

BAZAR.COM

PART1

Mohammed Jaddou - **12112568**

Ahmad Sultan - **12112656**

INTRODUCTION

The main goals of this project are to create a system for tracking orders, buying books, and maintaining a book catalog. It is a full-stack application that makes use of a server constructed with Express and Node.js, and CSV files are used for data manipulation and storage. There are several parts to the project:

- Book catalog management (adding, updating, and searching books).
- Order management (purchasing books, logging orders).
- Stock management (decreasing stock based on purchases).

The project simulates an e-commerce-like system for books, where users can search for books by topic, purchase them, and the stock is updated accordingly.

TECHNOLOGY STACK

- Backend: Node.js, Express
- CSV Handling: csv-parser, csv-writer
- Data Storage: CSV files
- Testing Tool: Postman for API testing
- Docker

FEATURES

BOOK CATALOG MANAGEMENT

The system allows users to:

- Search books by topic.
- View detailed information about a specific book by its ID.
- Update book prices and stock based on purchasing.

ORDER MANAGEMENT

The system also allows users to:

- Purchase books by specifying the quantity.
- Log each purchase into the orders.csv file.

API ENDPOINTS

The system's primary API endpoints are listed below:

Method	Endpoint	Description
GET	/search/:topic	Searches for books based on the topic specified in the request.
GET	/info/:item_number	Retrieves detailed information about a specific book using its item number (ID).
PUT	/update	Updates the price and stock of a book specified in the request body.
POST	/purchase/:item_number	Allows a user to purchase a book and logs the order in orders.csv.

RESULTS

The following section provides step-by-step instructions on how to test the API using Postman.

HTTP://LOCALHOST:3000/SEARCH/UNDERGRADUATE SCHOOL

The screenshot shows the Bazaar API testing interface. On the left, there's a sidebar with 'Collections' (selected), 'Environments', 'History', 'Flows', and 'Files (BETA)'. The main area shows a 'New Collection / search' tab. A 'GET' request is made to `http://localhost:3000/search/undergraduate school`. The response status is `200 OK` with a response time of 97 ms and a body size of 412 B. The response JSON is:

```
1 {
2   "topic": "undergraduate school",
3   "count": 2,
4   "items": [
5     {
6       "id": 3,
7       "title": "Xen and the Art of Surviving Undergraduate School"
8     },
9     {
10       "id": 4,
11       "title": "Cooking for the Impatient Undergrad"
12     }
13   ]
14 }
```

HTTP://LOCALHOST:3000/INFO/2

The screenshot shows the Bazaar API testing interface. On the left, there's a sidebar with 'Collections' (selected), 'Environments', 'History', 'Flows', and 'Files (BETA)'. The main area shows a 'New Collection / book info' tab. A 'GET' request is made to `http://localhost:3000/info/2`. The response status is `200 OK` with a response time of 31 ms and a body size of 359 B. The response JSON is:

```
1 {
2   "title": "RPCs for Noobs",
3   "topic": "distributed systems",
4   "quantity": 10,
5   "price": 50,
6   "item_number": 2,
7   "stock_status": "In Stock"
8 }
```

HTTP://LOCALHOST:3001/UPDATE/2

The screenshot shows the Postman interface with a collection named "bazar.com". A PUT request is being made to the endpoint `http://localhost:3001/update/2`. The request body is set to raw JSON with the value `{ "quantity": 10 }`. The response is a 200 OK status with the following JSON content:

```
1 {  
2   "success": true,  
3   "message": "Item updated"  
4 }
```

catalog.csv	
1	<code>id,title,topic,quantity,price</code>
2	<code>1,How to get a good grade in DOS in 40 minutes a day,distributed systems,10,30</code>
3	<code>2,RPCs for Noobs,distributed systems,10,50</code>
4	<code>3,Xen and the Art of Surviving Undergraduate School,undergraduate school,8,40</code>
5	<code>4,Cooking for the Impatient Undergrad,undergraduate school,12,25</code>
6	

HTTP://LOCALHOST:3000/PURCHASE/2

The screenshot shows the Postman interface. On the left, there's a sidebar with 'Collections' (selected), 'Environments', 'History', 'Flows', and 'Files (BETA)'. The main area shows a 'New Collection / purchase' with a 'POST' method selected. The URL is 'http://localhost:3000/purchase/2'. The response status is '200 OK' with a duration of '151 ms' and a size of '358 B'. The response body is a JSON object:

```
1 {  
2     "success": true,  
3     "item_number": 2,  
4     "title": "RPCs for Noobs",  
5     "price": 50,  
6     "message": "Successfully purchased \"RPCs for Noobs\""  
7 }
```

catalog.csv	
1	id,title,topic,quantity,price
2	1,How to get a good grade in DOS in 40 minutes a day,distributed systems,10,30
3	2,RPCs for Noobs,distributed systems,9,50
4	3,Xen and the Art of Surviving Undergraduate School,undergraduate school,8,40
5	4,Cooking for the Impatient Undergrad,undergraduate school,12,25
6	

DOCKER CONTAINERS

PS C:\Users\JADDOU3> docker ps						
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	
	NAMES					
7ab072d8aedf	bazarcom-front-end :3000->3000/tcp	"docker-entrypoint.s..."	6 days ago	Up 13 minutes	0.0.0.0:3000->3000/tcp, [::]	frontend_server
68747cf33e07	bazarcom-order-server :3002->3002/tcp	"docker-entrypoint.s..."	6 days ago	Up 13 minutes	0.0.0.0:3002->3002/tcp, [::]	order_server
227c4fd5edf8	bazarcom-catalog-server :3001->3001/tcp	"docker-entrypoint.s..."	6 days ago	Up 13 minutes	0.0.0.0:3001->3001/tcp, [::]	catalog_server