פרויקט מתכונים צד שרת - ***node***

בפרויקט זה עלייך לבנות שרת *RESTful API* בטכנולוגיית *express* עם חיבור ל-*mongo atlas*.

הסבר כללי

שמירת המידע

הנתונים יישמרו במסד נתונים של מונגו אטלס (יש לאפשר גישה ל-*cluster* מכל כתובות ה-*IP*).

עלייך לשמור לכל ישות (לפחות) את הנתונים הבאים:

- **משתמש -** שם משתמש, סיסמא (בדקי שנשלחה סיסמא חזקה), מייל (ייחודי), כתובת, תפקיד (מנהל/משתמש רשום וכו')
- **מתכון -** שם, תיאור, קטגוריה (או לחילופין מערך קטגוריות), זמן הכנה בדקות, רמת קושי (1-5), תאריך הוספה (ברירת מחדל), מערך שכבות (לכל שכבה יש לשמור: תיאור ומרכיבים), הוראות הכנה (מערך), תמונה (מחרוזת), האם פרטי? (אם כן - הוא יוחזר רק למשתמש שיצר אותו, אם לא - יוחזר לכולם), משתמש שהוסיף את המתכון (חשבי כיצד כדאי לשומרו).
- **קטגוריות -** קוד, תיאור, מס' מתכונים לקטגוריה זו, מערך מתכונים.

הגדירי אילוצים מתאימים על הסכמות (תוכלי להוסיף אילוצים נוספים על הנ"ל) וכן ערכי ברירת מחדל.

הפעולות

השרת יכיל את הפעולות הבאות:

- **משתמשים:**
  - התחברות
  - הרשמה
  - קבלת כל המשתמשים
  - מחיקת משתמש/עדכון סיסמא
- **מתכונים:**
  - קבלת כל המתכונים - *:/recipes?search=aaa&limit=5&page=2*
    - אפשרי שליפה של כל המתכונים גם לפי חיפוש טקסט חופשי וגם בהגבלת כמות לעמוד.
    - קבלת המתכונים למשתמש מסוים - לפי הטוקן
  - קבלת פרטי מתכון לפי קוד מתכון - */recipes/1111*
  - קבלת מתכון לפי זמן הכנה (מקבל עד כמה דקות ומחזירה את המתכונים שמשך ההכנה שלהם עד מספר דקות שהתקבל)
  - הוספת מתכון
  - עדכון מתכון
  - מחיקת מתכון
- **קטגוריות:**

(שימי לב שאין פונקציה של הוספת/עדכון קטגוריה, הקטגוריות יתווספו/יתעדכנו דרך הוספת/מחיקת מתכון)

  - קבלת כל הקטגוריות.
  - קבלת כל הקטגוריות וכל המתכונים לכל קטגוריה.
  - קבלת קטגוריה לפי קוד (או שם) וכל המתכונים בקטגוריה זו.

הרשאות משתמשים

קיימים 3 סוגי משתמשים: מנהל, משתמש רשום, משתמש אורח (לא רשום).

אפשרי הפעלת כל פעולה לפי הרשאת משתמש מתאימה.

דרישות חובה

1. לאורך **כל** הפרויקט השתמשי ב-*git* והעלי את השינויים ל-*GitHub*.
2. התקיני את הספריות הנדרשות לבניית השרת ולחיבור למסד הנתונים.
3. אפשרי גישה לכל הכתובות מצד לקוח (בהמשך תוכלי לתת גישה רק לפרויקט אנגולר מכתובת: *http://localhost:4200*).
4. בני את הפרויקט בצורה מסודרת עם חלוקה לתיקיות וקבצי קונפיגורציה (משתני סביבה) כפי שנלמד בכיתה.
5. בני סכמה ומודל לכל אחת מהישויות שנשמרות במסד הנתונים.
6. הגדירי בדיקות תקינות על הנתונים שמתקבלים מהלקוח (ע"י ספרית *joi*).
7. צרי *middleware* ללכידת שגיאות, כך שכל השגיאות יהיו בפורמט הבא: *{ error: { message: '...' } }*, צרי בנוסף גם *middleware* עבור נתיב שלא נמצא.
8. אימות המשתמש יהיה באמצעות *jwt*. צרי *middlewares* נדרשים עבור: אימות משתמש, אימות מנהל (ניתן ליצור *middlewares* נוספים לבחירתך).
9. הקפידי לעבור לפי פרוטוקול *REST*:
   - *Method*
   - *correct url*
   - הקפידי להחזיר סטטוס מתאים בכל בקשה (גם בשגיאות).
10. בדקי את השרת שבנית דרך *postman/thunder-client*, יש ליצור *collection* לכל *resource* ולייצא אותם (או לחילופין ליצור *collection* אחד לכל הפרויקט שבתוכו תיקיות לכל *resource*).
11. בני טבלה שמתארת את ה-*API* (בקובץ *excel/html* או בקובץ *ReadMe.md*), העלי אותה ג"כ ל-*GitHub*.

דרישות נוספות - בונוסים

- העלי את הפרויקט לשרת חיצוני (למשל *render, Netlify* וכו')
- אפשרי העלאת תמונה דרך השרת (*multer*)
- השתמשי בספריות נוספות שלא נלמדו (למשל: שליחת מייל)
- השתמשי בשאילתות מעניינות כמו: *aggregate* ועוד
- כל דבר אחר שהוא מעבר לדרישות שצוינו יכול להוסיף נקודות...