Project Report
La Librairie: The online bookstore

Student Name: Nguyen Hoang Long
Student ID: s3727634
Course: COSC2638 - Cloud Computing
Topic: Build a scalable app on Clouds

Table of Contain

Table of Contents

List of Tables

List of Figures

Project Information

Link: http://la-librairie.info
Admin Account: s3727634@rmit.edu.vn – Admin@123
Backup Link: http://d1rt11et9oyrr5.cloudfront.net
Backend Github:https://github.com/HoangLong256/bookstore-backend
Frontend Github:https://github.com/HoangLong256/bookstore-frontend
Member: Nguyen Hoang Long(100%)

Summary

This purpose of the project is to build an online shopping website for online shopping. This is a website where customer can visit to review and buy the books they want. The website also provides access to the owner for reviewing orders or adding more products. Moreover, based on this simple website, we can improve and expand it to an e-commerce website such as Amazon or Tiki.

1. Introduction

1.1. Motivation and Reason

Due to the effect of Covid-19, online shopping becomes a trend in our daily life, it helps us buy anything we want without going to the shopping store. Many people have transferred their local store to an online platform for marketing strategy. Thus, I have come up with an idea about a website that people can introduce their products and record the orders from the customer. Moreover, the website is easy to customize and transfer from the book store to any kind of selling products.

1.2. Features

a. Displaying the list of products

When the customer visits the website, it will display the list of books to the screen. The user can view books details by clicking on the button under book items.

b. Buying books

If the customer wants to buy the book, they can select the quantity and add it into their bag. After selecting the favourite books, he/she could go to the 'My Bag' tab to review the order or remove the unwanted item.
Then, the user can fill in the information form and submit the order.

c. Authorization

This is the function that the owner can do to manage the orders as well as products.

For the security reason, the signup function is disabled. Only the user with the provided account can log in to the database management tab.

d. Product Management

In this function, the admin can view all the product as well as add, edit or delete them.

e. Order Management

In this function, the admin can view all the product as well as update shipping status or delete them.

1.3. Real Life application

This website can be applied to the real-life immediately, especially the online bookstore. The owner can add the book detail and introduce their website to customers. Then, he can manage the orders by login into the management tab.

1.4. Advantages/Positive/New thing

- Serverless: This backend of the website is built by a serverless framework with AWS Lamda function. We do not need a server to run and maintain and we only pay the fee for the request we use. This means the cost of running and maintaining will be tremendously reduced.
- Microservices: The architecture of the website is microservices, that means we can access it with any devices and can build the frontend on any languages. In this case, the frontend of the website is built by ReactJS framework.
- Responsive: The project is a responsive website so that we can view it on small screens or large screens such as smartphones or laptops...

2. Related Work

Table 1: Related Work

Website	**URL**
Tiki	tiki.vn
Lazada	lazada.vn
Shopee	shopee.vn

3. Software Design/Architecture

3.1. Software Diagram

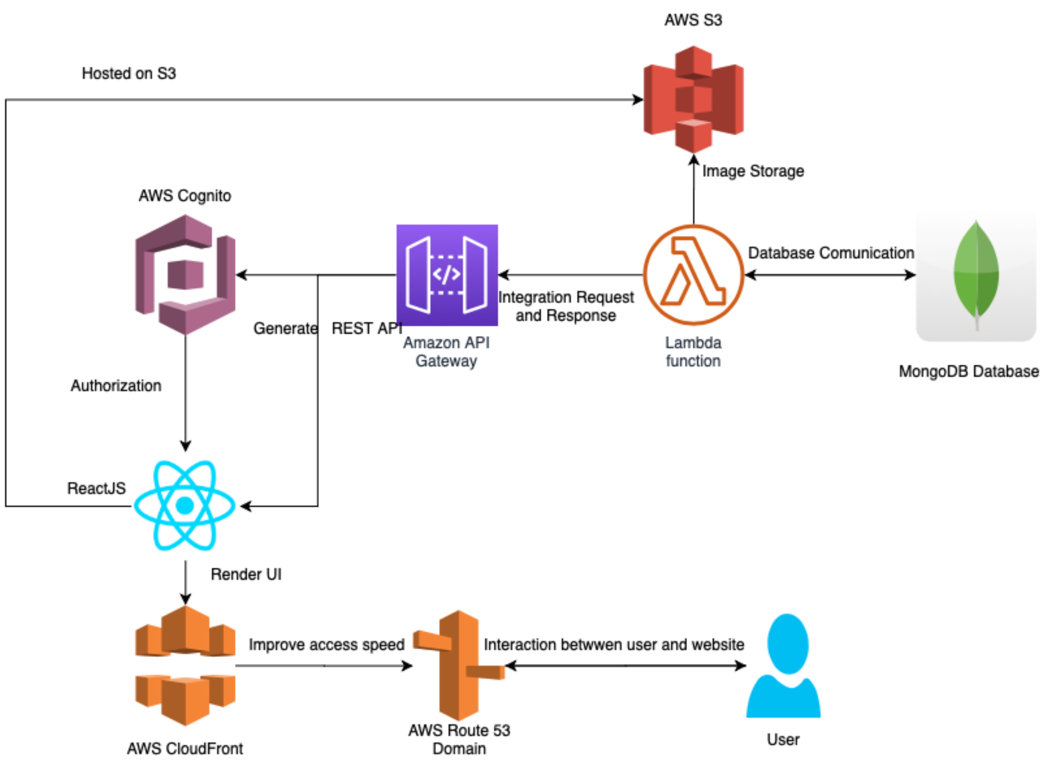

Figure 1: Architecture Design

3.2. Services

Table 2: Architecture Services

Services	**Purpose**
MongoDB	Data storage
AWS Lambda Function	Execute the backend code without worrying about server
AWS API Gateway	Generate HTTP API
AWS Cognito	User and API Authorization
AWS S3	Image Storage and Web Static Hosting
ReactJS	Frontend
AWS CloudFront	CDN that improve speed access
AWS Route 53	Domain name

Ï

3.3. Database Structure

In this project, we use two documents for database storage

a. Product

Table 3: Product Data Structure

Attribute	**Type**
ID	Object.ID
Title	String
Author	String
Publisher	String
Pages	Number
Price	Number
Availability	Boolean
Image	String
Year	Number

b. Order

Table 4: Order Data Structure

Attribute	**Type**
ID	Object.ID
Email	String
Phone	Number
Name	String
Address	String
Price	Number
Total Price	Number
Shipped	Boolean
Books	Array of Object

In each object of the array, we have those attribute

Table 5: Books Data Structure

Attribute	**Type**

ID	Object.ID (linked to Product ID)
Title	String (Generate based on Product ID)
Price	Number (Generate based on Product ID)
Quantity	Number

3.4. API

URL: https://aef1nhb6nb.execute-api.ap-southeast-1.amazonaws.com/dev

Table 6: API Endpoints

Endpoints	Method	Usage
/books	POST	Create product detail (book)
/books	GET	Get all products
/books/{id}	GET	Get a product by id
/books/{id}	PUT	Update a product by id
/books/{id}	DELETE	Delete a product by id
/orders	POST	Create orders detail
/orders	GET	Get all orders
/orders/{id}	GET	Get a orders by id
/orders/{id}	PUT	Update a orders by id
/orders/{id}	DELETE	Delete orders by id

4. Implementation-Developer Manual

4.1. Creating a database on MongoDB Atlas

Example Guide: https://hackernoon.com/building-a-serverless-rest-api-with-node-js-and-mongodb-2e0ed0638f47 [1]

- Enter website: https://www.mongodb.com
- Create a free account
- Create a database

4.2. Setting up the AWS account

Example Guide:
https://serverless-stack.com/chapters/create-an-aws-account.html [2]
https://serverless-stack.com/chapters/create-an-s3-bucket-for-file-uploads.html [2]

- Create AWS account
- Create IAM User
- Config the AWS CLI
- Create a S3 bucket for image upload

4.3. Creating the serverless backend

Example Guide:
https://hackernoon.com/building-a-serverless-rest-api-with-node-js-and-mongodb-2e0ed0638f47 [1]

a. Installing the Serverless Framework

Run: `npm install -g serverless`

b. Creating the backend project

- Open Terminal
- Clone the project: `git clone https://github.com/HoangLong256/bookstore-backend.git`
- Move to the project folder and run: `npm install`
- Repace your database config URL in the **variables.env** file.

4.4. Deploying the backend

Example Guide:
https://serverless-stack.com/chapters/deploy-the-apis.html [2]

a. Deploy backend

- In the project folder, run: `serverless deploy`

b. Secure backend with AWS Cognito

- Based on the example, create the Cognito Identity Pool
- Save the config keyword in a note for later usage.

4.5. Creating the frontend

Example Guide:
https://serverless-stack.com/chapters/create-a-new-reactjs-app.html [2]

- Open Terminal

- Clone the frontend project: git clone https://github.com/HoangLong256/bookstore-frontend.git
- Navigate to the folder, run: npm install
- Replace the config setting in the **src/config.js**
- Run the frontend local: npm run start

4.6. Deploying the frontend

Example Guide:
https://medium.com/dailyjs/a-guide-to-deploying-your-react-app-with-aws-s3-including-https-a-custom-domain-a-cdn-and-58245251f081 [3]
https://serverless-stack.com/chapters/deploying-a-react-app-to-aws.html [2]

- In the project folder, run: npm run build
- Create a S3 bucket
- Set static hosting in S3 bucket
- Upload all files in build folder to S3 bucket

4.7. Create a CloudFront Distribution

Example Guide:
https://serverless-stack.com/chapters/create-a-cloudfront-distribution.html [2]

- In AWS Console, move to CloudFront Service
- Create a distribution
- Copy the URL in S3 bucket to the domain name
- Set up the error page to solve the Single Page Application problem

4.8. Set up a domain with CloudFront and Route 53

Example Guide:
https://medium.com/dailyjs/a-guide-to-deploying-your-react-app-with-aws-s3-including-https-a-custom-domain-a-cdn-and-58245251f081 [3]
https://serverless-stack.com/chapters/setup-your-domain-with-cloudfront.html [2]

- Register a domain in AWS Route 53 Service
- Request a certificate in Certificate Manager Service
- Add alternative domain to CloudFront Distribution
- Point a domain to CloudFront

5. User Manual

5.1. Customer

To purchase a product, the customer will do the following steps:

- Visit the website
- Go to Store tab

- Select a favourite book and click the View Button
- Select the quantity and click the Order button
- Go to the My Bag tab
- Review the orders and click Check Out button
- Enter the information details
- Click the Place Your Order button

5.2. Admin

As the admin, the owner has a right to view as well as edit the product and order by following steps

- Visit the website
- Click the Login tab
- Enter the username and password
- Click the Login button
- Go to the Product or Order tab for database management purposes

6. References

[1]A. Rahic, "Building a Serverless REST API with Node.js and MongoDB", *Hackernoon.com*, 2019. [Online]. Available: https://hackernoon.com/building-a-serverless-rest-api-with-node-js-and-mongodb-2e0ed0638f47. [Accessed: 17- May- 2020].

[2]"Serverless Stack", *Serverless Stack*, 2020. [Online]. Available: https://serverless-stack.com. [Accessed: 17- May- 2020]

[3]A. Bestbier, "How to deploy your React App with AWS S3", *Medium*, 2019. [Online]. Available: https://medium.com/dailyjs/a-guide-to-deploying-your-react-app-with-aws-s3-including-https-a-custom-domain-a-cdn-and-58245251f081. [Accessed: 17- May- 2020].