DOS project

Manal Nidal Khalili
/11925863

Bazar.com
Book store

Multi-tier-Online-Book-Store

The store will leverage microservices at each layer of its two-tier web design, which consists of a front-end and a backend. Initial processing of user requests is done by the front-end tier. A catalogue server and an order server are the two parts of the backend.

Architecture

1. Catalogue Server: It has a database with details about the books that are available. This data contains specifics like the book's title, price, amount in stock, and subject matter. serves as a microservice devoted to organising and supplying data regarding the books that are available. It provides a RESTful API that supports tasks like changing book data and doing subject or item number queries for books, and it is implemented as a stand-alone process.

2. Order Server: In charge of overseeing client orders. To make the purchasing process easier, the order server—which is implemented as a microservice—communicates with the front-end and catalogue servers.This microservice offers a single operation endpoint via a RESTful API and is intended to manage the buying process effectively.

3. Frontend Server: Coordinates communication between the user interface and the backend services and serves as the first point of contact for user requests. The front-end server, which is implemented as a microservice, provides a RESTful API that allows it to support various user tasks.

Implementation

Express was used in conjunction with Node.js to develop this project, taking use of its ability to facilitate the creation of lightweight microservices. The SQLite database was specifically chosen because it offers a more straightforward and lightweight data storage option, which is consistent with the objective of developing a simplified and effective programme design.

usage

The project deals with Database sqlite3, so you must download this dependency on your device Download the file and place each server in a different project, configuring it for Node.js and Framework Express, to run the `front-end server`.

APIs

FrontEnd APIs

URL	Method	Description
`/info/:item_number`	`Get`	get information about a book by its `<item_number>`
`/search/:topic`	`Get`	search for books by their `<topic>`, returns all matching books itemNumber and tittle.
`/buy/:book_id`	`Post`	buy a book by its `<book_id>`

Catalog Server APIs

URL	Method	Description
`/query/info/:item_number`	`Get`	query all information about a book by its `<item_number>`, returns `JSON` object represinting matching books information.
`/query/search/:topic`	`Get`	query all the books with the specified `<topic>`, returns `JSON` object represinting matching books information.
`/update/:book_id`	`Post`	update the stock for book by its `<book_id>`

Order Server APIs

URL	Method	Description
`/buy/:book_id`	`Post`	send query for catalog server to buy a book by its `<book_id>`