בס"ד

# פרויקט ניהול קורסים אונליין

## מבוא

בפרויקט זה תבנו מערכת לניהול קורסים אונליין, הכוללת צד לקוח עם Angular 17+ וצד שרת עם Node.js. המערכת תאפשר למורים ולתלמידים לנהל קורסים, לצפות בהרצאות ולבצע פעולות נוספות.

## מסכים לבנייה

### מסך התחברות/הרשמה

- **פונקציונליות**: התחברות משתמשים קיימים, הרשמה של משתמשים חדשים.

תפריט

**- פונקציונליות**: תפריט לניווט במערכת. מוצג תמיד

### דף הבית

### **- פונקציונליות**: הצגת תקציר על האתר/רקע

### מסך קורסים (תלמידים)

- **פונקציונליות**: הצגת רשימת קורסים, אפשרות להצטרף או לעזוב קורס.

### מסך קורסים שלי (תלמידים + מנהל)

- **פונקציונליות**: הצגת פרטי הקורס, רשימת שיעורים, אפשרות להורדת חומרים(-אופציונאלי).

### מסך ניהול קורסים (למורים + מנהל)

- **פונקציונליות**: הוספה, עריכה ומחיקה של קורסים ושיעורים.

## פונקציונליות מרכזית באנגולר

1. **Reactive Forms**: לניהול טפסים בצורה דינאמית ונוחה.
2. **Angular Services**: לניהול לוגיקה עסקית ושיתוף נתונים בין קומפוננטות.
3. **HTTP Client**: לתקשורת עם צד השרת.
4. **Routing**: לניהול ניווט בין מסכים באפליקציה.
5. **Component Communication**: לשיתוף נתונים ואירועים בין קומפוננטות.
6. **Angular Material**: לשימוש ברכיבים מוכנים לעיצוב ממשק המשתמש.
7. **Pipes**: לשינוי פורמט של נתונים בתצוגה**/ Custom Directives**: ליצירת התנהגות מותאמת אישית על אלמנטים.
8. **Input/Output**: להעברת נתונים ואירועים בין קומפוננטות.

## צד השרת ב-NODE

הצד של השרת יכלול API לניהול משתמשים, קורסים ושיעורים. כמו כן, יכלול אימות משתמשים וניהול הרשאות.

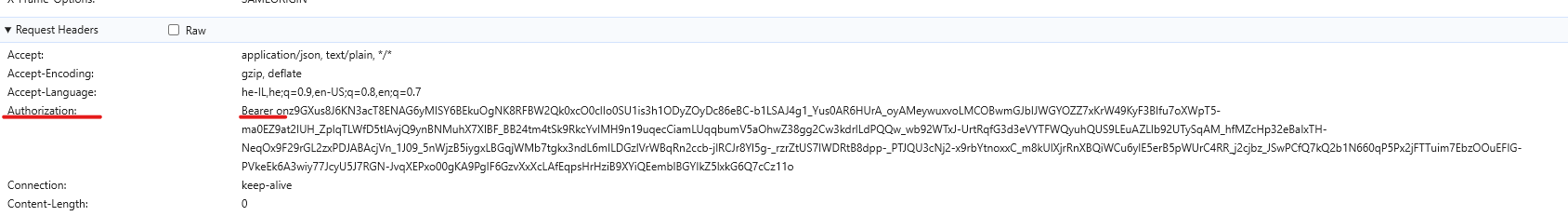
### דרישות:

- Node.js

- Express

- SQLite

### מבנה API: הערה לו2 😊 על מנת להעביר את הTOKEN עם כל קריאת API יש לשמור את הטוקן שמתקבל ב API של LOGIN לשמור ב seesion storage וליצור interceptor ושם לשלוף מה session storage ולצרף לAPI

שליחת הטוקן בצד האנגולר אמורה להיות ככה:  


#### Authentication:

* **POST** http://localhost:3000/api/auth/register - הרשמת משתמש חדש
  + **תיאור**: משמש לרישום משתמש חדש במערכת.
  + **הרשאות**: פתוח לכל המשתמשים (אין צורך ב-Token).
  + **פרמטרים**:
    - name (String): שם המשתמש.
    - email (String): כתובת דוא"ל של המשתמש.
    - password (String): סיסמה של המשתמש.
    - role (String): תפקיד המשתמש (student, teacher, admin).
  + **דוגמה לאובייקט JSON**:

{

"name": "John Doe",

"email": "john@example.com",

"password": "password123",

"role": "student"

}

* POST http://localhost:3000/api/auth/login - התחברות משתמש קיים
  + **תיאור**: משמש להתחברות משתמש קיים וקבלת Token.
  + **הרשאות**: פתוח לכל המשתמשים (אין צורך ב-Token).
  + **פרמטרים**:
    - email (String): כתובת דוא"ל של המשתמש.
    - password (String): סיסמה של המשתמש.
  + **דוגמה לאובייקט JSON**:

{

"email": "john@example.com",

"password": "password123"

}

#### Users:

* GET http://localhost:3000/api/users - קבלת כל המשתמשים (למנהל מערכת)
  + **תיאור**: משמש לקבלת רשימת כל המשתמשים במערכת.
  + **הרשאות**: נדרש Token של משתמש מנהל מערכת.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
* GET http://localhost:3000/api/users/:id - קבלת פרטי משתמש לפי ID
  + **תיאור**: משמש לקבלת פרטי משתמש לפי מזהה.
  + **הרשאות**: נדרש Token של משתמש מחובר.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
* PUT http://localhost:3000/api/users/:id - עדכון פרטי משתמש לפי ID
  + **תיאור**: משמש לעדכון פרטי משתמש לפי מזהה.
  + **הרשאות**: נדרש Token של משתמש מחובר.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
  + **פרמטרים**:
    - name (String): שם המשתמש.
    - email (String): כתובת דוא"ל של המשתמש.
    - password (String): סיסמה של המשתמש.
    - role (String): תפקיד המשתמש (student, teacher, admin).
  + **דוגמה לאובייקט JSON**:

{

"name": "John Doe",

"email": "john@example.com",

"password": "newpassword123",

"role": "teacher"

}

* DELETE http://localhost:3000/api/users/:id - מחיקת משתמש לפי ID
  + **תיאור**: משמש למחיקת משתמש לפי מזהה.
  + **הרשאות**: נדרש Token של משתמש מחובר.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.

#### Courses:

* GET http://localhost:3000/api/courses - קבלת כל הקורסים
  + **תיאור**: משמש לקבלת רשימת כל הקורסים במערכת.
  + **הרשאות**: נדרש Token של משתמש מחובר.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
* GET http://localhost:3000/api/courses/:id - קבלת פרטי קורס לפי ID
  + **תיאור**: משמש לקבלת פרטי קורס לפי מזהה.
  + **הרשאות**: נדרש Token של משתמש מחובר.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
* POST http://localhost:3000/api/courses - יצירת קורס חדש (למורים)
  + **תיאור**: משמש ליצירת קורס חדש במערכת.
  + **הרשאות**: נדרש Token של מורה.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
  + **פרמטרים**:
    - title (String): כותרת הקורס.
    - description (String): תיאור הקורס.
    - teacherId (Number): מזהה המורה היוצר את הקורס.
  + **דוגמה לאובייקט JSON**:

{

"title": "Introduction to Programming",

"description": "Learn the basics of programming.",

"teacherId": 1

}

* PUT http://localhost:3000/api/courses/:id - עדכון קורס לפי ID (למורים)
  + **תיאור**: משמש לעדכון פרטי קורס לפי מזהה.
  + **הרשאות**: נדרש Token של מורה.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
  + **פרמטרים**:
    - title (String): כותרת הקורס.
    - description (String): תיאור הקורס.
    - teacherId (Number): מזהה המורה היוצר את הקורס.
  + **דוגמה לאובייקט JSON**:

{

"title": "Advanced Programming",

"description": "Learn advanced programming concepts.",

"teacherId": 1

}

* DELETE http://localhost:3000/api/courses/:id - מחיקת קורס לפי ID (למורים)
  + **תיאור**: משמש למחיקת קורס לפי מזהה.
  + **הרשאות**: נדרש Token של מורה.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.

#### Lessons:

* GET http://localhost:3000/api/courses/:courseId/lessons - קבלת כל השיעורים בקורס מסוים
  + **תיאור**: משמש לקבלת רשימת כל השיעורים בקורס מסוים.
  + **הרשאות**: נדרש Token של משתמש מחובר.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
* GET http://localhost:3000/api/courses/:courseId/lessons/:id - קבלת פרטי שיעור לפי ID
  + **תיאור**: משמש לקבלת פרטי שיעור לפי מזהה.
  + **הרשאות**: נדרש Token של משתמש מחובר.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
* POST http://localhost:3000/api/courses/:courseId/lessons - יצירת שיעור חדש בקורס מסוים (למורים)
  + **תיאור**: משמש ליצירת שיעור חדש בקורס מסוים.
  + **הרשאות**: נדרש Token של מורה.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
  + **פרמטרים**:
    - title (String): כותרת השיעור.
    - content (String): תוכן השיעור.
    - courseId (Number): מזהה הקורס אליו שייך השיעור.
  + **דוגמה לאובייקט JSON**:

{

"title": "Introduction to Variables",

"content": "Learn about variables in programming.",

"courseId": 1

}

* PUT http://localhost:3000/api/courses/:courseId/lessons/:id - עדכון שיעור לפי ID (למורים)
  + **תיאור**: משמש לעדכון פרטי שיעור לפי מזהה.
  + **הרשאות**: נדרש Token של מורה.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.
  + **פרמטרים**:
    - title (String): כותרת השיעור.
    - content (String): תוכן השיעור.
    - courseId (Number): מזהה הקורס אליו שייך השיעור.
  + **דוגמה לאובייקט JSON**:

{

"title": "Advanced Variables",

"content": "Learn about advanced uses of variables.",

"courseId": 1

}

* DELETE http://localhost:3000/api/courses/:courseId/lessons/:id - מחיקת שיעור לפי ID (למורים)
  + **תיאור**: משמש למחיקת שיעור לפי מזהה.
  + **הרשאות**: נדרש Token של מורה.
  + **מיקום ה-Token**: יש לשלוח את ה-Token בכותרת Authorization כ: Bearer <your\_token>.

## הורדת והפעלת הפרויקט

### הורדת הפרויקט מ-GitHub

1. פתחי את שורת הפקודה או הטרמינל.
2. נווטי לתיקייה בה תרצי לשמור את הפרויקט.
3. הריצי את הפקודה הבאה להורדת הפרויקט מ-GitHub:

Git clone https://github.com/rivkamos/CourseOnlineServer.git

1. כנסי לתיקיית הפרויקט שהורדתם:

cd courses-online

### התקנת התלויות

1. התקיני את כל התלויות הנדרשות באמצעות הפקודה:

npm install

### הרצת השרת

1. הריצי את השרת באמצעות הפקודה:  
   node index.js
2. השרת יפעל על פורט 3000. את יכולה לגשת אליו באמצעות דפדפן בכתובת: http://localhost:3000.

### שמירת ה-DB בקובץ מקומי

הפרויקט שומר את מסד הנתונים בקובץ מקומי בשם  your-database-file.db בתיקיית הפרויקט. אין צורך לבצע כל פעולה נוספת לשם כך.

### מבנה הפרויקט

online-courses/

├── models/

│ ├── User.js

│ ├── Course.js

│ └── Lesson.js

├── routes/

│ ├── auth.js

│ ├── users.js

│ ├── courses.js

│ └── lessons.js

├── middleware/

│ └── auth.js

├── index.js

├── package.json

└── package-lock.json

## דוגמאות להרצת API דרך הפוסטמן

שימי לב בהורדת הפרויקט מהגיט יש קובץ בשם Course Online.postman\_collection.json  
טעני אותו לפוסטמן ותוכלי לראות כמה דוגמאות להרמת API

בהצלחה בפרויקט!