

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών  
Πανεπιστήμιο Θεσσαλίας, Βόλος



Προχωρημένη Διαχείριση Δεδομένων

Προγραμματιστική Εργασία Report  
Εαρινού Εξαμήνου 2023-24

Καθηγητής: Βασιλακόπουλος Μιχαήλ

---

## E-Bookstore

---

Συγγραφείς:  
Φανή Μπάνου 03322  
Ανδρομάχη Παπαδοπούλου 03314

{ fbanou, andpapadop } @e-ce.uth.gr

## **Περιεχόμενα**

- Στόχος της εργασίας μας
- Παραδοτέα αρχεία
- Οδηγίες εγκατάστασης/Οδηγίες εκκίνησης
- Back-End
- Admin
- Front-End
- Επίλογος
- References



**BOOKSTORE**

Books that inspire you

## 1 Στόχος της εργασίας μας

Η παρούσα εργασία αποσκοπεί στην ανάπτυξη ενός πλήρως λειτουργικού ηλεκτρονικού καταστήματος (e-shop) για την πώληση βιβλίων, στοχεύοντας στη δημιουργία μιας εύχρηστης, ασφαλούς και αποδοτικής πλατφόρμας. Αυτή θα επιτρέπει στους χρήστες να περιηγούνται, να αναζητούν και να αγοράζουν βιβλία με ευκολία. Επιπλέον, η εφαρμογή θα διαθέτει εξελιγμένες δυνατότητες διαχείρισης προϊόντων για τον διαχειριστή του καταστήματος, όπως η προσθήκη, η επεξεργασία, η διαγραφή και η προβολή των προϊόντων. Θα υποστηρίζει επίσης την κατηγοριοποίηση και την επισήμανση προϊόντων ως δημοφιλή, νέα ή εκπτωτικά. Οι δυνατότητες διαχείρισης πελατών θα περιλαμβάνουν εγγραφή και σύνδεση, εξασφαλίζοντας μια ομαλή και ασφαλή εμπειρία αγορών.

## 2 Παραδοτέα αρχεία

Το αρχείο .zip που σας παραδόσαμε περιέχει τις διαφάνειες της παρουσίασης, τα αρχεία της εφαρμογής με τους κώδικες και ένα αναλυτικό report. Το report αυτό περιλαμβάνει οδηγίες εγκατάστασης και λεπτομερή επεξήγηση της δημιουργίας και λειτουργίας του E-bookstore μας. Ο βασικός φάκελος περιέχει υποφακέλους για τη διαχείριση χρηστών και προϊόντων, αρχεία διαμόρφωσης και διάφορα βοηθητικά scripts, όπως φαίνεται στο συνημμένο σχήμα που περιγράφει τη δομή των αρχείων της εφαρμογής.



## 3 Οδηγίες εγκατάστασης/Οδηγίες εκκίνησης

### 3.1 Οδηγίες εγκατάστασης

Για να δημιουργήσουμε το project μας, κατεβάσαμε την **εφαρμογή Node.js** για να έχουμε στη διάθεσή μας το πρωτ και τις σχετικές εντολές που θα δείξουμε παρακάτω. Αρχικά, ανοίγουμε το Visual Studio Code και δημιουργόντες έναν φάκελο με το όνομα "bookstore". Μέσα στον φάκελο "bookstore", δημιουργούμε τρεις υποφακέλους: "frontend", "backend" και "admin".

**Για την ανάπτυξη του frontend:**

1. Ανοίγουμε το VS Code και μεταβαίνουμε στον φάκελο "frontend".
2. Εκτελούμε την εντολή `create-react-app`. για να δημιουργήσουμε το React app.

3. Εγκαθιστούμε το React Router με την εντολή `npm install react-router-dom`.
4. Αναπτύσσουμε τον κώδικα μας.

#### Για την ανάπτυξη του backend:

1. Δημιουργούμε τον φάκελο "backend".
2. Ανοίγουμε το VS Code και μεταβαίνουμε στον φάκελο "backend".
3. Εκτελούμε την εντολή `npm init` για να αρχικοποιήσουμε το project.
4. Εγκαθιστούμε το Express με την εντολή `npm install express`.
5. Εγκαθιστούμε το `jsonwebtoken` με την εντολή `npm install jsonwebtoken`.
6. Εγκαθιστούμε το Mongoose με την εντολή `npm install mongoose`.
7. Εγκαθιστούμε το Multer με την εντολή `npm install multer`.
8. Εγκαθιστούμε το CORS με την εντολή `npm install cors`.
9. Δημιουργούμε το αρχείο `index.js` όπου θα αναπτύξουμε τον κώδικα μου, όπως θα αναφερθεί παρακάτω.

Στη συνέχεια, **κατεβάζουμε την εφαρμογή MongoDB** και δημιουργούμε ένα νέο προφίλ και project. Κατά τη διάρκεια της δημιουργίας του project, λαμβάνουμε ένα connection string το οποίο αντιγράφουμε και θα χρησιμοποιήσουμε στο backend.

#### Για την ανάπτυξη του admin:

1. Δημιουργούμε τον φάκελο "admin".
2. Ανοίγουμε το VS Code και μεταβαίνουμε στον φάκελο "admin".
3. Εκτελούμε την εντολή `create vite@latest .` για να δημιουργήσουμε το Vite app.
4. Εγκαθιστούμε τα απαραίτητα πακέτα με την εντολή `npm install`.
5. Αναπτύσσουμε τον κώδικα μας.

## 3.2 Οδηγίες εκκίνησης

Για να εκκινήσουμε το project "bookstore", ακολουθούμε τα παρακάτω βήματα: ανοίγουμε το Visual Studio Code και τον φάκελο "bookstore". Για κάθε υποφάκελο, ανοίγουμε ένα τερματικό PowerShell. Για να ξεκινήσουμε το frontend, εκτελούμε την εντολή `npm start`, η οποία ανοίγει την ιστοσελίδα στον browser με διεύθυνση <http://localhost:3000/>. Για το backend, χρησιμοποιούμε την εντολή `node .\index.js`. Για να ξεκινήσουμε το admin, εκτελούμε την εντολή `npm run dev` και στη συνέχεια πληκτρολογύμε στον browser τη διεύθυνση <http://localhost:5173/>. Τέλος, ανοίγουμε την εφαρμογή MongoDB και επιλέγουμε "connect" για να συνδεθούμε στη βάση δεδομένων.

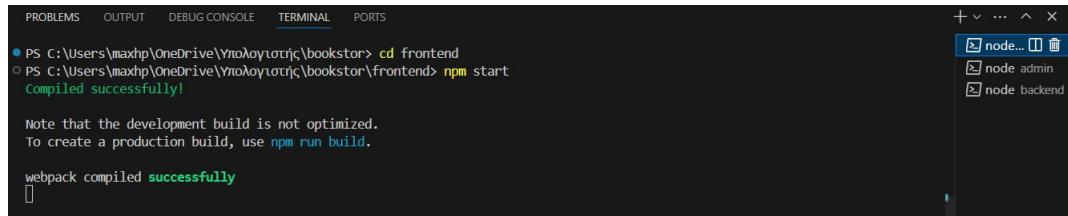
### PowerShell Backend

```
PS C:\Users\maxhp\OneDrive\Υπολογιστής\bookstor\backend> node .\index.js
Server Running on Port 4000
```

### PowerShell Admin

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\maxhp\OneDrive\Υπολογιστής\bookstor\admin> npm run dev
> admin@0.0.0 dev
> vite
VITE v5.2.8 ready in 589 ms
→ Local: http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```

## PowerShell Frontend



```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\maxhp\OneDrive\Υπολογιστής\bookstor> cd frontend
PS C:\Users\maxhp\OneDrive\Υπολογιστής\bookstor\frontend> npm start
Compiled successfully!

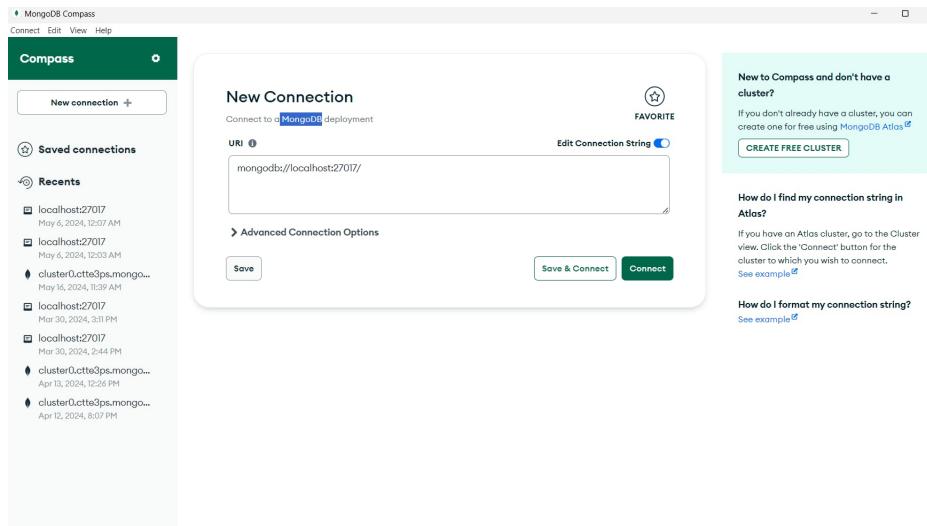
Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully

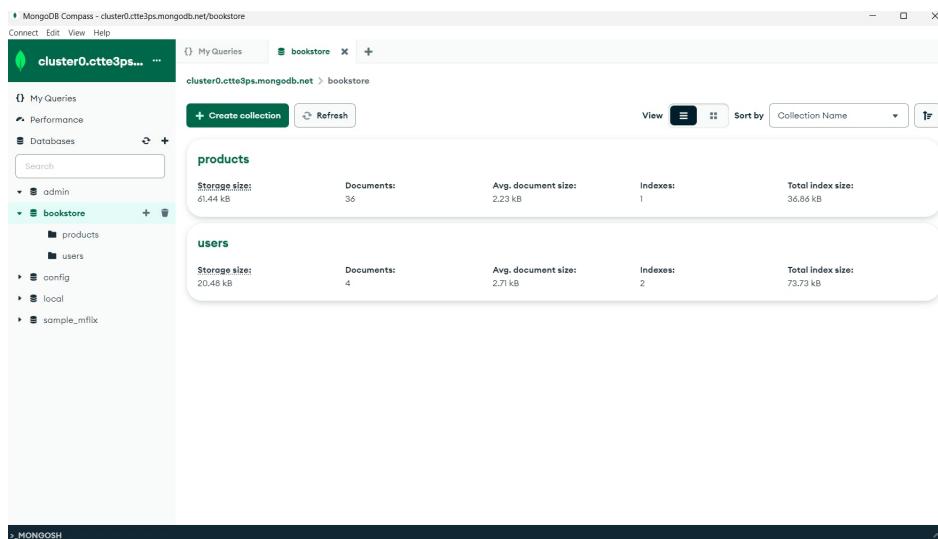
```

## MongoDB

Πατώντας Connect συνδεόμαστε στην βάση δεδομένων μας. Το connection string για την βάση δεδομένων μας είναι "mongodb+srv://maxhpapado:10012002a@cluster0.ctte3ps.mongodb.net/bookstore". Παρακάτω απεικονίζοντα οι πίνακες και οι καταχωρήσεις στην βάση δεδομένων μας:



The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with 'Saved connections' and 'Recents'. In the center, a 'New Connection' dialog is open, prompting for a 'URI' which is set to 'mongodb://localhost:27017/'. Below the URI input is an 'Advanced Connection Options' section. To the right of the dialog, there's a 'How do I find my connection string in Atlas?' section with a 'CREATE FREE CLUSTER' button.



The screenshot shows the MongoDB Compass interface connected to the 'bookstore' database. The left sidebar shows databases like 'admin', 'bookstore', 'config', 'local', and 'sample\_mflix'. The main area displays two collections: 'products' and 'users'. For the 'products' collection, the statistics are: Storage size: 61.44 kB, Documents: 36, Avg. document size: 2.23 kB, Indexes: 1, Total index size: 36.89 kB. For the 'users' collection, the statistics are: Storage size: 20.48 kB, Documents: 4, Avg. document size: 2.71 kB, Indexes: 2, Total index size: 73.73 kB.

The screenshot shows the MongoDB Compass interface connected to a cluster named 'cluster0.ctte3ps'. The 'bookstore' database is selected, and the 'products' collection is displayed. The interface includes a left sidebar for navigating databases and collections, and a main area for viewing documents. Each document is represented by a card showing its ID, name, category, image URL, price details, and a truncated description.

## 4 Back-End

Η ιστοσελίδα μας αναπτύχθηκε χρησιμοποιώντας το περιβάλλον Node.js για τη δημιουργία του back-end. Αξιοποιεί διάφορες βιβλιοθήκες και πρόσθετα για να επιτύχει τις λειτουργικότητες που απαιτούνται για την αποτελεσματική λειτουργία της πλατφόρμας. Ακολουθεί μια αναλυτική περιγραφή των κυριότερων στοιχείων και λειτουργών της εφαρμογής:

### 4.1 Αρχικοποίηση και Εισαγωγή Μονάδων

```

1 const port = 4000;
2 const express = require("express");
3 const app = express();
4 const mongoose = require("mongoose");
5 const jwt = require("jsonwebtoken");
6 const multer = require("multer");
7 const path = require("path");
8 const cors = require("cors");

```

Αρχικα ορίζεται η θύρα στην οποία θα ακούει ο server, ρυθμιζόμενη στο 4000. Στη συνέχεια εισάγονται οι βιβλιοθήκες:

- **Express**: για την κατασκευή server-side εφαρμογών και API.
- **Mongoose**: για τη σύνδεση με τη βάση δεδομένων MongoDB.
- **jsonwebtoken**: για την ασφαλή αυθεντικοποίηση στις εφαρμογές web.
- **multer**: για το ανεβάσμα αρχείων.
- **path**: για την επεξεργασία διαδρομών αρχείων.
- **cors (Cross-Origin Resource Sharing)**: για την επεξεργασία αιτήματων από διαφορετικές πηγές.

## 4.2 Ρύθμιση Middleware

```
1 app.use(express.json());
2 app.use(cors());
```

Το απόσπασμα κώδικα περιγράφει τη ρύθμιση δύο βασικών middleware σε μια εφαρμογή Express, που είναι απαραίτητα για την αποδοτική διαχείριση αιτημάτων και απαντήσεων HTTP:

- **express.json**: επιτρέπει την εύκολη διαχείριση και πρόσβαση σε JSON δεδομένα που έρχονται από τον client και τα προσθέτει στο req.body του αιτήματος.
- **cors**: επιτρέπει στην εφαρμογή να δεχθεί αιτήματα από διάφορες πηγές, κάτι που είναι χρήσιμο για front-end εφαρμογές που φιλοξενούνται σε διαφορετικά domains ή ports.

## 4.3 Σύνδεση με Βάση Δεδομένων

```
1 mongoose.connect("mongodb+srv://bookstore:bookstore@cluster0.sscoti5.mongodb.net/
bookstore");
```

Η εφαρμογή συνδέεται με την βάση δεδομένων MongoDB σε μια cloud υπηρεσία, τη MongoDB Atlas, χρησιμοποιώντας το Mongoose. Η βάση δεδομένων χρησιμοποιείται για την αποθήκευση και την επεξεργασία δεδομένων προϊόντων και χρηστών.

## 4.4 Διαχείριση API

Η εφαρμογή παρέχει διάφορα τελικά σημεία API για τη διαχείριση των αιτήσεων:

- **Βασική σελίδα**: Χρησιμοποιείται για να επιβεβαιωθεί ότι ο server της εφαρμογής λειτουργεί σωστά μετά την εκκίνηση.

```
1 app.get("/", (req, res) => {
2   res.send("Express App is Running")
3 })
```

- **Ανεβάσματα εικόνων**: Αφορά τη ρύθμιση του μηχανισμού αποθήκευσης για ανεβάσματα εικόνων στην ιστοσελίδα με τη χρήση της βιβλιοθήκης multer.

```
1 const storage = multer.diskStorage({
2   destination: './upload/images',
3   filename: (req, file, cb) => {
4     return cb(null, `${file.fieldname}_${Date.now()}${path.extname(file.originalname)})`);
5   }
6 })
7
8 const upload = multer({storage:storage})
```

Η συνάρτηση multer.diskStorage ρυθμίζει τον φάκελο προορισμού (./upload/images) όπου θα αποθηκεύονται τα αρχεία και τον τρόπο ονομασίας των αρχείων, χρησιμοποιώντας το όνομα του πεδίου του αρχείου, ένα χρονικό σφραγίδα για μοναδικότητα και την επέκταση του αρχικού αρχείου. Το αντικείμενο upload που δημιουργείται χρησιμοποιείται στη συνέχεια σε διάφορα σημεία για τη διαχείριση των ανεβασμάτων εικόνων, διασφαλίζοντας έτσι την οργανωμένη και ασφαλή αποθήκευση των αρχείων στον διακομιστή.

```

1 app.use('/images', express.static('upload/images'))
2 app.post("/upload", upload.single('product'), (req, res)=>{
3     res.json({
4         success:1,
5         image_url:'http://localhost:${port}/images/${req.file.filename}',
6     })
7 })

```

Το παραπάνω απόσπασμα κώδικα αναφέρεται στο ανέβασμα εικόνων και τη διαχείρισή τους. Αρχικά, καθορίζεται ο φάκελος upload/images ως τοποθεσία για την πρόσβαση στις εικόνες που έχουν ανέβει μέσω του διακομιστή. Αυτό γίνεται με την express.static, που κάνει τα αρχεία εικόνων διαθέσιμα για πρόσβαση μέσω URL. Στη συνέχεια, το POST endpoint επιτρέπει το ανέβασμα ενός αρχείου εικόνας με τη χρήση του upload.single('product'). Μετά το ανέβασμα, επιστρέφεται ένα JSON αντικείμενο που περιέχει την επιτυχία της διαδικασίας και την διεύθυνση URL της εικόνας, χτισμένη δυναμικά με βάση τη θύρα και τη διεύθυνση του διακομιστή, παρέχοντας έτσι άμεση πρόσβαση στο αρχείο που μόλις ανέβηκε.

- **Διαχείριση προϊόντων:** Περιλαμβάνει την προσθήκη, την κατάργηση και την ανάκτηση προϊόντων.

```

1 const Product = mongoose.model("Product", {
2     id :{
3         type: Number,
4         required: true ,
5     },
6     name:{ 
7         type: String ,
8         required: true ,
9     },
10    category:{ 
11        type: String ,
12        required: true ,
13    },
14    image:{ 
15        type: String ,
16        required: true ,
17    },
18    new_price:{ 
19        type:Number ,
20        required: true ,
21    },
22    old_price:{ 
23        type:Number ,
24        required: true ,
25    },
26    description:{ 
27        type: String ,
28        required: true ,
29    },
30    descriptionbox:{ 
31        type: String ,
32        required: true ,
33    },
34    date:{ 
35        type: Date ,
36        default:Date.now,
37    },
38    available:{ 
39        type:Boolean ,
40        default:true ,
41    },
42 })

```

Το απόσπασμα κώδικα που παρουσιάζεται αναφέρεται στον ορισμό ενός σχήματος για τα προϊόντα μέσω της βιβλιοθήκης Mongoose, που χρησιμοποιείται στην MongoDB. Το σχήμα αυτό καθορίζει τη δομή και τα πεδία

που πρέπει να έχει κάθε εγγραφή προϊόντος στη βάση δεδομένων για την αποτελεσματική διαχείριση και παρουσίαση στον ιστότοπο ηλεκτρονικού εμπορίου. Αναλυτικά:

- **id**: Ένας αριθμός που λειτουργεί ως μοναδικό αναγνωριστικό για κάθε προϊόν, ορίζεται ως υποχρεωτικό.
- **name**: Το όνομα του προϊόντος, επίσης υποχρεωτικό, τύπου string.
- **category**: Η κατηγορία στην οποία ανήκει το προϊόν, υποχρεωτικό πεδίο τύπου string.
- **image**: Το URL της εικόνας του προϊόντος, υποχρεωτικό, τύπου string.
- **new\_price και old\_price**: Οι τιμές του προϊόντος, πριν και μετά από έκπτωση αντίστοιχα, και τα δύο υποχρεωτικά πεδία τύπου number.
- **description**: Μια σύντομη περιγραφή του προϊόντος, υποχρεωτικό πεδίο τύπου string.
- **descriptionbox**: Μια σύντομη περιήγηση του βιβλίου, υποχρεωτικό πεδίο τύπου string.
- **date**: Η ημερομηνία προσθήκης του προϊόντος, με προκαθορισμένη τιμή την τρέχουσα ημερομηνία και ώρα που δημιουργείται η εγγραφή.
- **available**: Μια boolean τιμή που δηλώνει εάν το προϊόν είναι διαθέσιμο, με προκαθορισμένη τιμή true.

```

1 app . post( '/addproduct' , async (req , res )=>{
2     let products = await Product . find ( {} );
3     let id ;
4     if ( products . length >0) {
5         let last_product_array = products . slice ( -1 );
6         let last_product = last_product_array [ 0 ];
7         id = last_product . id +1 ;
8     }
9     else {
10         id =1 ;
11     }
12     const product = new Product ( {
13         id : id ,
14         name : req . body . name ,
15         category : req . body . category ,
16         image : req . body . image ,
17         new_price : req . body . new_price ,
18         old_price : req . body . old_price ,
19         description : req . body . description ,
20         descriptionbox : req . body . descriptionbox ,
21     });
22     console . log ( product );
23     await product . save ();
24     console . log ( " Saved " );
25     res . json ( {
26         success : true ,
27         name : req . body . name ,
28     })
29 })
```

Το παραπάνω απόσπασμα κώδικα αφορά την προσθήκη νέων προϊόντων σε μια βάση δεδομένων. Αρχικά, αναζητά όλα τα υπάρχοντα προϊόντα στη βάση δεδομένων. Αν υπάρχουν ήδη προϊόντα, υπολογίζει τον νέο μοναδικό id αυξάνοντας κατά ένα το id του τελευταίου προϊόντος. Αν δεν υπάρχουν προϊόντα, ορίζει το id ως 1. Στη συνέχεια, δημιουργεί ένα νέο προϊόν με τα δεδομένα που λαμβάνει από το αίτημα (req.body), όπως name, category, image, new\_price, old\_price, description, και descriptionbox. Το προϊόν αυτό αποθηκεύεται στη βάση δεδομένων και επιστρέφει ένα αντικείμενο JSON που επιβεβαιώνει την επιτυχία της διαδικασίας και το όνομα του προϊόντος που προστέθηκε. Αυτός ο μηχανισμός εξασφαλίζει ότι κάθε προϊόν που προστίθεται έχει μοναδικό id, αποφεύγοντας συγκρούσεις και διασφαλίζοντας την ορθή οργάνωση των δεδομένων της βάσης.

```

1 app . post( '/removeproduct' , async (req , res ) => {
2     await Product . findOneAndDelete ( { id : req . body . id } );
3     console . log ( " Removed " );
4     res . json ( {
5         success : true ,
6         name : req . body . name
7     })
8 })
```

Το απόσπασμα κώδικα αφορά τη διαγραφή προϊόντων από μια βάση δεδομένων. Χρησιμοποιώντας τη μέθοδο `findOneAndDelete()`, αναζητά και διαγράφει το προϊόν με βάση το id που προωθείται μέσω του αιτήματος (`req.body.id`). Μετά την επιτυχή διαγραφή, καταγράφει ένα μήνυμα στην κονσόλα και επιστρέφει μια απόκριση JSON που επιβεβαιώνει την επιτυχία της διαδικασίας και παρέχει το όνομα του διαγραμμένου προϊόντος, βοηθώντας έτσι την ενημέρωση του χρήστη για την επιτυχία ή αποτυχία της λειτουργίας διαγραφής.

```
1 app.get('/allproducts', async (req, res) => {
2   let products = await Product.find({}); 
3   console.log("All Products Fetched");
4   res.send(products);
5 })
```

Το απόσπασμα κώδικα αωαφέρεται στην ανάκτηση όλων των προϊόντων από μια βάση δεδομένων. Χρησιμοποιώντας τη μέθοδο `find()`, αναζητά όλες τις εγγραφές προϊόντων στη βάση δεδομένων, ουσιαστικά επιστρέφοντας κάθε διαθέσιμο προϊόν. Μετά την ανάκτηση των προϊόντων, εμφανίζει ένα μήνυμα στην κονσόλα ("All Products Fetched") για επιβεβαίωση της διαδικασίας και στέλνει τα προϊόντα πίσω στον client μέσω της απόκρισης `res.send(products)`. Αυτό είναι κρίσιμο για την εφαρμογή καθώς παρέχει τη λειτουργία προβολής του καταλόγου προϊόντων, επιτρέποντας την εύκολη πρόσβαση και διαχείριση τους.

- Αυθεντικοποίηση χρήστη:** Περιλαμβάνει εγγραφή και σύνδεση χρηστών, με τη δημιουργία και επαλήθευση JWTs.

```
1 const Users = mongoose.model('Users', {
2   name: {
3     type: String,
4   },
5   email: {
6     type: String,
7     unique: true
8   },
9   password: {
10    type: String
11  },
12   cartData: {
13    type: Object,
14  },
15   date: {
16    type: Date,
17    default: Date.now,
18  }
19 })
```

Το απόσπασμα κώδικα ορίζει ένα μοντέλο δεδομένων για τους χρήστες στη MongoDB χρησιμοποιώντας το Mongoose. Το μοντέλο `Users` περιλαμβάνει τα εξής πεδία:

- **name:** Ένα πεδίο τύπου String για το όνομα του χρήστη.
- **email:** Ένα πεδίο τύπου String για το email του χρήστη, το οποίο ορίζεται ως μοναδικό, εξασφαλίζοντας ότι δεν μπορούν να υπάρχουν δύο χρήστες με το ίδιο email.
- **password:** Ένα πεδίο τύπου String για τον κωδικό πρόσβασης του χρήστη.
- **cartData:** Ένα πεδίο τύπου Object, που μπορεί να αποθηκεύει δομημένα δεδομένα, όπως τα περιεχόμενα του καλαθιού αγορών του χρήστη.
- **date:** Ένα πεδίο τύπου Date, το οποίο έχει ως προκαθορισμένη τιμή την τρέχουσα ημερομηνία και ώρα, δηλαδή τη στιγμή δημιουργίας του.

Το μοντέλο αυτό είναι θεμελιώδες για τη διαχείριση των χρηστών στην εφαρμογή, παρέχοντας τη βάση για λειτουργίες όπως η εγγραφή, η σύνδεση και η διαχείριση του καλαθιού αγορών.

```
1 app.post('/signup', async (req, res) => {
2   let check = await Users.findOne({email: req.body.email});
3   if (check) {
4     return res.status(400).json({success: false, errors: "existing user found
5     with same email address"})
```

```

6     }
7
8     let cart = {};
9     for (let index = 0; index < 300; index++) {
10        cart[index] = 0;
11    }
12
13    const user = new Users ({
14        name:req.body.username,
15        email:req.body.email,
16        password:req.body.password,
17        cartData:cart,
18    })
19
20    await user.save();
21
22    const data = {
23        user:{
24            id:user.id
25        }
26    }
27
28    const token = jwt.sign(data , 'secret_ecom');
29    res.json({success:true , token})
30
31 })

```

Το απόσπασμα κώδικα που παρουσιάζεται αφορά την εγγραφή νέων χρηστών. Η διαδικασία εγγραφής ξεκινά με τον έλεγχο της μοναδικότητας του email του χρήστη: αν το email υπάρχει ήδη στη βάση δεδομένων, επιστρέφει ένα σφάλμα 400 και ενημερώνει τον χρήστη ότι το email είναι ήδη καταχωρημένο. Αν το email δεν υπάρχει ήδη, προχωρά στη δημιουργία ενός νέου αντικειμένου χρήστη, περιλαμβάνοντας το όνομα χρήστη, το email, τον κωδικό πρόσβασης και ένα αρχικό καλάθι αγορών (αντικείμενο με 300 θέσεις, όλες αρχικά μηδενισμένες). Μετά την αποθήκευση του νέου χρήστη στη βάση δεδομένων, δημιουργείται ένα JWT (JSON Web Token) που περιέχει το ID του χρήστη, το οποίο μπορεί να χρησιμοποιηθεί για την ταυτοποίηση του χρήστη σε μελλοντικές αιτήσεις προς το API. Τέλος, επιστρέφει το token και μια επιτυχή απόκριση στον χρήστη, ενημερώνοντας τον για την επιτυχή εγγραφή του στην πλατφόρμα.

```

1 app.post('/login' , async (req ,res) => {
2     let user = await Users.findOne({email:req.body.email});
3     if (user) {
4         const passCompare = req.body.password === user.password;
5         if (passCompare) {
6             const data = {
7                 user:{
8                     id:user.id
9                 }
10            }
11            const token = jwt.sign(data , 'secret_ecom');
12            res.json({success:true ,token});
13        }
14        else {
15            res.json({success:false ,errors:"Wrong Password"});
16        }
17    }
18    else {
19        res.json({success:false ,errors:"Wrong Email Id"})
20    }
21 })

```

Το απόσπασμα κώδικα αναφέρεται στη είσοδο χρηστών σε μια εφαρμογή που χρησιμοποιεί MongoDB. Αρχικά, αναζητά στη βάση δεδομένων έναν χρήστη με το email που παρέχεται μέσω του αιτήματος (req.body.email). Αν βρεθεί ο χρήστης, συγκρίνει τον παρεχόμενο κωδικό πρόσβασης με αυτόν που είναι αποθηκευμένος στη βάση δεδομένων. Εάν ο κωδικός είναι σωστός, δημιουργείται ένα JWT (JSON Web Token) που περιέχει το

ID του χρήστη, το οποίο μπορεί να χρησιμοποιηθεί για να ταυτοποιηθούν μελλοντικά αιτήματα του χρήστη. Το token επιστρέφεται στον χρήστη μαζί με μια επιβεβαίωση επιτυχίας. Αν ο κωδικός είναι λανθασμένος ή το email δεν βρεθεί, επιστρέφεται ένα μήνυμα λάθους, ενημερώνοντας τον χρήστη για το πρόβλημα.

- **Διαχείριση καλαθιού:** Επιτρέπει την προσθήκη και αφαίρεση προϊόντων στο καλάθι αγορών, καθώς και την ανάκτηση των δεδομένων του καλαθιού.

```

1 app . post( '/ addtocart' , fetchUser , async ( req , res ) => {
2     console . log ( "added" , req . body . itemId );
3     let userData = await Users . findOne ( { _id : req . user . id } );
4     userData . cartData [ req . body . itemId ] += 1 ;
5     await Users . findOneAndUpdate ( { _id : req . user . id } , { cartData : userData . cartData } )
6     ;
7     res . send ( "Added" )
8 })

```

Το απόσπασμα κώδικα αναφέρεται στην προσθήκη προϊόντων στο καλάθι αγορών ενός χρήστη. Αρχικά, χρησιμοποιεί το middleware fetchUser για να αυθεντικοποιήσει τον χρήστη και να προσδιορίσει το id του. Στη συνέχεια, αναζητά τα στοιχεία του χρήστη με βάση το id και αυξάνει την ποσότητα του προϊόντος, που προσδιορίζεται από το itemId, στο cartData του χρήστη. Αφού ενημερώσει το cartData στη βάση δεδομένων, επιβεβιώνει την προσθήκη με ένα μήνυμα "Added". Αυτή η λειτουργία επιτρέπει στους χρήστες να διαχειρίζονται δυναμικά τα περιεχόμενα του καλαθιού τους μέσα από την εφαρμογή.

```

1 app . post( '/ removefromcart' , fetchUser , async ( req , res ) => {
2     console . log ( "removed" , req . body . itemId );
3     let userData = await Users . findOne ( { _id : req . user . id } );
4     if ( userData . cartData [ req . body . itemId ] > 0 )
5         userData . cartData [ req . body . itemId ] -= 1 ;
6     await Users . findOneAndUpdate ( { _id : req . user . id } , { cartData : userData . cartData } )
7     ;
8     res . send ( "Removed" )
9 })

```

Το απόσπασμα κώδικα ορίζει την αφαίρεση προϊόντων από το καλάθι αγορών ενός χρήστη. Χρησιμοποιείται το middleware fetchUser για να εξακριβώσει και να προσδιορίσει τον ταυτοποιημένο χρήστη μέσω του id που παρέχεται στο αίτημα. Αφού ανακτήσει τα δεδομένα του χρήστη από τη βάση δεδομένων με το id, ελέγχει αν η ποσότητα του συγκεκριμένου προϊόντος (που προσδιορίζεται από το itemId του αιτήματος) στο cartData είναι μεγαλύτερη από μηδέν. Αν ναι, μειώνει την ποσότητα κατά ένα. Στη συνέχεια, ενημερώνει τα δεδομένα του καλαθιού στη βάση δεδομένων και επιστρέφει μήνυμα "Removed" ως επιβεβαίωση της ενέργειας. Έτσι διευκολύνει τη δυναμική διαχείριση των περιεχομένων του καλαθιού αγορών, επιτρέποντας στους χρήστες να αφαιρούν προϊόντα αποτελεσματικά.

```

1 app . post( '/ getcart' , fetchUser , async ( req , res ) => {
2     console . log ( "GetCart" );
3     let userData = await Users . findOne ( { _id : req . user . id } );
4     res . json ( userData . cartData );
5 })

```

Το απόσπασμα κώδικα περιγράφει την ανάκτηση των δεδομένων του καλαθιού αγορών ενός χρήστη. Χρησιμοποιείται το middleware fetchUser, το οποίο λειτουργεί ως μηχανισμός αυθεντικοποίησης για να επιβεβαιώσει την ταυτότητα του χρήστη και να διασφαλίσει ότι έχει πρόσβαση μόνο στα δικά του δεδομένα. Μετά την επιτυχή ταυτοποίηση, ανακτά τα δεδομένα του χρήστη από τη βάση δεδομένων MongoDB με βάση το id που αντιστοιχεί στον ταυτοποιημένο χρήστη. Το κύριο ενδιαφέρον είναι το cartData, το οποίο περιέχει τις πληροφορίες των προϊόντων στο καλάθι αγορών του χρήστη. Αφού ανακτηθούν τα δεδομένα, επιστρέφονται στον χρήστη σε μορφή JSON. Αυτή η λειτουργικότητα είναι ζωτικής σημασίας, καθώς επιτρέπει στους χρήστες να έχουν άμεση πρόσβαση στις τρέχουσες επιλογές τους, επιτρέποντας την εύκολη τροποποίηση και την τελική επιβεβαίωση των αγορών τους.

- **Ανάκτηση πρόσφατων προϊόντων**

```

1 app . get( '/ newcollections' , async ( req , res ) => {
2     let products = await Product . find ( {} );

```

```

3     let newcollection = products.slice(0).slice(-8);
4     console.log("NewCollection Fetched");
5     res.send(newcollection);
6   })

```

Το παραπάνω απόσπασμα κώδικα αναφέρεται στην ανάκτηση των πιο πρόσφατων προϊόντων. Χρησιμοποιεί τη μέθοδο Product.find() για να ανακτήσει όλα τα προϊόντα από τη βάση δεδομένων και επιλέγει τα τελευταία οκτώ με τη μέθοδο slice(-8). Αυτό το επιλεγμένο υποσύνολο αποτελεί τη νέα συλλογή προϊόντων που έχουν προστεθεί πρόσφατα στην πλατφόρμα. Καταγράφεται ένα μήνυμα στην κονσόλα για την επιβεβαίωση της επιτυχούς ανάκτησης και τα δεδομένα στέλνονται στον πελάτη μέσω res.send(newcollection), προσφέροντας έτσι μια ενημερωμένη προβολή των νέων προσθηκών στην εφαρμογή.

#### - Ανάκτηση δημοφιλών προϊόντων

```

1 app.get('/popularinmotivational', async (req, res) => {
2   let products = await Product.find({category:"motivational"});
3   let popular_in_motivational = products.slice(0,4);
4   console.log("Popular in motivational fetched");
5   res.send(popular_in_motivational);
6 })

```

Το απόσπασμα κώδικα περιγράφει την ανάκτηση των πιο δημοφιλών προϊόντων στην κατηγορία "motivational". Χρησιμοποιείται η συνάρτηση Product.find(category:"motivational") για να φιλτράρει τα προϊόντα ανάλογα με την κατηγορία τους. Μετά την ανάκτηση των προϊόντων, το πρόγραμμα επιλέγει τα τέσσερα πρώτα προϊόντα μέσω της μεθόδου slice(0,4), τα οποία είναι τα πιο δημοφιλή στην κατηγορία αυτή. Στη συνέχεια, καταγράφει ένα μήνυμα στην κονσόλα για την επιτυχή ανάκτηση των δημοφιλών προϊόντων και επιστρέφει αυτά τα προϊόντα στον πελάτη με την res.send(popular\_in\_motivational).

## 4.5 Middleware και Ασφάλεια

```

1 const fetchUser = async (req, res, next) => {
2   const token = req.header('auth-token');
3   if (!token) {
4     res.status(401).send({errors:"Please authenticate using valid token"})
5   }
6   else {
7     try {
8       const data = jwt.verify(token, 'secret_ecom');
9       req.user = data.user;
10      next();
11    } catch (error) {
12      res.status(401).send({errors:"please authenticate using valid token"})
13    }
14  }
15}

```

Το απόσπασμα κώδικα χρησιμοποιείται για την επαλήθευση της ταυτότητας των χρηστών μέσω JWT (JSON Web Tokens). Αρχικά, ελέγχει για την ύπαρξη ενός token στα headers του εισερχόμενου αιτήματος και προσπαθεί να το επαληθεύσει χρησιμοποιώντας τη συνάρτηση jwt.verify(). Αν το token είναι έγκυρο, αποθηκεύει τα δεδομένα του χρήστη στο req.user για περαιτέρω χρήση. Αν το token δεν είναι έγκυρο ή λείπει, επιστρέφει σφάλμα 401, ζητώντας έγκυρη πιστοποίηση. Αυτή η λειτουργία ενισχύει την ασφάλεια, εξασφαλίζοντας ότι μόνο εξουσιοδοτημένοι χρήστες προσπελαύνουν προστατευμένα σημεία της εφαρμογής.

## 4.6 Εκκίνηση του Server

```

1 app.listen(port,(error) => {
2   if (!error) {
3     console.log("Server Running on Port "+port);
4   }
5   else {
6     console.log("Error : " + error);
7   }
8 })

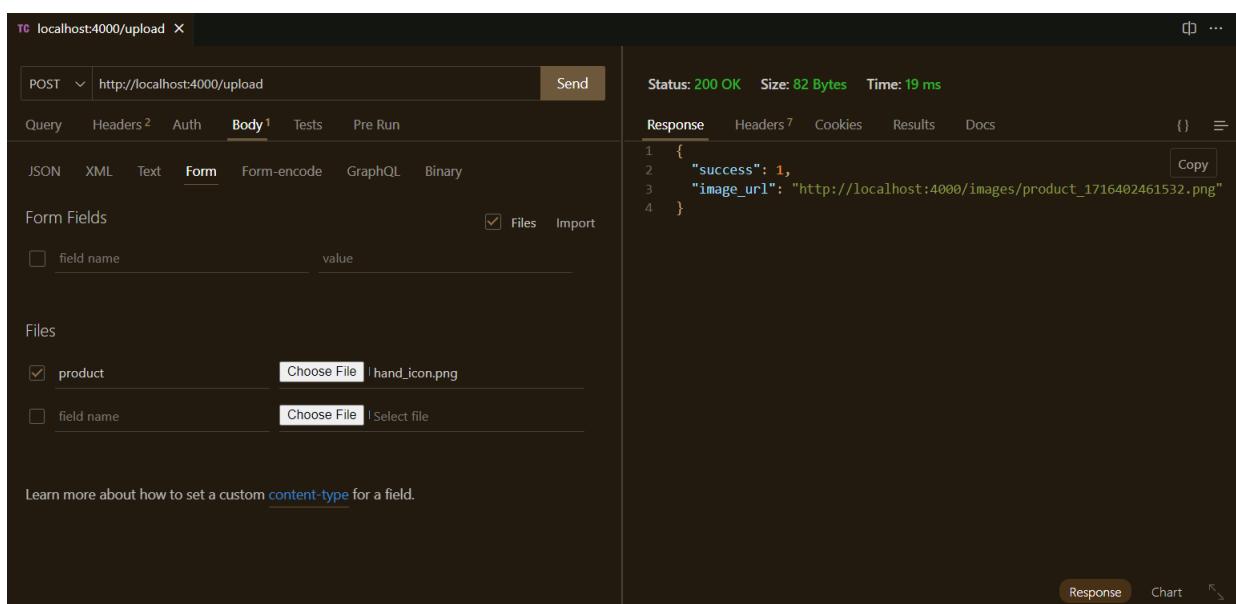
```

Αυτός ο κώδικας ρυθμίζει τον server να ακούει για εισερχόμενα αιτήματα στη θύρα που έχει οριστεί (θύρα 4000) και παρέχει μια callback συνάρτηση που εκτελείται κατά την εκκίνηση του server. Η διαδικασία εκκίνησης ελέγχεται για τυχόν σφάλματα με τη χρήση της μεταβλητής error στη callback συνάρτηση:

- Αν δεν υπάρχει σφάλμα:** Εμφανίζεται ένα μήνυμα στην κονσόλα που επιβεβαιώνει ότι ο server λειτουργεί και σε ποια θύρα. Αυτό μας βοηθά να γνωρίζουμε ότι ο server έχει ξεκινήσει επιτυχώς και είναι έτοιμος να δεχθεί αιτήματα από τους χρήστες.
- Αν υπάρχει σφάλμα:** Εμφανίζεται ένα μήνυμα σφάλματος με τη λεπτομέρεια του σφάλματος που προέκυψε. Αυτό είναι χρήσιμο για την αποσφαλμάτωση, καθώς μας δείχνει την ακριβή αιτία της αποτυχίας της εκκίνησης του server.

## 4.7 Δοκιμή

To Thunder Client είναι ένα εργαλείο δοκιμής API που λειτουργεί ως επέκταση του Visual Studio Code (VS Code). Αξιοποιήσαμε αυτή την επέκταση για να δοκιμάσουμε διεξοδικά τις λειτουργίες της ιστοσελίδας μας, επιβεβαιώνοντας ότι κάθε λειτουργία εκτελείται όπως αναμένεται. Παρακάτω παρουσιάζεται αναλυτικά το υλικό από κάθε δοκιμή που πραγματοποιήσαμε, προσφέροντας μια σαφή εικόνα της λειτουργικότητας και της απόδοσης της ιστοσελίδας μας.



Εικ. 1. Δοκιμή ανέβασμα εικόνας

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:4000/addproduct`. The response status is **200 OK**, size is **39 Bytes**, and time is **310 ms**. The response body is:

```

1 {
2   "success": true,
3   "name": "Can't Hurt Me"
4 }

```

**Εικ. 2.** Δοκιμή προσθήκης προϊόντος

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:4000/removeproduct`. The response status is **200 OK**, size is **39 Bytes**, and time is **87 ms**. The response body is:

```

1 {
2   "success": true,
3   "name": "Can't Hurt Me"
4 }

```

**Εικ. 3.** Δοκιμή αφαίρεσης προϊόντος

```

Status: 200 OK  Size: 79.45 KB  Time: 232 ms

Response Headers 7 Cookies Results Docs
1 [
2   {
3     "_id": "6645c95b1b8c4551c36060a8",
4     "id": 1,
5     "name": "Ένα τίποτα μπορεί να αλλάξει τα πάντα",
6     "category": "motivational",
7     "image": "http://localhost:4000/images/product_1715849562969.avif",
8     "new_price": 12.99,
9     "old_price": 18.3,
10    "description": "Συγγραφέας Clear James Εκδότης Ιβίσκος Αριθμός Σελίδων 432",
11    "descriptionbox": "Το διεθνές bestseller Atomic Habits που βρέθηκε αμέσως στην κορυφή και παραμένει εκεί -έχοντας ξεπεράσει το 1.000.000 αντίτυπα παγκοσμίως- τώρα στα Ελληνικά! Μικρά βήματα - Μεγάλες Αλλαγές Όποιοι κι αν είναι οι στόχοι μας, το βήμα αυτό μας αποκαλύπτει σίγουρας μεθόδους που θα μας απογειώσουν! Παραθέτοντας, έγκυρα πορίσματα της βιολογίας, της ψυχολογίας και της νευροεπιστήμης, ο James Clear, ένας από τους διεθνώς κορυφαίους ειδικούς, μας μαθαίνει με ακρίβεια τις κατάλληλες στρατηγικές για να αποκτήσουμε καλές συνήθειες, να ξεφορτωθούμε τις κακές και να κινητοποιήσουμε στη συμπεριφορά μας"
  Response Chart ↗

```

**Εικ. 4.** Δοκιμή προβολής όλων των προϊόντων

```

Status: 200 OK  Size: 9.81 KB  Time: 140 ms

Response Headers 7 Cookies Results Docs
1 [
2   {
3     "_id": "6645c95b1b8c4551c36060a8",
4     "id": 1,
5     "name": "Ένα τίποτα μπορεί να αλλάξει τα πάντα",
6     "category": "motivational",
7     "image": "http://localhost:4000/images/product_1715849562969.avif",
8     ,
9     "new_price": 12.99,
10    "old_price": 18.3,
11    "description": "Συγγραφέας Clear James Εκδότης Ιβίσκος Αριθμός Σελίδων 432",
12    "descriptionbox": "Το διεθνές bestseller Atomic Habits που βρέθηκε αμέσως στην κορυφή και παραμένει εκεί -έχοντας ξεπεράσει το 1.000.000 αντίτυπα παγκοσμίως- τώρα στα Ελληνικά! Μικρά βήματα - Μεγάλες Αλλαγές Όποιοι κι αν είναι οι στόχοι μας, το βήμα αυτό μας αποκαλύπτει σίγουρας μεθόδους που θα μας απογειώσουν! Παραθέτοντας, έγκυρα πορίσματα της βιολογίας, της ψυχολογίας και της νευροεπιστήμης, ο James Clear, ένας από τους διεθνώς κορυφαίους ειδικούς, μας μαθαίνει με ακρίβεια τις κατάλληλες στρατηγικές για να αποκτήσουμε καλές συνήθειες, να ξεφορτωθούμε τις κακές και να κινητοποιήσουμε στη συμπεριφορά μας, ώπτε να οδηγηθούμε σε αξιοθαύμαστα αποτελέσματα. Επιπλέον."
  Response Chart ↗

```

**Εικ. 5.** Δοκιμή προβολής δημοφιλών προϊόντων

```

Status: 200 OK  Size: 12.67 KB  Time: 215 ms

Response Headers 7 Cookies Results Docs
1 [
2   {
3     "_id": "6645d7601b8c4551c3606189",
4     "id": 30,
5     "name": "AQA A level computer science",
6     "category": "ece",
7     "image": "http://localhost:4000/images/product_1715853152610
8       .avif",
9     "new_price": 69.99,
10    "old_price": 78.99,
11    "description": "Συγγραφέας Bob Reeves Εκδότης Hodder Education
      Αριθμός Σελίδων 448",
12    "descriptionbox": "Exam Board: AQA Level: AS/A-level Subject:
      Computer Science First Teaching: September 2015 First Exam:
      June 2016 This title has been approved by AQA for use with
      the AS and A-level AQA Computer Science specifications. AQA
      A-level Computer Science gives students the chance to think
      creatively and progress through the AQA AS and A-level
      Computer Science specifications. Detailed coverage of the fundamental
      principles of computing, whilst a range of activities help to
      develop the programming skills and computational thinking
      skills at A-level and beyond. - Enables students to build a
      Response Chart ↗"

```

Εικ. 6. Δοκιμή προβολής νέων προϊόντων

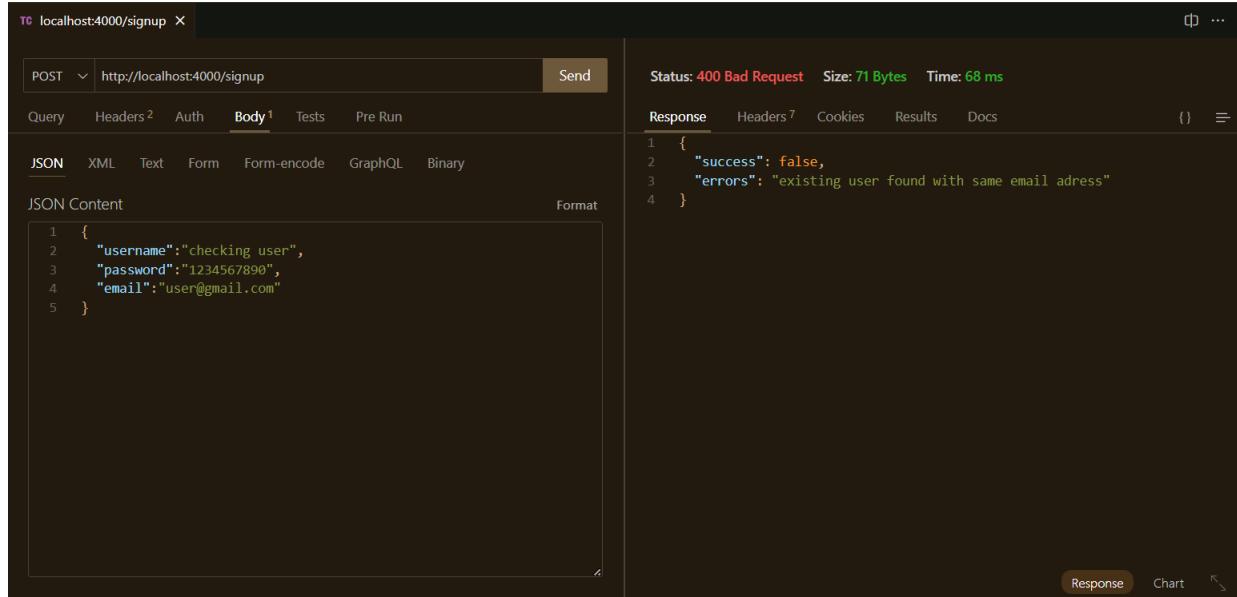
```

Status: 200 OK  Size: 187 Bytes  Time: 181 ms

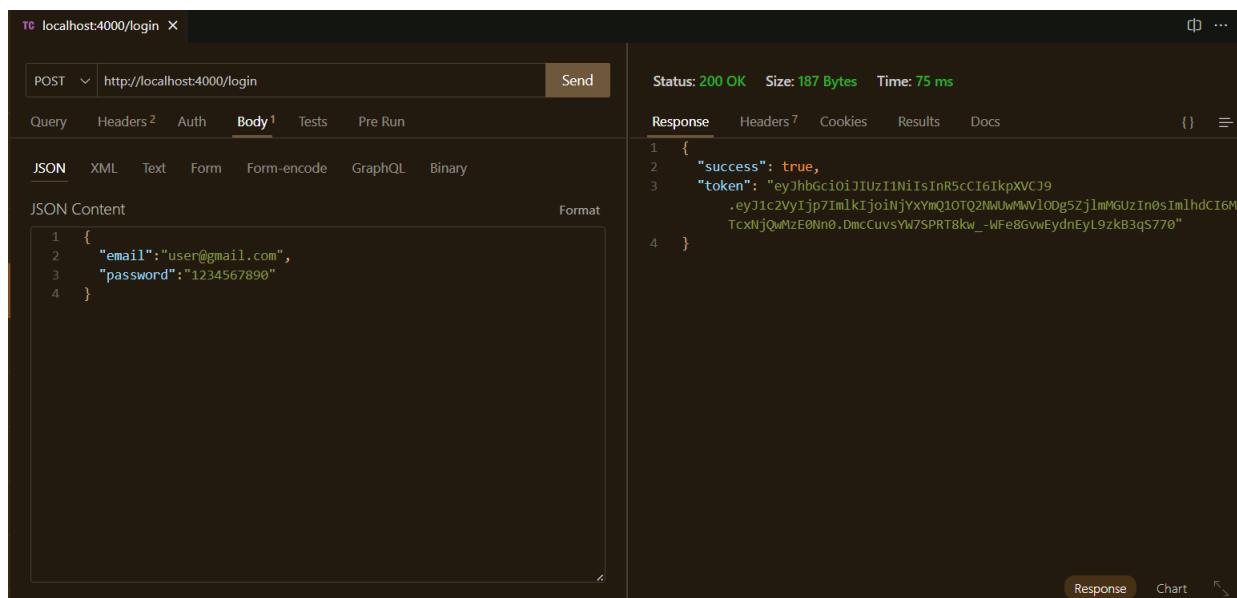
Response Headers 7 Cookies Results Docs
1 {
2   "success": true,
3   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
4     .eyJpc3VlYjIpbTImkIjoiNjY0ZTNiMzB1NjNiMmViMGYXZi5MjE3InosImh0dCI6M
      TcxNjQwMjk5Mn0.SMdTeMU068mwvjmDzgXBSAYZFszPlrqn8loI_5dybk"

```

Εικ. 7. Δοκιμή εγγραφής νέου χρήστη



**Εικ. 8.** Δοκιμή εγγραφής χρήστη με υπάρχων email



**Εικ. 9.** Δοκιμή σύνδεσης χρήστη

The screenshot shows a POST request to `http://localhost:4000/login`. The request body is JSON with fields `email` and `password`. The response status is 200 OK, size is 43 bytes, and time is 70 ms. The response body is:

```

1 {
2   "success": false,
3   "errors": "Wrong Email Id"
4 }

```

**Εικ. 10.** Δοκιμή σύνδεσης χρήστη με λάθος email

The screenshot shows a POST request to `http://localhost:4000/login`. The request body is JSON with fields `email` and `password`. The response status is 200 OK, size is 43 bytes, and time is 74 ms. The response body is:

```

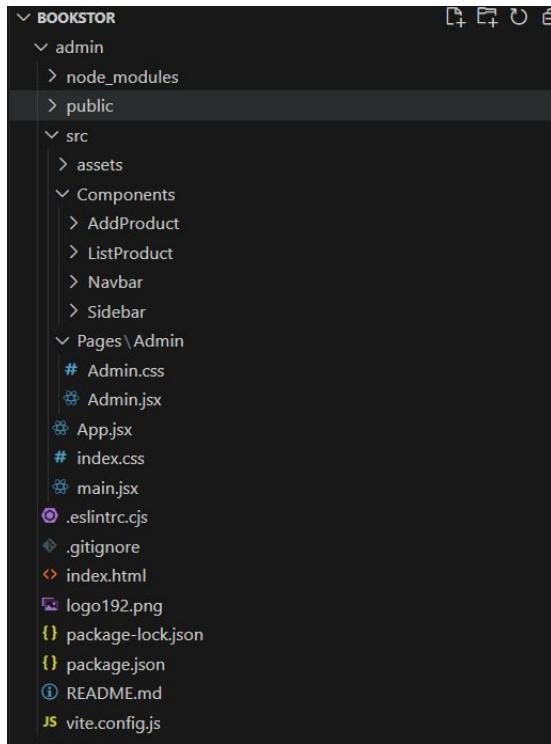
1 {
2   "success": false,
3   "errors": "Wrong Password"
4 }

```

**Εικ. 11.** Δοκιμή σύνδεσης χρήστη με λάθος password

## 5 Admin

Ας περάσουμε στον φάκελο και στο κομμάτι του **Admin**. Όπως βλέπουμε παρακάτω στην φωτογραφία, υπάρχουν τρεις μεγάλοι φάκελοι. Εμείς θα ασχοληθούμε με τον φάκελο ‘src’, ο οποίος περιέχει τον κώδικα που αναπτύξαμε για την εργασία μας. Ο φάκελος ‘src’ περιέχει άλλους τρεις υποφακέλους: ‘Assets’, ‘Components’, ‘Pages’ και τρία αρχεία: ‘App.jsx’, ‘index.css’, ‘main.jsx’, τα οποία θα δείξουμε και θα αναλύσουμε παρακάτω. Κάθε φάκελος που θα αναφέρουμε περιέχει δύο αρχεία: ένα .jsx(αρχείο JavaScript που χρησιμοποιείται στο React για να ορίζει τη δομή, τη λογική και τη λειτουργία των UI components μέσω του HTML-like syntax (JSX)) και ένα .css, που αφορά το styling.



### 5.1 First Folder Assets

Περιέχει όλες τις φωρογραφίες που χρησιμοποιήθηκαν στο site μας.

### 5.2 Second Folder Components

#### 1 AddProduct

Ο κώδικας δημιουργεί ένα React component με το όνομα ‘AddProduct’, το οποίο παρέχει μια διεπαφή για την προσθήκη νέων προϊόντων σε ένα κατάστημα. Το component χρησιμοποιεί το ‘useState’ για να διαχειριστεί την κατάσταση των λεπτομερειών του προϊόντος (‘productDetails’) και της εικόνας του προϊόντος (‘image’). Περιλαμβάνει πεδία εισαγωγής για τον τίτλο του προϊόντος, την τιμή, την προσφορά, την περιγραφή, την περιγραφή κουτιού και την κατηγορία του προϊόντος. Υπάρχει επίσης μια περιοχή για την ανέβασμα εικόνας, που επιτρέπει στους χρήστες να επιλέξουν μια εικόνα για το προϊόν. Οι συναρτήσεις ‘imageHandler’ και ‘changeHandler’ ενημερώνουν την κατάσταση όταν ο χρήστης επιλέγει μια εικόνα ή αλλάζει τα πεδία εισαγωγής αντίστοιχα. Η συνάρτηση ‘Add\_Product’ χειρίζεται την υποβολή των δεδομένων, αποστέλλοντας πρώτα την εικόνα στον διακομιστή και μετά ενημερώνοντας τα στοιχεία του προϊόντος με τη διεύθυνση URL της εικόνας πριν αποστέλλει το ίδιο το προϊόν στον διακομιστή. Τέλος, αποδίδεται ένα κουμπί που, όταν πατηθεί, καλεί τη συνάρτηση ‘Add\_Product’ για να προσθέσει το προϊόν.

## 2 ListProduct

To ‘ListProduct’ component είναι ένα React component που εμφανίζει μια λίστα με όλα τα προϊόντα του καταστήματος και παρέχει τη δυνατότητα διαγραφής προϊόντων. Χρησιμοποιεί το ‘useState’ για να διαχειρίζεται την κατάσταση των προϊόντων και το ‘useEffect’ για να φορτώνει τα δεδομένα των προϊόντων κατά την αρχική φόρτωση της σελίδας μέσω της συνάρτησης ‘fetchInfo’. Η συνάρτηση ‘remove\_product’ στέλνει ένα αίτημα POST στο API για να διαγράψει ένα προϊόν και ανανεώνει τη λίστα των προϊόντων. Το component αποδίδει τη λίστα των προϊόντων, όπου κάθε προϊόν εμφανίζεται με την εικόνα του, το όνομα, την παλιά και νέα τιμή, την κατηγορία και ένα εικονίδιο διαγραφής, επιτρέποντας τη διαγραφή του προϊόντος με ένα κλικ.

## 3 Navbar

To ‘Navbar’ component είναι ένα React component που δημιουργεί μια απλή γραμμή πλοήγησης περιλαμβάνοντας ένα λογότυπο και μια εικόνα προφίλ. Χρησιμοποιεί το ‘useState’ για να εισάγει το React και το CSS αρχείο για το styling, καθώς και τις εικόνες ‘navLogo’ και ‘navProfile’ για το λογότυπο και την εικόνα προφίλ αντίστοιχα.

## 4 Sidebar

To ‘Sidebar’ component είναι ένα React component που δημιουργεί μια πλαϊνή μπάρα πλοήγησης για μια εφαρμογή διαχείρισης προϊόντων, περιλαμβάνοντας συνδέσμους για την προσθήκη νέων προϊόντων και την προβολή της λίστας προϊόντων. Χρησιμοποιεί το ‘Link’ από το ‘react-router-dom’ για την πλοήγηση και εισάγει εικόνες ως εικονίδια για τους συνδέσμους. Περιέχει μια εικόνα και ένα κείμενο για τους συνδέσμους ”Add Product” και ”Product List”, προσφέροντας μια εύχρηστη πλοήγηση μέσα στην εφαρμογή.

## 5.3 Third Folder Pages

### 1 Admin

To ‘Admin’ component είναι ένα React component που παρέχει μια διεπαφή διαχείρισης για προϊόντα, περιλαμβάνοντας ένα sidebar για πλοήγηση και χρησιμοποιώντας το ‘react-router-dom’ για τη διαχείριση των διαδρομών. Περιέχει δύο κύριες διαδρομές: μία για την προσθήκη προϊόντων μέσω του ‘AddProduct’ component και μία για την προβολή της λίστας προϊόντων μέσω του ‘ListProduct’ component, επιτρέποντας έτσι στον διαχειριστή να προσθέτει και να διαχειρίζεται προϊόντα από ένα κεντρικό σημείο.

## 5.4 Files

### 1 App.jsx

To ‘App’ component είναι το κύριο component μιας React εφαρμογής, το οποίο συνδυάζει τη γραμμή πλοήγησης και τη σελίδα διαχείρισης. Περιέχει ένα ‘div’ που αποδίδει δύο components: το ‘Navbar’, που λειτουργεί ως η γραμμή πλοήγησης της εφαρμογής, και το ‘Admin’, που είναι η κύρια σελίδα διαχείρισης με λειτουργίες για τη διαχείριση προϊόντων. Συνολικά, το ‘App’ component δημιουργεί ένα ολοκληρωμένο περιβάλλον χρήστη για τη διαχείριση ενός ηλεκτρονικού καταστήματος.

### 2 index.css

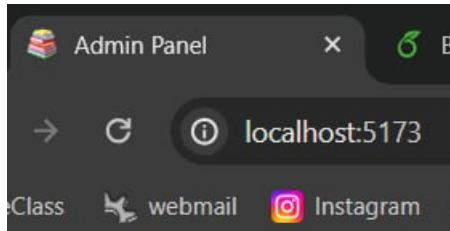
Αρχείο .css για styling

### 3 main.jsx

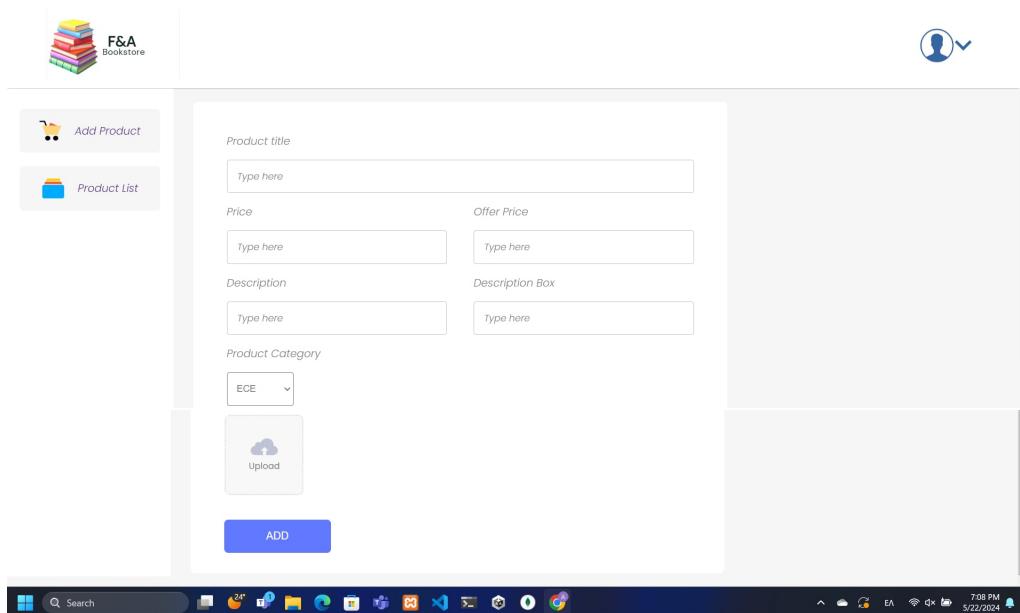
Ο κώδικας αυτός ξεκινά και εμφανίζει μια React εφαρμογή στη σελίδα, χρησιμοποιώντας το στοιχείο με id ‘root’ στο HTML. Χρησιμοποιεί το ‘ReactDOM.createRoot’ για να ορίσει πού θα εμφανιστεί η εφαρμογή και το ‘BrowserRouter’ για να υποστηρίζει την πλοήγηση μέσα στην εφαρμογή. Τυλίγει την εφαρμογή με το ‘React.StrictMode’ για να βοηθήσει στον εντοπισμό προβλημάτων κατά την ανάπτυξη. Τέλος, εμφανίζει το κύριο component ‘App’, κάνοντας την εφαρμογή έτοιμη για χρήση.

## 5.5 Screenshots from the Site

Ας ξεκινήσουμε από την αρχή. Η πρώτη εικόνα λοιπόν δείχνει τον τίτλο "Admin Panel" στην καρτέλα του προγράμματος περιήγησης, με τη διεύθυνση URL να είναι "localhost:5173".



Στην συνέχεια ας περιηγηθόμε στην Add Product, στην οποία ο Administration μπορεί να προσθέτει καινούρια προϊόντα:



Ας δούμε τώρα την σελίδα Product List, στην οποία ο Administration μπορεί να βλέπει τα προϊόντα που έχει στην βάση δεδομένων του αλλα και να τα διαγράφει:

Products	Title	Old Price	New Price	Category	Remove
	Ένα τίτολο μπορεί να αλλάξει τα πάντα	18.3€	12.99€	motivational	X
	Ανάμεσα σε ηλίθιους	17.7€	11.15€	motivational	X
	Can't Hurt Me	19.9€	17.99€	motivational	X
	Σε οι άξει να ενυπήγους	14.89€	10.95€	motivational	X
	Το Δάρο	14.26€	10.72€	motivational	X
	Ιστορία	11.1€	8.35€	motivational	X

Ας προσθέσουμε μαζί ενα νεό προϊόν. Αρχικά, απαιτούμε τη συμπλήρωση όλων των πεδίων και την προσθήκη μιας φωτογραφίας του προϊόντος πατώντας πάνω στο αντίστοιχο πεδίο. Έπειτα, κάνουμε κλικ στο κουμπί 'Add' και εμφανίζεται το μήνυμα επιτυχούς προσθήκης. Εαν κάποια από τα πεδία δεν συμπληρωθεί το προϊόν δεν προστίθεται.

**fill in**

Product title

Price

 Offer Price
 

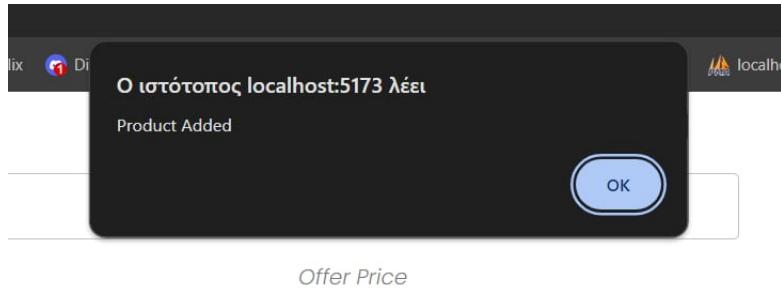
Description

 Description Box
 

Product Category

 choose category
 

add photo

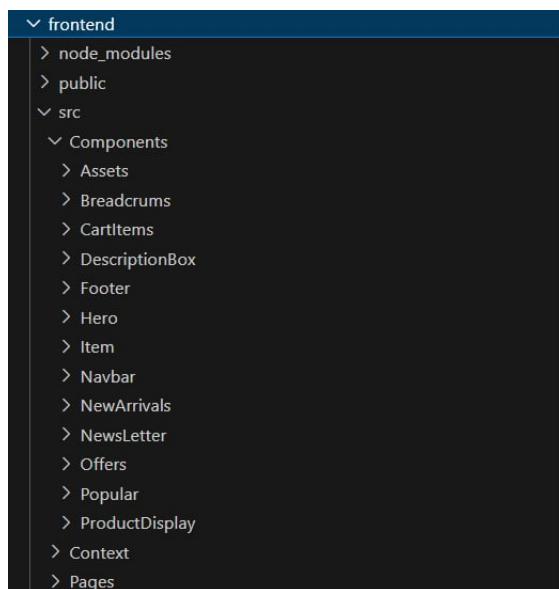


Αν θελήσουμε να το διαγράψουμε πηγαίνουμε στην λίστα με τα προϊόντα, το εντοπίζουμε και πατάμε το "remove icon":



## 6 Front-End

Ας περάσουμε στον φάκελο και στο κομμάτι του **Front End**. Όπως βλέπουμε παρακάτω στην φωτογραφία, υπάρχουν τρεις μεγάλοι φάκελοι. Εμείς θα ασχοληθούμε με τον φάκελο 'src', ο οποίος περιέχει τον κώδικα που αναπτύξαμε για την εργασία μας. Ο φάκελος 'src' περιέχει άλλους τρεις υποφακέλους: 'Components', 'Context' και 'Pages', τους οποίους θα δείξουμε και θα αναλύσουμε παρακάτω.



## 6.1 First Folder Components

Αρχικά, για το **Components**. Κάθε φάκελος που θα αναφέρουμε περιέχει δύο αρχεία: ένα .jsx(αρχείο JavaScript που χρησιμοποιείται στο React για να ορίζει τη δομή, τη λογική και τη λειτουργία των UI components μέσω του HTML-like syntax (JSX)) και ένα .css, που αφορά το styling.

- 1 Assets**  
Περιέχει όλες τις φωτογραφίες που χρησιμοποιήθηκαν στο site μας.
- 2 CartItems**  
Κώδικας για τη δημιουργία του περιβάλλοντος του καλαθιού αγορών.
- 3 Breadcrumb**  
Για την δημιουργία της διαδρομής (root directory) στην σελίδα κάθε βιβλίου
- 4 DescriptionBox**  
Δημιουργεί το πλαίσιο όπου τοποθετείται η περιγραφή ή η περίληψη κάθε βιβλίου στη σελίδα του.
- 5 Footer**  
Κώδικας για το κάτω μέρος κάθε σελίδας, που περιλαμβάνει το λογότυπο και πληροφορίες της εταιρίας.
- 6 Hero**  
Τμήμα στην αρχική σελίδα του site με διαφημιστικό και στυλιστικό ρόλο.
- 7 Item**  
Κώδικας για τον τρόπο εμφάνισης κάθε προϊόντος στην αρχική σελίδα ή στις κατηγορίες.
- 8 Navbar**  
Το μενού του site, που περιέχει την αρχική σελίδα, τις τρεις κατηγορίες βιβλίων και το καλάθι αγορών.
- 9 NewArrivals**  
Τμήμα στην αρχική σελίδα του site που παρουσιάζει τις νέες παραλαβές βιβλίων.
- 10 Newsletter**  
Τμήμα στην αρχική σελίδα όπου ο χρήστης υποβάλλει το email του για να λαμβάνει ειδοποιήσεις και διαφημίσεις από την εταιρία.
- 11 Offers**  
Τμήμα στην αρχική σελίδα που παρουσιάζει βιβλία σε προσφορά.
- 12 Popular**  
Τμήμα στην αρχική σελίδα που παρουσιάζει γνωστά και viral βιβλία της κατηγορίας “motivational”.
- 13 ProductDisplay**  
Κώδικας για τη διαμόρφωση της σελίδας κάθε βιβλίου.

## 6.2 Second Folder Context

Ο φάκελος αυτός περιέχει ένα αρχείο, ShopContextProvider.jsx , το οποίο είναι ένα React component που χρησιμοποιεί το Context API για να διαχειρίζεται την κατάσταση ενός ηλεκτρονικού καταστήματος (e-bookstore). Αρχικά, εισάγει τις συναρτήσεις createContext, useEffect και useState από το React. Δημιουργεί ένα νέο context για το κατάστημα με το ShopContext και καθορίζει την αρχική κατάσταση του καλαθιού με τη συνάρτηση getDefaultCart, η οποία δημιουργεί ένα καλάθι με 301 θέσεις, κάθε μία από τις οποίες έχει αρχική τιμή 0. Το ShopContextProvider χρησιμοποιεί το useState για να διαχειριστεί την κατάσταση των προϊόντων και των αντικειμένων στο καλάθι

(cartItems). Χρησιμοποιεί το `useEffect` για να φορτώσει τα δεδομένα των προϊόντων από ένα API και τα δεδομένα του καλαθιού από τον διακομιστή όταν το component γίνεται mount. Η συνάρτηση `addToCart` προσθέτει ένα προϊόν στο καλάθι και ενημερώνει τον διακομιστή αν υπάρχει authentication token, στέλνοντας ένα αίτημα POST στο κατάλληλο endpoint. Η συνάρτηση `removeFromCart` αφαιρεί ένα προϊόν από το καλάθι και ενημερώνει τον διακομιστή με τον ίδιο τρόπο. Η συνάρτηση `getTotalCartAmount` υπολογίζει το συνολικό ποσό του καλαθιού με βάση τις ποσότητες και τις τιμές των προϊόντων, ενώ η συνάρτηση `getTotalCartItems` υπολογίζει τον συνολικό αριθμό των αντικειμένων στο καλάθι. Το `contextValue` είναι ένα αντικείμενο που περιέχει όλες τις συναρτήσεις και τις καταστάσεις, όπως το `getTotalCartItems`, το `getTotalCartAmount`, τα `cartItems`, το `addToCart` και το `removeFromCart`, και παρέχεται στα child components μέσω του `ShopContext.Provider`. Τέλος, το `ShopContextProvider` εξάγεται για χρήση σε άλλα μέρη της εφαρμογής, παρέχοντας τις απαραίτητες λειτουργίες και δεδομένα στα child components μέσω του React Context API.

### 6.3 Third Folder Pages

Ο τρίτος και τελευταίος φάκελος αποτελείται από έναν φάκελο CSS που περιέχει αρχεία .css, και πέντε αρχεία .jsx: τα `Cart.jsx`, `LoginSignup.jsx`, `Product.jsx`, `Shop.jsx`, και `ShopCategory.jsx`, που θα αναλύσουμε παρακάτω.

#### 1 Cart

To ‘Cart’ component είναι ένα απλό React component που εμφανίζει το περιεχόμενο του καλαθιού. Εισάγει το React και το ‘CartItems’ component, και στη συνέχεια επιστρέφει ένα ‘div’ που περιέχει το ‘CartItems’ component. Με αυτόν τον τρόπο, το ‘Cart’ component λειτουργεί ως περιτύλιγμα που περιλαμβάνει και εμφανίζει τα στοιχεία του ‘CartItems’. Τέλος, εξάγει το ‘Cart’ component για να χρησιμοποιηθεί σε άλλα μέρη της εφαρμογής.

#### 2 LoginSignup

Ο κώδικας αυτός δημιουργεί ένα React component με το όνομα ‘LoginSignup’ που χειρίζεται τη διαδικασία σύνδεσης και εγγραφής χρηστών. Εισάγει το React και το hook ‘useState’ για τη διαχείριση της κατάστασης, καθώς και το αρχείο CSS για το styling.

Αρχικά, δημιουργεί δύο καταστάσεις: μία για να καθορίζει αν ο χρήστης βρίσκεται στη σελίδα σύνδεσης ή εγγραφής (‘state’) και μία για να αποθηκεύει τα δεδομένα της φόρμας (‘formData’). Η συνάρτηση ‘changeHandler’ ενημερώνει τα δεδομένα της φόρμας όταν ο χρήστης πληκτρολογεί. Υπάρχουν δύο συναρτήσεις, ‘login’ και ‘signup’, που εκτελούν τις αντίστοιχες ενέργειες σύνδεσης και εγγραφής μέσω αιτημάτων `fetch` σε ένα API. Αν η ενέργεια είναι επιτυχής, τοποθετούν ένα auth-token στο `localStorage` και ανακατευθύνουν τον χρήστη στην αρχική σελίδα. Σε περίπτωση αποτυχίας, εμφανίζουν μήνυμα σφάλματος.

Η λειτουργία του component περιλαμβάνει ένα πεδίο για το όνομα χρήστη (μόνο στην εγγραφή), καθώς και πεδία για email και κωδικό πρόσβασης. Ο χρήστης μπορεί να αλλάξει μεταξύ της σελίδας σύνδεσης και εγγραφής μέσω των αντίστοιχων επιλογών. Υπάρχει επίσης ένα checkbox για αποδοχή των όρων χρήστης και της πολιτικής απορρήτου.

#### 3 Product

Ο κώδικας δημιουργεί ένα React component με το όνομα ‘Product’, το οποίο χρησιμοποιεί τα hooks ‘useContext’ και ‘useParams’ για να αποκτήσει πρόσβαση στα δεδομένα του ‘ShopContext’ και να πάρει το ‘productId’ από το URL αντίστοιχα. Στη συνέχεια, βρίσκει το προϊόν που έχει το ίδιο ID με το ‘productId’. Το component αποδίδει ένα ‘div’ που περιέχει τα components ‘Breadcrumb’, ‘ProductDisplay’, και ‘DescriptionBox’, περνώντας το αντίστοιχο προϊόν ως prop σε καθένα από αυτά, επιτρέποντας την εμφάνιση των στοιχείων του προϊόντος με οργανωμένο τρόπο.

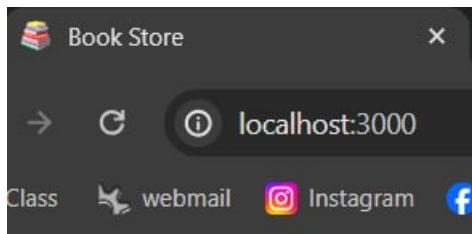
#### 4 Shop

Ο κώδικας αυτός δημιουργεί ένα React component με το όνομα ‘Shop’, το οποίο οργανώνει και εμφανίζει τα διάφορα τμήματα της αρχικής σελίδας του καταστήματος. Το ‘Shop’ component εισάγει και χρησιμοποιεί τα ακόλουθα components: ‘Hero’, ‘Popular’, ‘Offers’, ‘NewArrivals’, και ‘NewsLetter’. Το component αποδίδει ένα ‘div’ που περιλαμβάνει αυτά τα components διαδοχικά, δημιουργώντας έτσι μια πλήρη σελίδα καταστήματος που περιλαμβάνει ένα διαφημιστικό τμήμα, δημοφιλή προϊόντα, προσφορές, νέες αφίξεις και μια ενότητα για εγγραφή σε ενημερωτικό δελτίο. Τέλος, το ‘Shop’ component εξάγεται για χρήση σε άλλα μέρη της εφαρμογής.

**5 | ShopCategory** Το ‘ShopCategory‘ component είναι ένα React component που χρησιμοποιεί το ‘useContext‘ hook για να αποκτήσει πρόσβαση στη λίστα των προϊόντων από το ‘ShopContext‘. Εμφανίζει την κατηγορία προϊόντων στην αντίστοιχη σελίδα καταστήματος, περιλαμβάνοντας μια ενότητα με πληροφορίες για τον αριθμό των προϊόντων και ένα dropdown για την ταξινόμηση. Μέσα σε ένα ‘div‘ με κλάση ‘shop-category‘, φιλτράρει και εμφανίζει τα προϊόντα που ανήκουν στην κατηγορία που καθορίζεται από τα props, χρησιμοποιώντας το ‘Item‘ component για κάθε προϊόν. Τέλος, περιλαμβάνει ένα κουμπί ”Explore more“ για την εξερεύνηση περισσότερων προϊόντων.

## 6.4 Screenshots from the Site

Ας ξεκινήσουμε από την αρχή. Η πρώτη εικόνα λοιπόν δείχνει τον τίτλο και το favicon (εικονίδιο) του ιστότοπου μας ”Book Store“ στην καρτέλα του προγράμματος περιήγησης, με τη διεύθυνση URL να είναι ”localhost:3000“.



Στην συνέχεια ας περιηγηθόμεμε στην αρχική σελίδα του site μας. Η σελίδα αυτή περιέχει ένα διαφημιστικό τμήμα, δημοφιλή προϊόντα, προσφορές, νέες αφίξεις, μια ενότητα για εγγραφή σε ενημερωτικό δελτίο και τέλος το footer, οπως θα δείξουμε στις παρακάτω φωτογραφίες. Να σημειωθεί πως τα βιβλία που εμφανίζονται στις κατηγορίες δημοφιλή, προσφορές και νέες αφίξεις είναι δυναμικά και ανανεώνονται κάθε φορά που υπάρχει αλλαγή στην βάση δεδομένων μας

The screenshot shows the homepage of the Book Store website. At the top, there's a navigation bar with links for 'Shop', 'ECE', 'Romantic', 'Motivational', 'Logout', and a shopping cart icon with a '2' notification. Below the header, a large pink banner features the text 'CHECK NEW ARRIVALS' and 'new books on site' with a hand icon. To the right of the banner is an illustration of a person sitting cross-legged, reading a book, surrounded by a stack of colorful books. Below the banner, a red button says 'Latest Additions →'. Further down, a section titled 'POPULAR IN MOTIVATIONAL' displays four book covers:

- Atomic Habits** by James Clear: 12.99€ 10.39€
- ανάμεσα σε πλιθίους** by Thomas Erikson: 11.15€ 17.77€
- CAN'T HURT ME** by David Goggins: 17.99€ 19.99€
- Σου αγίζει να ευτυχέσεις** by Agni Mariakaki: 10.95€ 14.89€

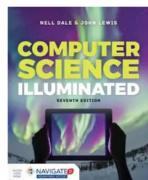
Each book cover includes a small description and price.

# Exclusive Offers For You

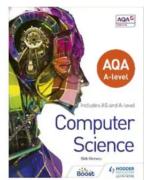
ONLY ON BEST SELLERS PRODUCTS

[Check Now](#)

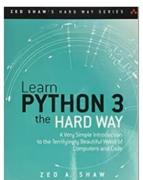
## NEW ARRIVALS



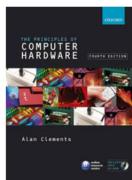
Computer Science Illuminated  
79.99€ 64.99€



AQA A level Computer Science  
69.99€ 70.99€



Learn Python 3 the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code  
38.99€ 43.99€



Principles of Computer Hardware  
99.99€ 120.99€



Computer Architecture  
109.99€ 129.99€



Compute-IT: Student's Book 2 – Computing for KS3  
39.99€ 42.99€



Learning Computer Architecture with Raspberry Pi  
29.99€ 31.99€



AQA GCSE (9-1) Computer Science 8525  
27.99€ 36.99€

## Get Exclusive Offers On Your Email

Subscribe to stay updated

Your Email

Subscribe



### BOOKSTORE

Books that inspire you

[About us](#) [Products](#) [Contact](#)



Ας δούμε αναλυτικά κάθε σελίδα της κάθε κατηγορίας βιβλίων. Οπως και προηγουμένως ετσι και σε κάθε σελίδα βιβλίων, τα βιβλία εμφανίζονται δυναμικά και ανανεώνονται κάθε φορά που υπάρχει αλλαγή στην βάση δεδομένων μας. Ας ξεκινήσουμε με τα βιβλία για Ηλεκτρολόγους μηχανικούς και μηχανικούς υπολογιστών "ECE":

 BOOKSTORE
Shop
ECE
Romantic
Motivational
[Logout](#)
 2

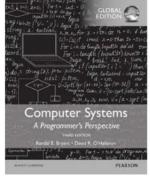
Showing 1-12 out of 36 products

Sort by ▾



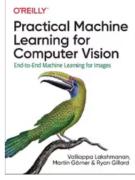
Computer Vision and Image Analysis for Industry 4.0

139.99€ 148.99€



Computer Systems: A Programmer's Perspective, Global Edition

122.99€ 134.99€



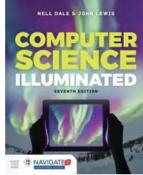
Practical Machine Learning for Computer Vision

79.99€ 88.99€



Object-Oriented Systems Analysis and Design Using UML

93.99€ 98.99€



Computer Science Illuminated

79.99€ 84.99€



AQA A level Computer Science

69.99€ 70.99€



Learn Python 3 the Hard Way

38.99€ 43.99€



The Art of War: Computer Hardware

99.99€ 120.99€



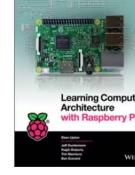
Computer Architecture

109.99€ 129.99€



Compute-It Computing for KS3

39.99€ 42.99€



Learning Computer Architecture with Raspberry Pi

29.99€ 31.99€



AQA GCSE (9-1) Computer Science 8525

27.99€ 30.99€

[Explore more](#)



BOOKSTORE

Books that inspire you

[About us](#)
[Products](#)
[Contact](#)

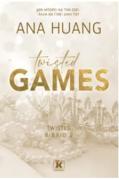
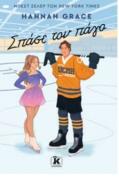
Search
26°
22:55 PM
5/22/2024

Συνεχίζουμε με την κατηγορία "Romantic":

 BOOKSTORE
Shop
ECE
Romantic
Motivational
[Logout](#)
 2

Showing 1-12 out of 36 products

Sort by ▾

 Twisted love 11.15€ 17.7€	 Twisted games 15.98€ 17.7€	 Twisted hate 16.99€ 10.0€	 Τελείωνει με εμάς 12.99€ 17.7€
 Εκεί πού τραγούδανε οι καραβίδες 14.99€ 20€	 Σκοτεινή αγάπη 12.99€ 17.7€	 Δημοσιογράφος δακρύων 18.99€ 22€	 Το πράδονό ζημειονατάριο 13.99€ 10.0€
 Αρχίζει με εμάς 11.15€ 17.7€	 Σπάσε τον πάρο 17.7€ 10.0€	 Θεονό, η λύκανθρωπη της πόλης 12.99€ 17.7€	 Λένα Μάντα Μια συγγένιμη για το τέλος 12.99€ 14.9€

[Explore more](#)


BOOKSTORE

Books that inspire you

[About us](#)
[Products](#)
[Contact](#)



ENG US
2:31 PM
5/22/2024

Τελευταία κατηγορία του site μας είναι τα "Motivational":

**BOOKSTORE**

Shop ECE Romantic Motivational Logout

Showing 1-12 out of 36 products

Sort by ▾

Ένα πίστωτα μπορεί να αλλάξει τα πάντα 12.99€ 10.99€	Ανάμεσα σε ηλίθιους 11.15€ 11.7€	Can't Hurt Me 17.99€ 16.9€	Σου αξίζει να ευτυχίσεις 10.95€ 14.09€
Το Δέρο 10.72€ 14.26€	Ικιγκάι 8.35€ 11€	Το κλάμπ των 5 π.μ. 12.99€ 17.7€	Είμαι ότι καλύτερο μου έχει συμβει 7.75€ 9.9€
Βάλε τα οριά σου 12.2€ 16.6€	Να ζεις, ν' αγαπάς και να μαθαίνεις 12.52€ 14.9€	Το βανό είσαι εσύ 14.99€ 16.6€	Manifest 14.39€ 17.99€

Explore more

**BOOKSTORE**  
Books that inspire you

About us Products Contact

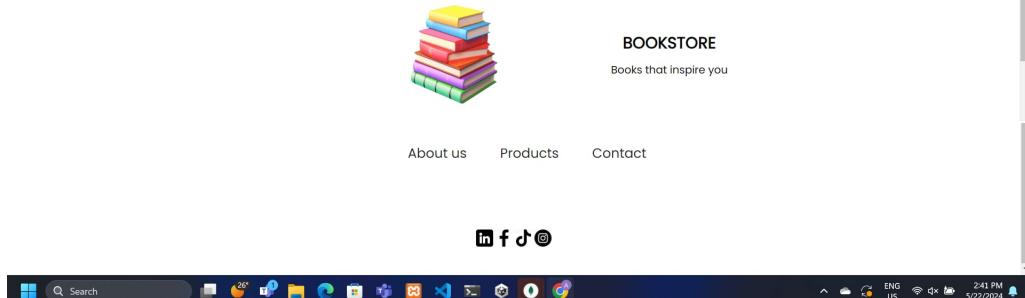
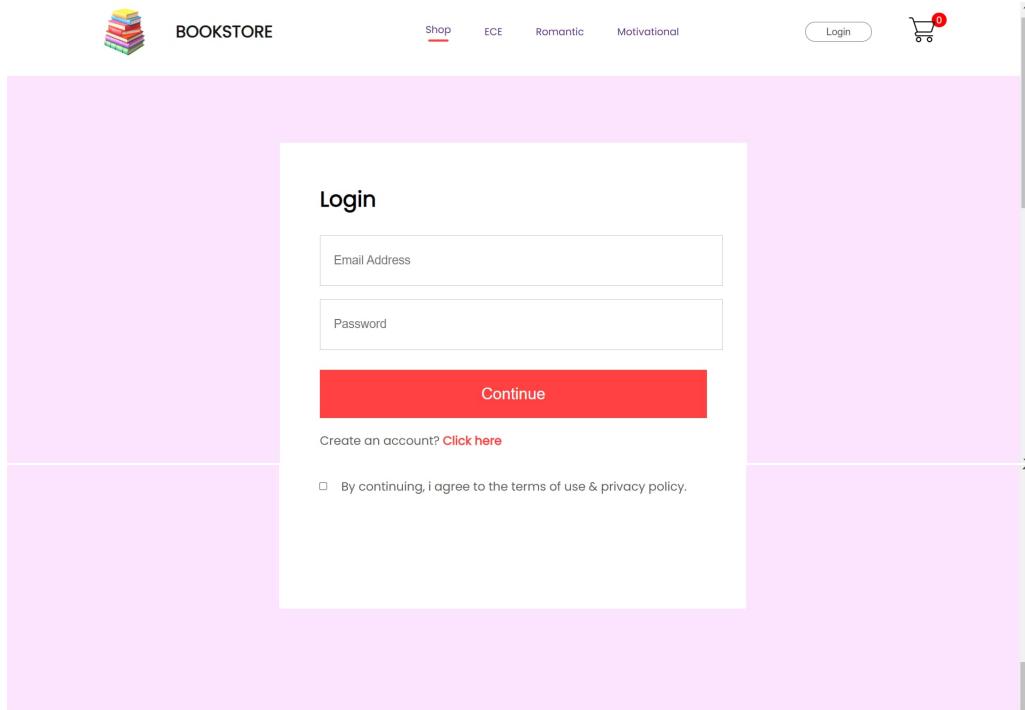
[in](#) [f](#) [d](#) [o](#)

26° Search ENG UK 2:38 PM 5/22/2024

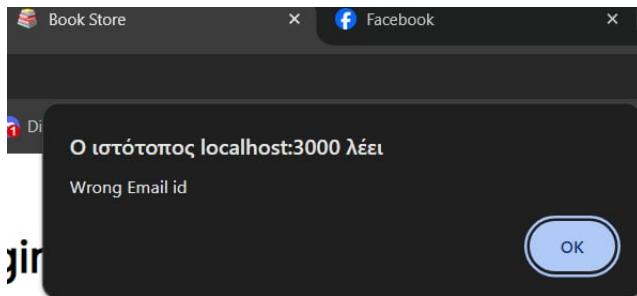
Πατώντας πάνω στη φωτογραφία ενός βιβλίου που μας ενδιαφέρει, ανακατευθυνόμαστε στη σελίδα του. Αυτή η σελίδα περιέχει πληροφορίες για το βιβλίο, όπως τον τίτλο, την τιμή, την περιγραφή (συγγραφέα, εκδόσεις, αριθμό σελίδων) και την περίληψη του. Πατώντας το κουμπί "Προσθήκη στο Καλάθι" (ADD TO CART), το βιβλίο προστίθεται στο καλάθι μας. Στις παρακάτω φωτογραφίες, έχουμε ήδη πατήσει το κουμπί "Προσθήκη στο Καλάθι" (ADD TO CART). Με αυτή την ενέργεια, δίπλα στο εικονίδιο του καροτσιού εμφανίστηκε ο αριθμός ένα, που υποδηλώνει πόσα προϊόντα έχω στο καλάθι μου.

The screenshot shows a product page for the book 'Object-Oriented Systems Analysis and Design Using UML'. At the top, there's a navigation bar with icons for books, 'BOOKSTORE', 'Shop', 'ECE' (highlighted in red), 'Romantic', 'Motivational', 'Logout', and a shopping cart icon with a red notification badge. Below the navigation is a breadcrumb trail: HOME > SHOP > Ece > Object-Oriented Systems Analysis And Design Using UML. The main content area features a large image of the book cover, which is blue with white text and a network-like graphic. To the left of the main image are five smaller thumbnail images of the book from different angles. To the right of the image, the book title is displayed in bold: 'Object-Oriented Systems Analysis and Design Using UML'. Below the title is a rating of 4 stars from 122 reviews. The original price is 98.99€, and the discounted price is 93.99€. The book is published by McGraw-Hill Education - Europe, part of the UK Higher Education Computing Computer Science series, with 688 pages. A red 'ADD TO CART' button is below the price. Underneath the book details, the category 'Category: ece' is listed. On the left, there's a sidebar with a 'Description' tab selected, containing a detailed description of the book's content and purpose. At the bottom of the page, there's a footer with a logo of stacked books, the text 'BOOKSTORE Books that inspire you', links to 'About us', 'Products', and 'Contact', social media icons (LinkedIn, Facebook, YouTube, Instagram), and a Windows taskbar at the very bottom showing various open applications and system status.

Θα συνεχίσουμε με το Login/Signup page:

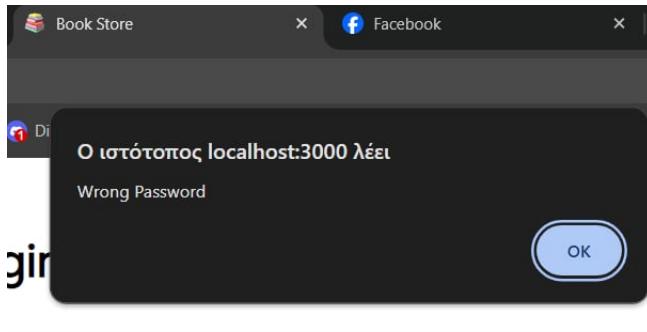


Συμπληρώνοντας τα στοιχεία σώστα συνδέεσαι και ανακατευθύνεσαι στην αρχική σελίδα. Εάν ομως τοποθετήσεις λάθος Email εμφανίζεται αυτό το μήνυμα :



xhpapado@gmail.co

Και εάν τοποθετήσεις λάθος τον κωδικό σου εμφανίζεται αυτό το μήνυμα :



Αφού συνδεθείς, το μενού αλλάζει και το κουμπί LOGIN αντικαθίσταται από το κουμπί LOGOUT. Πατώντας το κουμπί LOGOUT, αποσυνδέεσαι από τον λογαριασμό σου.



Εάν όμως δεν έχεις λογαριασμό πατώντας στο κουμπί LOGIN από την αρχική σελίδα και έπειτα εδώ σου εμφανίζεται η φόρμα για να εγγραφείς :

Login

Email Address
Password
<input type="button" value="Create account? Click here"/>

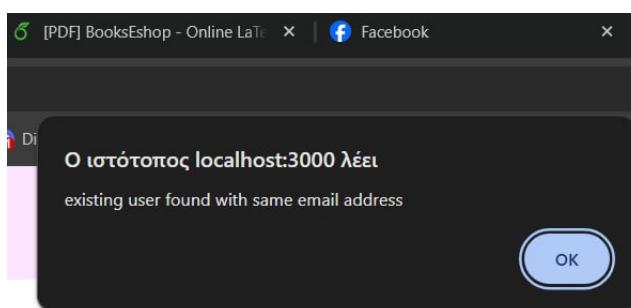
By continuing, i agree to the terms of use & privacy policy.

Πατώντας λοιπόν στο "click here" ανακατευθυνόμαστε στην Sign up page :

The screenshot shows a web browser window with a pink header bar. The main content area displays a "Sign up" form with three input fields: "Your Name", "Email Address", and "Password". Below the form is a red "Continue" button. Underneath the button, there is a link "Already have an account? [Login here](#)". At the bottom of the form, there is a checkbox labeled "By continuing, I agree to the terms of use & privacy policy.".

Below the sign-up form, the website's header is visible, featuring a logo of stacked books, the word "BOOKSTORE", and the tagline "Books that inspire you". Navigation links for "About us", "Products", and "Contact" are also present. At the very bottom of the screen, the Windows taskbar is shown with various pinned icons and system status indicators.

Κατά την εγγραφή, εάν δεν συμπληρώσετε όλα τα απαιτούμενα πεδία, θα ανακατευθυνθείτε ξανά στην ίδια σελίδα με μήνυμα λάθους που αναφέρει ότι υπάρχουν ελλιπή στοιχεία. Σε περίπτωση που το email που εισάγεται είναι ήδη καταχωρημένο, θα εμφανιστεί το εξής μήνυμα:



Συνεχίζουμε με το καλάθι μας. Εχουμε προσθέσει ήδη δυο βιβλία σε αυτό:

The screenshot shows a website for a bookstore. At the top, there's a navigation bar with a logo of stacked books, the word "BOOKSTORE", and links for "Shop", "ECE", "Romantic", "Motivational", "Logout", and a shopping cart icon with a "2" indicating two items. Below this is a table of products in the cart:

Products	Title	Price	Quantity	Add/Remove	Total
	Ανάμεσα σε ηλίθους	11.15€	<input type="text" value="1"/>	+ -	11.15€
	Twisted hate	16.99€	<input type="text" value="1"/>	+ -	16.99€

Below the cart table is a section titled "Cart Totals" showing the subtotal, shipping fee, and total:

Subtotal	28.14€
Shipping Fee	Free
<b>Total</b>	<b>28.14€</b>

A red button labeled "PROCEED TO CHECKOUT" is visible. To the right, there's a field for entering a promo code with a "Submit" button. The website footer includes a logo of stacked books, the word "BOOKSTORE", the tagline "Books that inspire you", and links for "About us", "Products", and "Contact". Social media icons for LinkedIn, Facebook, YouTube, and Instagram are also present.

Πατώντας το συν (Add) αυξάνεται η ποσότητα του κατα ένα :

This screenshot shows the same shopping cart table as above, but with a red arrow pointing to the "+" button next to the quantity input field for the first item, "Ανάμεσα σε ηλίθους". The quantity input field contains the number "1".

Products	Title	Price	Quantity	Add/Remove	Total
	Ανάμεσα σε ηλίθους	11.15€	<input type="text" value="1"/>	+ <span style="color:red;">↖</span> -	11.15€
	Twisted hate	16.99€	<input type="text" value="1"/>	+ -	16.99€

Επομένως πλέον έχουμε 2 ποσότητες του ίδιου βιβλίου:

Products	Title	Price	Quantity	Add/Remove	Total
	Ανάμεσα σε ηλιθίους	11.15€	2	+ -	22.3€
	Twisted hate	16.99€	1	+ -	16.99€

Αντίστοιχα πατώντας το πλην (Remove) μειώνεται η ποσότητα του κατα ένα :

Products	Title	Price	Quantity	Add/Remove	Total
	Ανάμεσα σε ηλιθίους	11.15€	2	+ -	22.3€
	Twisted hate	16.99€	1	+ -	16.99€

Σε αυτή την φωτογραφία έχουμε πατήσει το κουμπί remove 2 φορές με αποτέλεσμα η ποσότητα του προιόντος να είναι 0 και να εξαφανίζεται από το καλάθι:

Products	Title	Price	Quantity	Add/Remove	Total
	Twisted hate	16.99€	1	+ -	16.99€

Τέλος πατώντας το κουμπί "PROCEED TO CHECKOUT" ολοκληρώνουμε την παραγγελία μας και εμφανίζεται το παρακάτω μήνυμα :

PROCEED TO CHECKOUT



Η παραγγελία σας ολοκληρώθηκε.  
Ευχαριστούμε πολύ για την στήριξη σας!

## 7 Επίλογος

Συμπερασματικά, η παρούσα εργασία αποσκοπεί στην ανάπτυξη ενός πλήρως λειτουργικού ηλεκτρονικού καταστήματος (e-shop) για την πώληση βιβλίων, εστιάζοντας στη δημιουργία μιας εύχρηστης, ασφαλούς και αποδοτικής πλατφόρμας. Η πλατφόρμα επιτρέπει στους χρήστες να περιηγούνται, να αναζητούν και να αγοράζουν βιβλία με ευκολία, ενώ προσφέρει εξελιγμένες δυνατότητες διαχείρισης προϊόντων για τον διαχειριστή, όπως προσθήκη, επεξεργασία, διαγραφή και προβολή προϊόντων, κατηγοριοποίηση σε δημοφιλή, νέα ή εκπτωτικά, διαχείριση πελατών με εγγραφή και σύνδεση. Μελλοντικές επεκτάσεις περιλαμβάνουν τη διαχείριση παραγγελιών με προβολή και ακύρωση, σύστημα αξιολόγησης με σχόλια από πελάτες, και διαχείριση αποθεμάτων με αυτόματη ανανέωση βάσει πωλήσεων. Αυτές οι βελτιώσεις θα ενισχύσουν περαιτέρω την εμπειρία χρήστη και τη λειτουργικότητα της πλατφόρμας, καθιστώντας την ακόμα πιο αποδοτική και ευχάριστη. Τέλος, είμαστε ευχαριστημένες με το τελικό αποτέλεσμα και την λειτουργικότητα της ιστοσελίδας μας.

## References

- [1] MongoDB. *Best NoSQL Databases in 2023: Key Features and Benefits*. 2023. url: <https://www.mongodb.com/resources/basics/databases/nosql-explained/best-nosql-database>.
- [2] YouTube. *All React Hooks Explained in 2 Hours*. url: <https://www.youtube.com/watch?v=6wf5dIrryoQ>.
- [3] YouTube. *Basic API Calls with Thunder Client*. url: <https://www.youtube.com/watch?v=c3sqFK7zBKE>.
- [4] YouTube. *E-Commerce API with NodeJS*. url: <https://www.youtube.com/watch?v=hPv9QwvliEM&list=PLzb46hGUzitBp584kLyn6l3i6yC-rXlmN>.
- [5] YouTube. *How to Connect Node.JS with MongoDB using Mongoose*. url: <https://www.youtube.com/watch?v=ACUXjXtG8J4&t=39s>.
- [6] YouTube. *How to create node js project using npm*. url: <https://www.youtube.com/watch?v=0JD9TIKRRWc>.
- [7] YouTube. *How to install MongoDB on Windows 10/11*. url: <https://www.youtube.com/watch?v=OjMOQr457Qs>.
- [8] YouTube. *How to Install Node.js on Window 11*. url: <https://www.youtube.com/watch?v=06X51c6WHsQ>.
- [9] YouTube. *How To Make Navbar In React JS*. url: <https://www.youtube.com/watch?v=5m0-T2o9zuk>.
- [10] YouTube. *How To Make Sign In Sign Up Form Using React JS*. url: <https://www.youtube.com/watch?v=8QgQKRcAUvM>.
- [11] YouTube. *Learn CSS in 20 Minutes*. url: [https://www.youtube.com/watch?v=1PnVor36\\_40](https://www.youtube.com/watch?v=1PnVor36_40).
- [12] YouTube. *ReactJS Tutorial*. url: <https://www.youtube.com/watch?v=QFaFIcGhPoM&list=PLC3y8-rFhvwg3vaYJgHGnModB54rxOk3>.
- [13] YouTube. *What Is JavaScript*. url: [https://www.youtube.com/watch?v=CBWnBi-awSA&list=PLjwm\\_8O3suyM61TZY1w5ufD12nRQCtd2N](https://www.youtube.com/watch?v=CBWnBi-awSA&list=PLjwm_8O3suyM61TZY1w5ufD12nRQCtd2N).