Client-Server Architectures 5COSC022W

Module Leader: Mr. Cassim Farook

Group Number: L5 CS G06

Name: S S U Sachintha Chamod Piyathunga

IIT ID: 20221948

UOW ID: w2053013

Abstract

This report focuses on the conceptualization, design, and implementation of a RESTful API system undertaken as part of the "Client-Server Architectures" module led by Mr. Cassim Farook. It examines the systematic application of client-server architecture principles to address a real-world scenario, including the formulation of API endpoints, data modeling, and a detailed analysis of server-client interactions and constraints. The report emphasizes the importance of maintaining security, scalability, and usability in distributed systems while aligning with the module's learning objectives. By applying methodologies taught throughout the module, the report presents a robust and comprehensive "BookstoreAPI" tailored to meet the specific requirements of the case study.

Acknowledgement

I would like to express my heartfelt gratitude to Mr. Cassim Farook, our module leader, for his invaluable guidance and support throughout the "Client-Server Architectures" module. His expertise and encouragement have been pivotal in deepening my understanding of client-server principles and fostering my critical thinking skills.

I would also like to extend my thanks to my university, Informatics Institute of Technology (IIT), for providing an excellent learning environment and resources that made the development of the “BookstoreAPI” project possible.

Lastly, I am deeply grateful to my family for their unwavering support, encouragement, and belief in my abilities. Their constant motivation has been a cornerstone of my academic journey.

Declaration

I hereby declare that this report is my original work and will be submitted as part of my individual coursework for the **Client-Server Architectures** module. I confirm that all sources of information used in this report have been properly acknowledged and cited, and no part of this work has been submitted elsewhere for academic or professional purposes.

Student Name	: Silpadhipathi Senarath Upalige Sachintha Chamodh Piyathunga
UoW ID Number	: w2053013
IIT ID Number	: 20221948

Signature :

Introduction

This report documents the test cases for the “MY-Bookstore” RESTful API developed using JAX-RS (Java API for RESTful Web Services) with JSON data format. The API implements CRUD operations for managing books, authors, customers, shopping carts, and orders. Testing was conducted using Postman to verify endpoint functionality, error handling, and response formats. This report focuses on documenting test scenarios and their outcomes in a structured tabular format.

API Endpoint Test Case Tables

Author Resource Endpoints						
POST /authors						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
POST /authors	Create valid author	POST	{ "name":"Robert Martin", "biography":"Software Engineer" }	201 Created	Author JSON	Pass
POST /authors	Invalid data	POST	{ "name":"" }	400 Bad Request	Error JSON	Pass
GET /authors						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result
GET /authors	Get all authors	GET	-	200 OK	[{author1},	Pass

					{author2}]	
GET /authors/{id}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /authors/1	Get valid author	GET	-	200 OK	Author JSON	Pass
GET /authors/999	Get invalid author	GET	-	404 Not Found	{ "error":"Author Not Found" }	Pass
PUT /authors/{id}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
PUT /authors/1	Update author	PUT	{ "biography":"Updated bio" }	200 OK	Updated author JSON	Pass
PUT /authors/1	Invalid data	PUT	{ "name":null }	400 Bad Request	Error JSON	Pass
PUT /authors/999	Update Invalid author	PUT	{ ... }	404 Not Found	{ "error":"Author Not Found" }	Pass
DELETE /authors/{id}						

Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
DELETE /authors/1	Delete valid author	DELETE	-	204 No Content	-	Pass
DELETE /authors/999	Delete invalid author	DELETE	-	404 Not Found	{ "error":"Author Not Found" }	Pass
GET /authors/{id}/books						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /authors/1/books	Get author's books	GET	-	200 OK	[{book1}, {book2}]	Pass
GET /authors/999/books	Invalid author ID	GET	-	404 Not Found	Error JSON	Pass
Book Resource Endpoints						
POST /books						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
POST /books	Create book successfully	POST	{ "title":"Clean Code", "authorId":1, "isbn":"9780132350884",	201 Created	Book JSON with	Pass

			"publicationYear":2008, "price":35.99, "stockQuantity":10 }		generated ID	
POST /books	Fail with invalid data	POST	{ "title":null, "price":-10 }	400 Bad Request	Error JSON	Pass
GET /books						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /books	Get all books	GET	-	200 OK	[{book1}, {book2}, ...]	Pass
GET /books/{id}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /books/1	Get book by valid ID	GET	-	200 OK	Complete book JSON	Pass
GET /books/999	Get book by invalid ID	GET	-	404 Not Found	{ "error":"Book Not Found" }	Pass
PUT /books/{id}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)

PUT /books/1	Update book successfully	PUT	{ "price":39.99 }	200 OK	Updated book JSON	Pass
PUT /books/1	Invalid ID	PUT	{ "price":39.99 }	400 Bad Request	{ "error":"Book Not Found" }	Pass
PUT /books/999	Non-existing ID	PUT	{ "price":20 }	404 Not Found	{ "error":"Book Not Found" }	Pass
DELETE /books/{id}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
DELETE /books/1	Delete valid book	DELETE	-	204 No Content	-	Pass
DELETE /books/999	Delete invalid book	DELETE	-	404 Not Found	{ "error":"Book Not Found" }	Pass
Customer Resource Endpoints						
POST /customers						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)

POST /customers	Create valid customer	POST	{ "name":"John Doe", "email":"john@test.com", "password":"pass123" }	201 Created	Customer JSON	Pass
POST /customers	Invalid email	POST	{ "email":"invalid-email" }	400 Bad Request	Error JSON	Pass
GET /customers						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /customers	Get all customers	GET	-	200 OK	[{customer 1}, {customer 2}]	Pass
GET /customers/{id}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /customers/1	Get valid customer	GET	-	200 OK	Customer JSON	Pass
GET /customers/999	Get invalid customer	GET	-	404 Not Found	{ "error":"Customer Not Found" }	Pass
PUT /customers/{id}						

Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
PUT /customers/1	Update customer	PUT	{ "email":"new.email@domain.com" }	200 OK	Updated customer JSON	Pass
PUT /customers/1	Invalid data	PUT	{ "password":"" }	400 Bad Request	Error JSON	Pass
PUT /customers/999	Update Invalid customer	PUT	{ ... }	404 Not Found	{ "error":"Customer Not Found" }	Pass
DELETE /customers/{id}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
DELETE /customers/1	Delete valid customer	DELETE	-	204 No Content	-	Pass
DELETE /customers/999	Delete invalid customer	DELETE	-	404 Not Found	{ "error":"Customer Not Found" }	Pass
Cart Resource Endpoints						

POST /customers/{id}/cart/items						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
POST /customers/ 1/cart/items	Add valid item	POST	{ "bookId":1, "quantity":2 }	201 Created	-	Pass
POST /customers/ 1/cart/items	Invalid data	POST	{ "bookId":1, "quantity":100 }	400 Bad Request	{ "error":"Invalid Input" }	Pass
GET /customers/{customerId}/cart						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /customers/ 1/cart	Get valid cart	GET	-	200 OK	Cart JSON	Pass
GET /customers/ 999/cart	Invalid customer	GET	-	404 Not Found	{ "error":"Cart Not Found" }	Pass
PUT /customers/{customerId}/cart/items/{bookId}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
PUT /customers/ 1/cart/items /1	Update quantity	PUT	{ "quantity":3 }	200 OK	Updated cart JSON	Pass

PUT /customers/1/cart/items/999	Invalid item	PUT	{ "quantity":1 }	404 Not Found	{ "error":"Item Not Found" }	Pass
DELETE /customers/{customerId}/cart/items/{bookId}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
DELETE /customers/1/cart/items/1	Delete item	DELETE	-	200 OK	-	Pass
DELETE /customers/1/cart/items/999	Delete invalid item	DELETE	-	404 Not Found	{ "error":"Item Not Found" }	Pass
Order Resource Endpoints						
POST /customers/{id}/orders						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
POST /customers/1/orders	Place order	POST	-	201 Created	Order JSON	Pass
POST /customers/1/orders	Empty cart	POST	-	400 Bad Request	{ "error":"Cart is empty" }	Pass
GET /customers/{customerId}/orders						

Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /customers/ 1/orders	Get all orders	GET	-	200 OK	[{order1}, {order2}]	Pass
GET /customers/{customerId}/orders/{orderId}						
Endpoint	Description	HTTP Method	Request Body	Expected HTTP	Expected Response	Actual Result (Pass/ Fail)
GET /customers/ 1/orders/1	Valid order	GET	-	200 OK	Order JSON	Pass
GET /customers/ 1/orders/99 9	invalid order	GET	-	404 Not Found	{ "error":"O rder Not Found" }	Pass

Demo video link -
https://drive.google.com/file/d/1l35Yivq4pmGDpbZU0_MBdH424Bhy2akX/view?usp=sharing