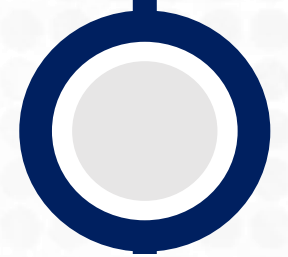


Node.js

Data Access Service & Data Seed

18/12/2024

Data Seed



Data Seed

בחלק זה נכיר ונלמד מה היא "זריעת נתונים" ומתי נרצה לעשות זאת.
בסיום הנושא תוכלו לענות על השאלות הבאות:

```
Server is running on port 8080
Connected to the database
User created: john@email
User created: jane@email
User created: jim@email
```

- מה זה "זריעת נתונים" (Data Seed)?
- כיצד נוכל ליצור מסמכים מראש בעת הרצת התוכנית במידה והם לא קיימים עדיין?
- מה זה נותן לנו?
- מדוע כדאי לנו להחזיק נתונים אלו בקובץ JSON?

```
[
  {
    "username": "John Doe",
    "email": "john@email",
    "password": "password"
  },
  {
    "username": "Jane Melony",
    "email": "jane@email",
    "password": "password"
  },
  {
    "username": "Jim Berry",
    "email": "jim@email",
    "password": "password",
    "authLevel": 2
  }
]
```

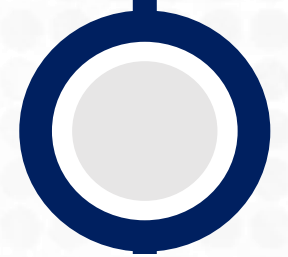
```
Server is running on port 8080
Connected to the database
User created: john@email
User created: jane@email
User created: jim@email
```

```
[
  {
    "_id": "6740c7ce4fbec3a9d63b4026",
    "username": "John Doe",
    "password": "password",
    "email": "john@email",
    "authLevel": 1,
    "__v": 0
  },
  {
    "_id": "6740c7ce4fbec3a9d63b4027",
    "username": "Jane Melony",
    "password": "password",
    "email": "jane@email",
    "authLevel": 1,
    "__v": 0
  },
  {
    "_id": "6740c7ce4fbec3a9d63b4028",
    "username": "Jim Berry",
    "password": "password",
    "email": "jim@email",
    "authLevel": 2,
    "__v": 0
  }
]
```

המשיכו את התרגיל הקודם ופתרו את התרגילים לפי הסדר.

תרגיל	תיאור המשימה
Ex-1	צרו קובץ JSON ובו מערך עם נתונים של 3 משתמשים.
Ex-2	צרו פונקציה שבמידה ומשתמשים אלו עדיין לא קיימים –יוצרת אותם באמצעות שימוש בקובץ ה-JSON.
Ex-3	קראו לפונקציה זו בעת הפעלת השרת ובדקו שהנתונים נוספו.
Ex-4	מחקו משתמשים אלו, הפעילו את השרת מחדש ובדקו שהמשתמשים נוצרו מחדש לאחר ההפעלה.

Data Access Service



Data Access Service

בחלק זה נכיר ונלמד מה הוא **Data Access Service** ובמה הוא תורם לנו לארגון ומבנה השרת. בסיום הנושא תוכלו לענות על השאלות הבאות:

```
router.get('/:id', auth, async (req, res) => {  
  try {  
    const user = await getOne(req.params.id);  
    return res.json(user);  
  } catch (error) {  
    return res.status(500).json({ message: error.message });  
  }  
});
```

- מה הוא "Data Access Service"?
- מדוע שנרצה למדל את כל הפעולות שמתבצעות מול מסד הנתונים לקובץ שונה?
- מדוע שלא נכתוב לוגיקה זו ישירות בפונקציית ה-callback של הנתוב?
- באיזה תיקייה נרצה למקם שירותים כאלו ואחרים ל-users?
- מה נותן לנו האיגוד של כל התיקיות שתחת התיקייה "users"?

```
const getOne = async (id) => {  
  const user = User.findById(id);  
  
  if (!user) {  
    throw new Error('User not found');  
  }  
  
  return user;  
};
```

Data Access Service - תרגול

המשיכו את התרגיל הקודם ופתרו את התרגילים לפי הסדר.

```
const password = req.headers["x-auth-token"];

if (password !== PASSWORD) {
  return res.status(403).json({ message: "Unauthorized" });
}
```

תרגיל	תיאור המשימה
Ex-1	צרו תיקייה בשם <code>services</code> עבור הקובץ <code>.dataAccesService.js</code> .
Ex-2	כתבו בקובץ פונקציות עבור <code>הוספת\מחיקת\קריאת\עדכון</code> משתמשים וייצאו אותן.
Ex-3	ייבאו את הפונקציות לקובץ הנתיבים של <code>users</code> והשתמשו בהן בנתיבים התואמים.
Ex-4	הוסיפו נתיב עבור קריאות <code>PATCH</code> שישנה רק את השדה <code>authLevel</code> של המשתמש, אם כרגע הערך באובייקט המקורי הוא 1 – אז ישתנה ל-2, אחרת – ישתנה ל-1.
Ex-5	דרשו בכל הנתיבים שיצרתם <code>header</code> בשם <code>['x-auth-token']</code> , על מנת להיות תקין, עליו להכיל סיסמה אשר תואמת לסיסמה שתקבעו מראש בקובץ ה- <code>env</code> . במידה והמשתמש לא שלח את הסיסמא הנכונה ב- <code>header</code> בעת בקשה לנתיב – החזירו לו שגיאה מסוג 403 בעלת הודעה תואמת. אתגר: עשו זאת באמצעות שימוש ב- <code>middleware</code> .

תודה על ההקשבה

אני וצוות המכללה כאן עבורכם