

# תרגול 10 - Node.js & Client

כל הזכויות שמורות לקרן יעקב 2024

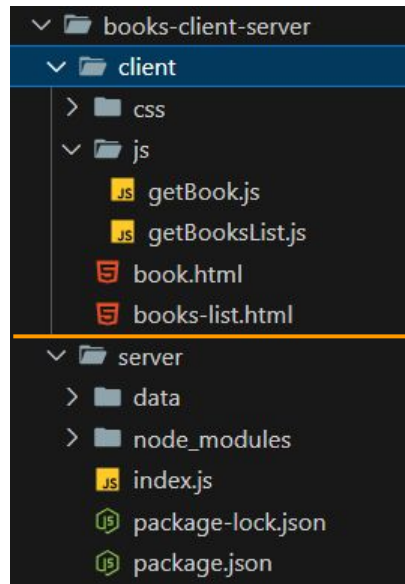
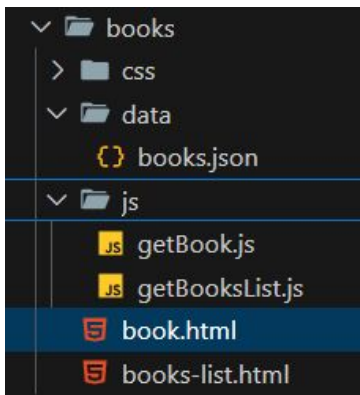


# 1. טעינת JSON בצד השרת

# תרגיל 1 – טעינת JSON בצד השרת


- ניקח את הקוד של תרגיל הספרים מתרגול מס' 7 ונעביר את ה-JSON עם המידע לצד השרת. הקוד נמצא במודל - "10.3 - קוד התחלתי לתרגול - ספרים"
- מה נצטרך לשנות עם המעבר הזה - מבחינת הקבצים? מבחינת הקוד?
- ראינו ששתי האופציות עובדות, אבל איפה יותר נכון לשים את ה-JSON עם המידע, בצד הלקוח או בצד השרת?

# תרגיל 1 - חלוקה לתיקיות



- ניצור תיקיה חדשה ובה תהיינה שתי תיקיות - אחת לצד השרת ואחת לצד הלקוח.
- תיקיית צד השרת ותיקיית צד הלקוח לא ישבו ביחד, אלא על שרתים נפרדים.
- את הקוד ההתחלתי נשים בתיקיית צד הלקוח, בלי תיקיית data ששם יושב ה-json עם המידע
- בתיקיית צד השרת ניצור קובץ index.js
- נעביר את תיקיית data עם ה-json של המידע
- נאתחל את הפרוייקט ע"י npm init
- נוסיף את הספריה express לפרוייקט
- נרשום את הקוד ההתחלתי להרמת השרת


# תרגיל 1 - הרמת שרת

Shenkar2024Keren > 10.1-Nodejs-lec > books-client-server > server >  index.js

```
1  const express = require('express');
2  const app = express();
3  const port = process.env.PORT || 8080;
4
5  app.listen(port);
6  console.log(`listening on port ${port}`);
```

● כדי לפנות מצד הלקוח לצד השרת ולבקש את ה-json עם המידע, מה נצטרך להכין בצד השרת?

# תרגיל 1 – טעינת ה-json והכנת route

```
Shenkar2024Keren > 10.1-Nodejs-lec > books-client-server > server >  index.js
1  const express = require('express');
2  const app = express();
3  const port = process.env.PORT || 8080;
4  const booksData = require('./data/books.json');
5
6  app.get("/books", (req, res) => {
7    |   res.json(booksData);
8  });
9
10 app.listen(port);
11 console.log(`listening on port ${port}`);
```

- הוספנו את טעינת ה-json ויצרנו route שמאפשר את קבלת המידע
- איך נוכל לבדוק את הקוד מצד הלקוח?

# תרגיל 1 – הרצת השרת ובדיקת הקוד

```
← → ↺ ⓘ localhost:8080/books

{
  "category": "Books",
  "products": [
    {
      "id": "123",
      "name": "Cinderella",
      "price": 16.65
    },
    {
      "id": "456",
      "name": "Snow White",
      "price": 7.03
    },
    {
      "id": "789",
      "name": "Mickey Mouse",
      "price": 9.23
    }
  ]
}
```

● ננסה להפעיל את הראוט מהדפדפן או מפוסטמן... עובד!

● עכשיו אפשר להריץ את הקוד שלנו בצד הלקוח

# תרגיל 1 - צד הלקוח

```
JS getBooksList.js X
Shenkar2024Keren > 10.1-Nodejs-lec > books-client-server > c
17 }
18
19 window.onload = () => {
20   fetch("http://127.0.0.1:8080/books")
21     .then(response => response.json())
22     .then(data => showData(data));
23 }
```

● נצטרך לשנות ב-fetch את הנתיב, במקום לקרוא לקובץ ה-json שהיה בצד הלקוח, נפעיל את הראוט בצד השרת ונקבל את המידע.

● נפתח את העמוד books-list.html

● אנחנו רואים רק את הכותרת **Products**

● נפתח את הקונסול כדי לבדוק אם יש שגיאות

```
top Filter All levels 2 Issues: 1 1
Access to fetch at 'http://127.0.0.1:8080/books' from origin 'http://127.0.0.1:5500' has been blocked by CORS policy: No 'Access-Control-Allow-Origin' header is present on the requested resource. If an opaque response serves your needs, set the request's mode to 'no-cors' to fetch the resource with CORS disabled.
GET http://127.0.0.1:8080/books net::ERR_FAILED 200 (OK) getBooksList.js:20
Uncaught (in promise)
TypeError: Failed to fetch
    at window.onload (getBooksList.js:20:5)
```



# תרגיל 1 – בעיית CORS

- בעיית CORS פותרים בצד השרת, ע"י מתן גישה לגשת מדומיין או דומיינים אחרים
- נוסיף את ה-middleware לצד השרת כמו בצילום המסך, מה יקרה?

```
index.js X
Shenkar2024Keren > 10.1-Nodejs-lec > books-client-server > server > index.js > ...
1  const express = require('express');
2  const app = express();
3  const port = process.env.PORT || 8080;
4  const booksData = require('./data/books.json');
5
6  app.get("/books", (req, res) => {
7    res.json(booksData);
8  });
9
10 app.use((req, res, next) => {
11   res.set({
12     'Access-Control-Allow-Origin': '*',
13     'Access-Control-Allow-Headers': 'Origin, X-Requested-With, Content-Type, Accept',
14     'Access-Control-Allow-Methods': 'GET, POST, PUT, DELETE',
15     'Content-Type': 'application/json'
16   });
17   next();
18 });
19
20 app.listen(port);
21 console.log(`listening on port ${port}`);
```

# תרגיל 1 – פתרון בעיית CORS

הבעיה לא נפתרה. צריך לשים את ה-middleware במקום הנכון, לפני שניגשים לראוט GET

עבשיו נפתח שוב את העמוד

books-list.html

עובד!

נתקן גם את הקריאה בקובץ

getBook.js

```
index.js X
Shenkar2024Keren > 10.1-Nodejs-lec > books-client-server > server > index.js > ...
1  const express = require('express');
2  const app = express();
3  const port = process.env.PORT || 8080;
4  const booksData = require('./data/books.json');
5
6  app.use((req, res, next) => {
7    res.set({
8      'Access-Control-Allow-Origin': '*',
9      'Access-Control-Allow-Headers': 'Origin, X-Requested-With, Content-Type, Accept',
10     'Access-Control-Allow-Methods': "GET, POST, PUT, DELETE",
11     'Content-Type': 'application/json'
12   });
13   next();
14 });
15
16 app.get("/books", (req, res) => {
17   res.json(booksData);
18 });
19
20 app.listen(port);
21 console.log(`listening on port ${port}`);
```

## Books

## Products

- [Cinderella](#)
- [Snow White](#)
- [Mickey Mouse](#)

# תרגיל 1 – קוד צד שרת

```
const express = require('express');
const app = express();
const port = process.env.PORT || 8080;
const booksData = require('./data/books.json');

app.use((req, res, next) => {
  res.set({
    'Access-Control-Allow-Origin': '*',
    'Access-Control-Allow-Headers': 'Origin, X-Requested-With, Content-Type, Accept',
    'Access-Control-Allow-Methods': 'GET, POST, PUT, DELETE',
    'Content-Type': 'application/json'
  });
  next();
});

app.get("/books", (req, res) => {
  res.json(booksData);
});

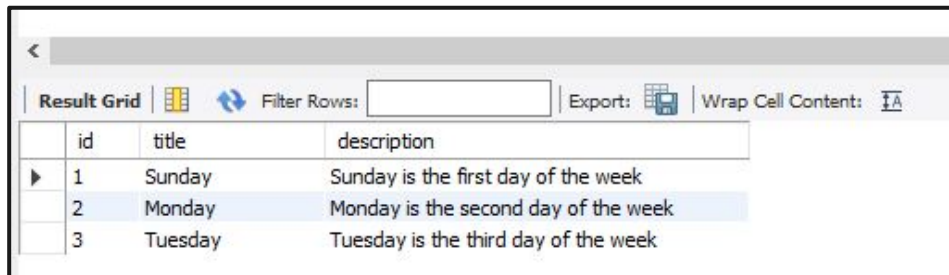
app.listen(port);
console.log(`listening on port ${port}`);
```



# 2. התחברות לדטבייס

## תרגיל 2 – התחברות לדטבייס

● בהכנה לשיעור התבקשתם ליצור כל אחד טבלה משלו עם 3 רשומות:

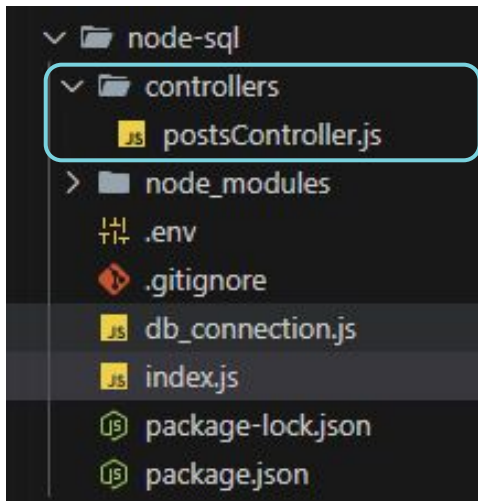


|   | id | title   | description                          |
|---|----|---------|--------------------------------------|
| ▶ | 1  | Sunday  | Sunday is the first day of the week  |
|   | 2  | Monday  | Monday is the second day of the week |
|   | 3  | Tuesday | Tuesday is the third day of the week |

● נשתמש בחיבור לדטבייס ובטבלה שיצרנו כדי לבצע פעולות על הדטבייס – הבאת כל המידע ועדכון רשומה

## תרגיל 2 – ניצור קונטרולר

- את הלוגיקה לא נכניס לקובץ הראשי של השרת ולכן ניצור תיקיית controllers בתיקיית הפרוייקט ושם ניצור את postsController.js




- הקריאה לקונטרולר תעשה מתוך קובץ index.js

```
6 const { postsController } = require('./controllers/postsController.js');
```

- והשימוש יעשה (כרגע) מתוך הראוטים השונים, ניצור מתודה להבאת כל הפוסטים:

```
19 app.get("/posts", async (req, res) => {  
20   res.status(200).send(await postsController.getPosts());  
21 });
```



## תרגיל 2 – ניצור פונקציות בקונטרולר

Shenkar2024Keren > 10.1-Nodejs-lec > node-sql > controllers > postsController.js > ...

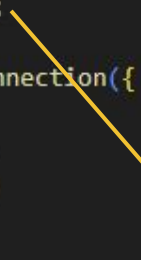
```
1  exports.postsController = {
2    // GET localhost:8081/posts
3    async getPosts() {
4      const { dbConnection } = require('../db_connection');
5      const connection = await dbConnection.createConnection();
6
7      const [rows] = await connection.execute(`SELECT * from tbl_99_posts`);
8
9      connection.end();
10
11     return rows;
12   },
13   // GET localhost:8081/posts/2
14   async getPost(postId) { ...
23   },
24   // POST localhost:8081/posts
25   async addPost(body) { ...
34   },
35   // PUT localhost:8081/posts/5
36   async updatePost(body, postId) { ...
45   },
46   // DELETE localhost:8081/posts/5
47   async deletePost(postId) { ...
56   }
57 };
```

- נתחיל עם הפלואו של הבאת כל הפוסטים
- למה השתמשנו בקוד אסינכרוני? מה היה קורה ללא קוד אסינכרוני?

## תרגיל 2 - קובץ התחברות לדטבייס

Shenkar2024Keren > 10.1-Nodejs-lec > node-sql > db\_connection.js > ...

```
1  exports.dbConnection = {  
2    async createConnection() {  
3      const mysql = require('mysql2/promise');  
4  
5      const connection = await mysql.createConnection({  
6        host      : process.env.DB_HOST,  
7        user      : process.env.DB_USERNAME,  
8        password  : process.env.DB_PASSWORD,  
9        database  : process.env.DB_NAME  
10     });  
11  
12     return connection;  
13   }  
14 }
```



- החיבור לדטבייס משותף לכולם.
- בכל פעם שנרצה לגשת לדטבייס, נפתח את החיבור ובסוף השימוש נסגור אותו (הקוד הזה בקונטרולר)
- מאיפה מגיעים פרטי החיבור לדטבייס? (process.env)
- נצטרך להוסיף לפרוייקט את הספריה mysql2



## תרגיל 2 – קוד להעתקה עד עכשיו – הרמת שרת

```
require('dotenv').config();
const express = require('express');
const app = express();
const port = process.env.PORT || 8080;

const { postsController } = require('./controllers/postsController.js');

app.use(express.json());
app.use(express.urlencoded({extended: true}));

app.use((req, res, next) => {
  res.set('Access-Control-Allow-Origin', '*');
  res.set('Access-Control-Allow-Headers', 'Origin, X-Requested-With, Content-Type, Accept');
  res.set('Access-Control-Allow-Methods', 'GET, POST, PUT, DELETE');
  res.set('Content-Type', 'application/json');
  next();
});

app.get("/posts", async (req, res) => {
  res.status(200).send(await postsController.getPosts());
});

app.listen(port);
console.log(`listening on port ${port}`);
```

## תרגיל 2 - קוד להעתקה עד עכשיו - קונטרולר

```
exports.postsController = {  
  // GET localhost:8081/posts  
  async getPosts() {  
    const { dbConnection } = require('../db_connection');  
    const connection = await dbConnection.createConnection();  
  
    const [rows] = await connection.execute(`SELECT * from tbl_99_posts`);  
  
    connection.end();  
  
    return rows;  
  }  
};
```

## תרגיל 2 - קוד להעתקה עד עכשיו - דטבייס

```
exports.dbConnection = {  
  async createConnection() {  
    const mysql = require('mysql2/promise');  
  
    const connection = await mysql.createConnection({  
      host      : process.env.DB_HOST,  
      user      : process.env.DB_USERNAME,  
      password  : process.env.DB_PASSWORD,  
      database  : process.env.DB_NAME  
    });  
  
    return connection;  
  }  
}
```

## תרגיל 2 – עדכון פוסט

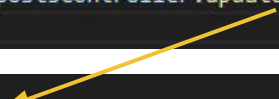
- עכשיו אתם לבד – נממש את הפלואו של עדכון פוסט, דברים לחשוב עליהם:
  - בצד הלקוח – נפתח את פוסטמן, נכניס את המתודה ואת הכתובת הנכונה, אילו עוד פרטים נצטרך?
  - בצד השרת – איזה ראוט יקבל את הבקשה לעדכון? איך נוציא משם את המידע המתאים?
  - בקונטרולר – איזו פונקציה לממש ואיזו שאילתת SQL נבצע?
  - בסוף נבדוק שהכל עובד ואם לא – נתקן

## תרגיל 2 – עדכון פוסט – ראוט ומתודה

- בשביל העדכון, נצטרך לקבל מהמשתמש גם id בראוט וגם body עם הפרטים שהשתנו
- בקונטרולר נפתח חיבור לדטבייס, נבצע את השאילתה של העדכון ונסגור את החיבור לדטבייס.
- כרגע נחזיר true אבל בהמשך נטפל בשגיאות שיכולות לקרות במהלך העדכון.

```
23 app.put("/posts/:postId", async (req, res) => {  
24   const { body } = req;  
25   res.status(200).send(await postsController.updatePost(body, req.params.postId));  
26 });
```

```
13 // PUT localhost:8081/posts/5  
14 async updatePost(body, postId) {  
15   const { dbConnection } = require('../db_connection');  
16   const connection = await dbConnection.createConnection();  
17  
18   await connection.execute(`UPDATE tbl_99_posts SET title = "${body.title}", description = "${body.description}" WHERE id=${postId}`);  
19  
20   connection.end();  
21  
22   return true;  
23 }
```



## תרגיל 2 – עדכון פוסט – צד הלקוח

The screenshot shows a REST client interface with a PUT request to `localhost:8081/posts/4`. The 'Body' tab is selected, and the request body is a JSON object: `{ "title": "This is a new title 5", "description": "This is a new description 5" }`. The 'Headers' tab shows 9 headers, and the 'Test Results' tab shows a response of `1 true`.

```
1 {
2   "title": "This is a new title 5",
3   "description": "This is a new description 5"
4 }
```

Body Cookies Headers (10) Test Results

Pretty Raw Preview Visualize JSON

```
1 true
```

● נבצע עדכון בדטבייס ע"י הפוסטמן

● נבדוק בדטבייס שאכן הרשומה השתנתה

|   | id   | title                 | description                          |
|---|------|-----------------------|--------------------------------------|
| ▶ | 1    | Sunday                | Sunday is the first day of the week  |
|   | 2    | Monday                | Monday is the second day of the week |
|   | 3    | Tuesday               | Tuesday is the third day of the week |
|   | 4    | This is a new title 5 | This is a new description 5          |
| * | NULL | NULL                  | NULL                                 |

# לבד בבית

---

- המשיכו עם ה-3 ראוטים החסרים כדי לתרגל: הוספה, הבאת פוסט אחד ומחיקה.
- (יש לכם את כל מה שאתם צריכים בשביל להמשיך)