגיקה של יצירת תלונה (validation בסיסי: category חוקי, message לא ריק)
- בדיקה לשירות שמוסיף תלונה למסד (עם mocking למודל/DB)

Integration Tests

- `POST /api/complaints` → מחזיר הצלחה ושומר במסד (אפשר / in-memory / DB test (mock
- `GET /api/complaints` בלי אימות → `401/403`
- `POST /api/admin/login` עם סיסמה שגויה → `401`
- `POST /api/admin/login` עם סיסמה נכונה ואז `GET /api/complaints` → מחזיר רשימה

בדיקות צד לקוח (React)

- בדיקה שטופס "שליחת תלונה" שולח בקשה ומציג הודעת הצלחה (עם mock ל-API)
- בדיקה ש-`ProtectedRoute` חוסם `admin/` ללא אימות
- בדיקה ש-Admin page מציג תלונות אחרי (mock fetch)

שלבי עבודה מוצעים (פאזות)

פאזה 1 — הקמה ותשתיות

- יצירת `client` (React) ו-`server` (Express)
- הגדרת `env` לשרת
- חיבור למסד (MongoDB Atlas / Supabase)

פאזה 2 — API בסיסי

- מודל Complaint
- `POST /api/complaints`
- בדיקות בסיס ליצירה

פאזה 3 — UI בסיסי

- Home + Submit page
- טופס מלא + קריאה ל-API

פאזה 4 — Admin Auth + הגנה

- `POST /api/admin/login`
- שמירת auth בלקוח (token / session)
- `GET /api/complaints` מוגן
- `ProtectedRoute` בלקוח

פאזה 5 — בדיקות מקיפות + ניקיון ארכיטקטוני

- Unit + Integration לשרת
- בדיקות React
- טיפול שגיאות מסודר (ErrorPage / toast / messages)

אתגרים למתקדמים (לא חובה)

- הוספת `DELETE /api/complaints/:id` (מוגן אדמין)
 - גרף תלונות לפי קטגוריה (Client: chart)
 - "נשלח לפני X ימים" (human readable time)
 - API JSON + פילטרים: `api/complaints?category=food/`
-

screenshots:

Home Page

תיבת תלונות אנונימיות בבסיס צבאי

שלחו תלחת בצורה אנונימית ונשווה.

שליחת חלונה

תפקדים בלבה

ססימה:

כניסה לארסין

Submit Complaint Page

שליחת תלונת אנונימית

חחום חחלונה:

אוכל

וויכח חחלונה:

שכוכ כקיאח רחלונה שלך...

שליחת

חתלונה נשלחה בעלתה

Admin Login Page

כניסה לאודו האדפין

ססימה:

החתברות

סיסמת שפיה, נסת שוב.

Admin Complaints Page

רשימת חתלונות

נכלסכדציוהציידכקדוח

חחום	חוכח חחלונה	רשור האוריי
אוכל	האכל בחוד חור אוכל לא נססקו ררי.	4 א: 1. 2024, בחא
ציוד	חסר שר חדר שר כישוח כחת רשים.	3 א: 3. 2024, בחא
פקדות	הסקדדות לא ברוחם מככלות.	3 א: 3. 2024, בחא
אחר.	אוי שכל חחלושר מוקור.	09:20, 2024, בחא