# **טכנולוגיות אינטרנט מתקדמות - 61776 (WEB)**

**הגשת פרויקט**

**<** **virtual tour> < B4> <7>**

|  |  |
| --- | --- |
| **שם חבר.ת הצוות** | **תז** |
| הלאל חמוד | 213018104 |
| עינב פואד | 212697239 |
| עדן אבו אחמד | 212802169 |
| איליא אליאס | 208173021 |
| רובין חניפס | 207479874 |
| רוביר סליבא | 207431859 |

הפרויקט שלנו הוא **מערכת גלריה וירטואלית אינטראקטיבית**, המיועדת להצגת תערוכות דיגיטליות ופריטים אמנותיים במרחב מקוון. המערכת מיועדת לשני סוגי משתמשים: **לקוחות (מבקרים)** ו־**מנהלי מערכת**.

* **לקוחות (מבקרים)** יכולים להירשם ולהתחבר למערכת, לרכוש כרטיסים לצפייה בתערוכות, לדפדף בין גלריות שונות, ולצפות בפריטים הכוללים תיאורים מפורטים, תמונות וקטעי שמע ולעדכן את הפרטים שלו. מטרת חוויית המשתמש היא לספק תחושת "סיור במוזיאון" מהבית, בצורה נוחה ונעימה לשימוש.
* **מנהלי מערכת** מקבלים גישה לפאנל ניהול המאפשר להם לנהל את כלל תכני המערכת: יצירה, עדכון ומחיקה של גלריות ופריטים; צפייה ברשימת משתמשים וניהול תפקידיהם; מיפוי הגלריות על גבי מפה וירטואלית; ושליטה על נראות כל גלריה. בנוסף, מנהלים יכולים לגשת ללוח ניתוח צפיות מתקדם המציג גרפים אינטראקטיביים של כמות צפיות לכל גלריה ופריט.

המערכת שמה דגש על ממשק נוח, אבטחת מידע, חוויית משתמש מותאמת למובייל ודסקטופ, ותמיכה בזמן אמת בניתוח מידע.

|  | **טכנולוגיות** |
| --- | --- |
| Frontend | React.js, React Router, Axios |
| Backend | Node.js, Express, JWT |
| Database | MongoDB, Mongoose |
| Styling | Tailwind CSS, UI component library |

שמות הטכנולוגיות המרכזיות בכל אחד מהחלקים:

**לינק לגיט**

[**git link**](https://github.com/helalha7/GalleryXProject.git)

**לינק לאתר**

[**https://galleryx-tawny.vercel.app/**](https://galleryx-tawny.vercel.app/)

|  |  |  |
| --- | --- | --- |
| **שם חבר הצוות** | **משימות שהוקצו** | **משימות שהושלמו** |
| הלאל חמוד | שיבוץ גלריות על המפה, תצוגה אינטראקטיבית לפי מיקום, כפתור שינוי נראות (visibility toggle) | הושלם |
| עינב פואד | מימוש הוספת גלריה, עדכון/מחיקה של גלריות | הושלם |
| עדן אבו אחמד | הגדרת הרשאות אדמין, יצירת דוחות לפי גלריה ו־Artifact  גרף אינטראקטיבי | הושלם |
| איליא אליאס | הוספת artifact כולל טופס, צפייה/עריכה/מחיקה | הושלם |
| רובין חניפס | צפייה בדוחות צפיות לפי גלריות, פירוט לפי artifacts, חיבור לדאטה אמיתי מה־DB | הושלם |
| רוביר סליבא | בניית ממשק אדמין מלא – שילוב כפתורי ניהול, עיצוב עם Tailwind, ניווט בין רכיבי ניהול | הושלם |

**דרישות:**

* **דרישות פונקציונליות :**
* המשתמש יכול להירשם ולהתחבר כ־"מבקר" או "מנהל מערכת".
* מבקר מחובר יכול לרכוש כרטיסים באמצעות הממשק.
* מבקר יכול לדפדף בין גלריות ולצפות בפריטים עם תיאור, תמונה ושמע.
* כל פריט בגלריה מוצג עם מידע מפורט, תמונות ו/או קבצי אודיו.
* מנהל יכול לצפות ברשימת המשתמשים ולשנות להם תפקיד.
* מנהל יכול ליצור, לעדכן ולמחוק גלריות.
* מנהל יכול להוסיף, לערוך ולמחוק פריטים בכל גלריה.
* מנהל יכול להפעיל / להסתיר גלריות ולהציב אותן במפה וירטואלית.
* מערכת הניהול מציגה גרף ניתוח צפיות לכל גלריה.
* ריחוף בעכבר על נקודה בגרף מציג צפיות ברמת פריט בודד.
* **דרישות לא פונקציונליות :**

**Performance**

* זמן טעינת עמוד לא יעלה על 3 שניות בחיבור רגיל.
* המערכת תתמוך בלפחות 50 משתמשים פעילים בו־זמנית ללא ירידה בביצועים.

**Usability**

* הממשק יהיה רספונסיבי (Mobile + Desktop) וידידותי למשתמש.
* הגרפים יהיו אינטראקטיביים וברורים לפרשנות.

**Security**

* תהליך האימות יתבצע עם JWT או Session Storage בצורה מוצפנת.
* פעולות רכישה וניהול יהיו זמינות רק למשתמשים מחוברים ומורשים.
* כל פעולה ניהולית תדרוש הרשאות מתאימות מצד המנהל.

**Reliability & Availability**

* כל המידע (משתמשים, גלריות, פריטים) ישמר במסד נתונים MongoDB.
* נתוני צפיות יתעדכנו בזמן אמת או ברענון ידני של העמוד.

**Portability**

* המערכת תתמוך בפריסה על גבי פלטפורמות כמו Vercel

**- דרישות ממשק חיצוניות:**

**ממשק משתמש (UI)**

* React.js משמש לבניית רכיבים אינטראקטיביים.
* Tailwind CSS לעיצוב רספונסיבי ונעים למשתמש.
* ממשק הניהול כולל טפסים, גרפים ו־modals

**מסד נתונים (DB)**

* MongoDB משמש לאחסון נתוני המשתמשים, הגלריות, הפריטים והצפיות.
* Mongoose מנוהל בצד השרת עבור ORM

**API**

* REST API בנוי ב־Express.js ומטפל בפעולות CRUD ואימות.
* כל קריאה ל־API עוברת אימות ואימות הרשאות לפי סוג המשתמש.

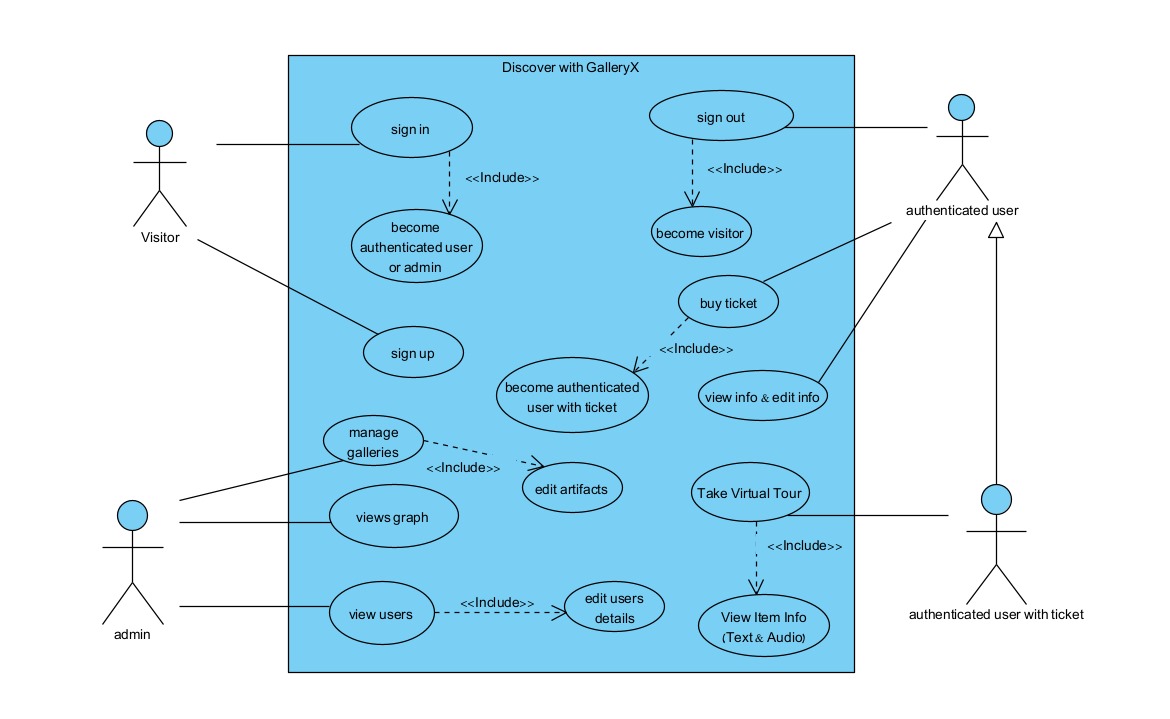
**שירותים חיצוניים**

* שימוש ב־JWT ( JSON Web Token)לאימות משתמשים.
* ספריות גרפים כמו Recharts או Chart.js להצגת צפיות.
* פריסה מתבצעת על Vercel עם בניית Production אוטומטית**.**

A computer screen shot of a diagram

AI-generated content may be incorrect.**ארכיטקטורה של האתר:**

**דיאגרמת use case:**



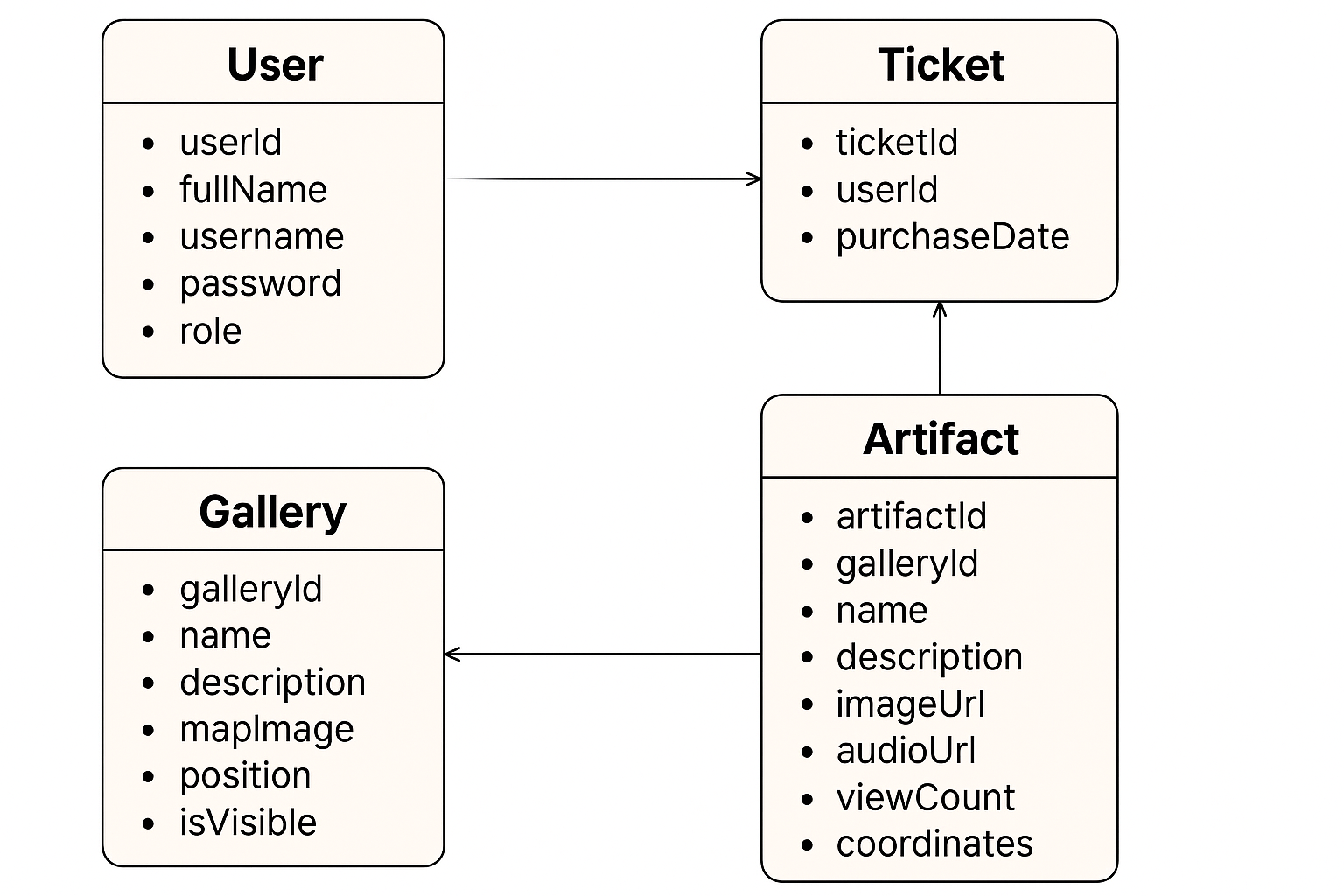
**דיאגרמת DB:**

הפרויקט משתמש ב־MongoDB כבסיס נתונים חיצוני בענן.  
כל פריטי המידע נשלפים בזמן אמת מה־DB דרך API שכתוב ב־Node.js/Express ומחובר עם mongoose.

להלן דיאגרמה המציגה את הקשרים בין הישויות:

- משתמשים מקושרים לכרטיסים  
 -כל כרטיס שייך למשתמש

-גלריות מכילות פריטים (Artifacts)  
 -כל פריט מקושר לגלריה

****

**תיק למתכנת:**

**הפונקציות המרכזיות בקוד:**

**useRequireAuth, useRequireAdmin, useRequireTicket**

* כל אחת מהן מונעת גישה לעמודים למי שלא מחובר או ללא הרשאות מתאימות.
* הן שומרות על **ההיגיון האבטחתי** של האתר.
* **שמירה על גישה תקינה היא קריטית**.
* מופיעות כמעט בכל עמוד עם תוכן דינמי → **בשימוש רחב בפרונטאנד**.

**artifactController.js**

**handleCreateArtifact(req)**  
מקבל פרטי פריט (artifact) בבקשת POST, ושולח אותם ל־createArtifactService.

* *מחזיר:*  
  סטטוס 201 עם הפריט החדש או שגיאה 400.

**handleGetAllArtifacts()**

שולף את כל הפריטים מה־DB.

* *מחזיר:*  
  סטטוס 200 עם מערך של פריטים.

**handleGetArtifactsByGallery(req)**  
מקבל שם גלריה מ־query string, ממיר לאותיות גדולות, ושולף את כל הפריטים של אותה גלריה.

* *מחזיר:*  
  סטטוס 200 עם רשימת הפריטים, או שגיאה אם הפרמטר חסר.

**handleIncrementArtifactViews(req)**  
מקבל שם פריט מ־query string, ומגדיל את מספר הצפיות עבורו במסד הנתונים.

* *מחזיר:*  
  סטטוס 200 אם הצליח, או 404 אם נכשל.
* **handleUpdateArtifact(req, { params })**
* מקבל מזהה פריט (id) ונתונים לעדכון מתוך גוף הבקשה. מעדכן את הפריט במסד הנתונים.
* *מחזיר:*  
  סטטוס 200 אם הצליח, או 400 אם נכשל.

**handleDeleteArtifact(req, { params })**

מוחק פריט לפי מזהה (id) שנשלח בנתיב הבקשה.

* *מחזיר:*  
  סטטוס 200 עם אישור מחיקה, או 404 אם לא נמצא.

**ticketController.js**

**purchaseTicketController(req)**  
בודק אם המשתמש מחובר (req.user). אם כן – יוצר כרטיס חדש עבורו דרך purchaseTicket.

* *מחזיר:*  
  תשובה עם סטטוס 201 Created וכרטיס חדש. אם המשתמש לא מחובר – סטטוס 401 Unauthorized.

**getTicketController(req)**

* מחזיר את הכרטיס של המשתמש המחובר (אם יש כזה), באמצעות getUserTicket.
* *מחזיר:*  
  סטטוס 200 OK עם נתוני הכרטיס, או 404 אם אין כרטיס קיים.

**galleryController.js**

**addGalleryController(req)**

* מקבלת בקשת HTTP עם נתוני גלריה בפורמט JSON.
* קוראת לפונקציה addGallery מהשכבת service לצורך יצירה במסד הנתונים.
* מחזירה תשובת Response עם סטטוס 201 (Created) ו־success: true.

**getGalleriesController()**

* לא מקבלת פרמטרים – שולפת את כל הגלריות ממסד הנתונים.
* מחזירה רשימה של גלריות עם סטטוס 200.

**getGalleryController(req, { params })**

* מקבלת מזהה גלריה (params.id) מתוך כתובת הבקשה.
* מבצעת שליפה מה־DB באמצעות fetchGalleryById.
* אם לא נמצא – מחזירה 404, אחרת מחזירה את הגלריה.

**updateGalleryController(req, { params })**

* מקבלת מזהה גלריה ו־JSON עם ערכים לעדכון.
* קוראת לפונקציית editGallery ומחזירה את הגלריה המעודכנת.
* תשובה עם סטטוס 200.

**deleteGalleryController(req, { params })**

* מקבלת מזהה גלריה ומוחקת אותה בעזרת removeGallery.
* מחזירה success: true עם סטטוס 200.

**updateUserController.js**

* מאפשר למשתמשים או למנהלים לשנות פרטים חשובים.
* כולל אימות מחדש, עדכון role או אימייל.
* חלק מה־**ניהול משתמשים והגדרות חשבון**.

**Middleware לאימות JWT**

* נבדק על כל API פרטי.
* מונע גישה לא מורשית ל־routes רגישים.
* ללא זה, כל האתר פרוץ → **קו ההגנה הקריטי של הבקאנד**.

בחרתי את אלה כי:

* **נוגעות ב־Core** של האפליקציה (גלריות, פריטים, כרטיסים, הרשאות)
* **מופעלות באופן תדיר** בפרונט ובבקאנד
* בלעדיהן – האתר **לא יהיה מאובטח ולא יתפקד**

API שהשתמשנו בהם:

|  | **מתודה** | **תיאור** |
| --- | --- | --- |
| /api/galleries | GET | שליפת כל הגלריות |
| /api/galleries | POST | יצירת גלריה חדשה |
| /api/galleries/:id | GET | שליפת גלריה לפי מזהה |
| /api/galleries/:id | PATCH | עדכון גלריה |
| /api/galleries/:id | DELETE | מחיקת גלריה |

|  | **מתודה** | **תיאור** |
| --- | --- | --- |
| /api/users/:id | PUT | עדכון משתמש (פרטים או תפקיד) |
| /api/users/:id | DELETE | מחיקת משתמש |

|  | **מתודה** | **תיאור** |
| --- | --- | --- |
| /api/ticket | POST | רכישת כרטיס |
| /api/ticket | GET | צפייה בכרטיס הקיים של המשתמש |
|  |
| /api/upload | POST | העלאת קובץ ל־Cloudinary |

|  | **מתודה** | **תיאור** |
| --- | --- | --- |
| /api/auth/login | POST | התחברות משתמש |
| /api/auth/register | POST | הרשמה למערכת |

**-Client (lib/api/) שימוש מהצד של ה**

| **פונקציה** | **מתודה** | **נתיב** | **תיאור** |
| --- | --- | --- | --- |
| login | POST | /api/auth/login | התחברות משתמש |
| register | POST | /api/auth/register | הרשמת משתמש |

:DB קישור ל

<mongodb+srv://helalha7:helal212@cluster0.vzaqe8w.mongodb.net/>

קטעי קוד מיוחדים:

| **מיקום בפרויקט** | **תיאור הקוד** | **רפרנס** |
| --- | --- | --- |
| lib/middleware/requireAuth.js | Middleware שמוודא שהמשתמש מחובר על פי JWT token | מבוסס על מדריך JWT באתר [Auth0](https://auth0.com/blog/create-api-using-express-and-jwt-authentication/) |
| core/services/cloudinaryService.js | קוד להעלאת קבצים ל-Cloudinary דרך API | תואם לדוקומנטציה של [Cloudinary Node SDK](https://cloudinary.com/documentation/node_image_upload) |
| hooks/useRequireAuth.js | שימוש ב־React hook לבדיקת הרשאות לפני טעינת דפים | השראה ממדריכים ב־[Next.js Documentation](https://nextjs.org/docs/app/building-your-application/routing/middleware) |
| route.js בקבצי API | מבנה סטנדרטי של API ב־Next.js App Router | מבוסס על [Next.js App Router examples](https://nextjs.org/docs/app/building-your-application/routing/route-handlers) |

פרומפטים שהשתמשנו בהם בכלי AI:

| **פרומפט שנכתב** | **שימוש בפרויקט** |
| --- | --- |
| "Write a controller function in Next.js to handle creating, updating, and deleting gallery items stored in MongoDB." | יצירת הפונקציות addGalleryController, updateGalleryController וכו' |
| "How to protect API routes in Next.js using middleware and JWT tokens?" | בסיס למימוש הקבצים requireAuth.js, requireAdmin.js |
| "Give me responsive component layout in Tailwind CSS for a sidebar and dashboard" | סייע בבניית רכיבים עיצוביים בממשק הניהול |

שמות משתמש וסיסמאות :

Admin: visitor:

Username :helal username :Rb

Password :helal password :rb

הסרטון של הפרויקט: