# RevShop (Application Development)

Center of Excellence 10-May-2023

Version 1.0

Copyright ©: 2023, Revature.

# Table of Contents

Table of Contents	2
Application Overview	5
Core Functional Scope	3
Standard Functional Scope	
Definition of Done	2
Competency wise scoping	2
Non-Functional Expectations	

## **Application Overview**

The RevShop project aims to develop a secure, user-friendly, and versatile e-commerce application for both buyers and sellers. The core functionalities for buyers include browsing products, adding products to a cart, checkout, and payment processing. Sellers can add products, manage inventory, and fulfill orders. The project's completion will be demonstrated through a cloud-hosted working version, technical presentation, and associated diagrams.

## Core Functional Scope

### **Buyer user account:**

As a buyer, I should be able to:

- 1. Register on the platform.
- 2. Login into the application using email and password.
- 3. View product details including image, price, description, and user review.
- 4. Browse products by category or keywords.
- 5. Add or remove products from the cart and provide quantity.
- 6. Checkout and enter shipping and billing information.
- 7. Get email notifications when an order is placed.
- 8. View order history.
- 9. Review products.
- 10. Save the product as a favorite.
- 11. Make payment using the payment gateway.

#### Seller account:

As a seller, I should be able to:

- 1. Register as a seller with email, password, and business details.
- 2. Login into the application using email and password.
- 3. Manage inventory of products.
- 4. Add new products with price and description.
- 5. See placed orders.
- 6. Receive email notifications when a user places an order.
- 7. Provide discounted price along with the maximum retail price.
- 8. View product review.
- 9. Get web notifications when the product's quantity is less than the threshold. (Seller sets the threshold value).

# Standard Functional Scope

Registered users should be able to log in, change the password and request for a forgotten password (which will be sent to their registered email).

# Definition of Done

- Working application demonstration.
- Sharing the associates' code repo for technical evaluation with:
  - o ERD Diagram
  - o Architecture Diagram

# Competency wise scoping

Competency	Application Type	Expectations
Java SQL REST / Java SQL REST Unix /	Console Based Application	User Inputs:
Java SQL REST Gradle / C# SQL REST /		1. Ability to accept the user inputs
Python SQL / JavaScript SQL / JavaScript		from console
NoSQL		2. Providing recommended format
		in which the user to key in the
		inputs
		3. Validate the user given inputs for
		format and convert to appropriate
		type for application usage.
		System outputs:
		Use formatted outputs for better
		readability and understanding.
		E.g., currency and date values
		should be formatted well.
		2. Display the reports in the
		appropriate format such as tables
		etc.
		User Navigation:
		1. Provide a number-based menu
		items for the user to navigate for
		different use cases
		2. Handling user selections and
		providing appropriate screen /
		feature to the user
		Validation and Error Handling:
		1. Validate the user inputs for its
		types and format.
		Display functional related user
		messages (either for
		input/error/output) - no system
		error codes.
		3. Handle the exceptions and errors
		gracefully.

	T	T
		<ol> <li>Logging:         <ol> <li>Ensure the application is using proper logging framework and methods.</li> <li>Ensure the application's log level is configured using configuration files so that it can be changed without changing the code.</li> <li>Also ensure that the application logging is configured to output to the mentioned log file.</li> </ol> </li> </ol>
		Testing:  1. Ensure sufficient test cases are written using appropriate testing frameworks.  2. Ensure the code coverage closed to be 80%
		Sagnitu
		Security:  1. Ensure the SQL/NoSQL injection threat is taken care.
		Coding Standard:  1. Use the industry coding standards and conventions.  2. Modular based code development for better reusability.  3. Ensure proper usage of resource objects such as database connectivity objects to avoid resource leakages.  4. Ensure proper usage of design patterns and application layering (such as Business Service, DAO Layer etc.) wherever applicable.
Web Fundamentals (HTML, CSS and JS)	Web Navigational Prototype	<ol> <li>User Experience:         <ol> <li>Have an intuitive design for the user to work with the application without any training or guidance</li> <li>Have clean &amp; consistent UI, color theme and easy to use navigations</li> <li>Use bootstrap framework for responsive pages</li> <li>Have proper tab indexing for users to navigate between the fields without usage of mouse.</li> </ol> </li> </ol>
		User Inputs & outputs:  1. Have appropriate HTML fields for the user inputs

		2 Whanson possible use the client
		2. Wherever possible use the client-
		side validations for the user input  3. Display the appropriate user
		info/error message with
		appropriate colors and icons
		арр. ор. насе селен селен
		Performance:
		1. Use compressed images / assets
		to increase the page performance
		2. Use the application validated
		using the Chrome's Lighthouse
		tool and improved based on the
		report
		Dataset:
		For any prepopulated data such
		as to render table rows use JSON
		file as the DataSource and use
		standard open-source library to
		read and render in the web pages.
		General standards:
		Ensure the w3 standards are
		implemented for better
		accessibility. E.g., Using alt
		attribute for image tag.
		2. Ensure the SEO recommended
		meta tags are added.
Web Development with Angular / Web	Enhanced Web Navigational Prototype	Same as Web Fundamental
Development with Angular JS/ Web Development with React / Web	using the selected framework (either Angular/React/Vue.JS)	Competency above, with the following additions:
Development with React Redux	/ Ingular/ react/ vac.35/	Tollowing additions.
		Framework Specific
		Ensure the appropriate APIs are
		used for any of the API calls
		2. Ensure the routing is centrally
		configured
		3. Best practices & design patterns are to be followed
		4. Implement the end-to-end testing
		framework and get to know the
		headless execution of end-to-end
		framework.
		Douboum out outfoot-
		Deployment artifacts:  1. The deployment artifacts should
		be minified and obfuscated if
		required.
		·
		Security:
		_
		Ensure the CORS restriction is applied, if applicable.

Spring Boot / ASP.NET API REST / Java SQL REST / Java SQL REST Unix / Java SQL REST Gradle / SpringBoot with Authentication & Messaging  Identifying and Developing All APIs for any front-end application to consume  REST Standard 1. Ensure the followed for Operation of Operation of Messaging  Proper do with Input be document.  Ensure the defined to the enabled for monitorin dashboard services has Use the repatterns to the enabled for monitorin dashboard services.  Modularity:  I. If the appl from an emproject, tremodules.  Refactor to on the cur  Logging:  Legging:  Ensure the curve the enabled for monitorin dashboard services.  Refactor to on the curve the c	
Spring Boot / ASP.NET API REST / Java SQL REST / Java SQL REST Unix / Java SQL REST Gradle / SpringBoot with Authentication & Messaging  Identifying and Developing All APIs for any front-end application to consume  REST Standard 1. Ensure the followed for Operation of Operation of Messaging  Proper do with Input be document.  Ensure the defined to the enabled for monitorin dashboard services has Use the repatterns to the enabled for monitorin dashboard services.  Modularity:  I. If the appl from an emproject, tremodules.  Refactor to on the cur  Logging:  Legging:  Ensure the curve the enabled for monitorin dashboard services.  Refactor to on the curve the c	
Spring Boot / ASP.NET API REST / Java as enviror secure cree  Spring Boot / ASP.NET API REST / Java SQL REST Unix / Java SQL REST Gradle / SpringBoot with Authentication & Messaging  Identifying and Developing All APIs for any front-end application to consume  In Ensure the followed of Operation definition of the second of the second definition	Authenticated Routing
Spring Boot / ASP.NET API REST / Java SQL REST / Java SQL REST Unix / Java SQL REST Gradle / SpringBoot with Authentication & Messaging  Identifying and Developing All APIs for any front-end application to consume  In Ensure the followed for Operation definition) Secure the secure the public and the defined to the consume of the consumer	
Spring Boot / ASP.NET API REST / Java SQL REST / Java SQL REST Unix / Java SQL REST Gradle / SpringBoot with Authentication & Messaging  Identifying and Developing All APIs for any front-end application to consume  1. Ensure the followed for operation definition) 2. Secure the 3. Define a or public and 4. Proper do with input be docum 5. Ensure the defined to 6. Provide / to route it single cha 7. Ensure the enabled for monitorin dashboard services (in the patterns to services)  8. Use the repatterns to services (in the patterns of the patterns of the patterns of the patterns of the proper log methods.  2. Refactor to on the cur  1. Ensure the followed for patterns to service in the proper log methods.  2. Ensure the followed for patterns to service in the proper log methods.  3. Ensure the followed for patterns to service in the proper log methods.  4. Proper do with input be docum 5. Ensure the proper log methods.  5. Ensure the followed for patterns to service in the proper log methods.  6. Ensure the proper log methods.  7. Ensure the proper log methods.  8. Ensure the proper log methods.  9. Ensure the proper log methods.  1. Ensure the followed for patterns to service in the proper log methods.  9. Ensure the proper log methods.  9. Ensure the followed for patterns to service in the followed for patterns to servic	at the secrets are stored
Spring Boot / ASP.NET API REST / Java SQL REST / Java SQL REST Unix / Java SQL REST Gradle / SpringBoot with Authentication & Messaging    Identifying and Developing All APIs for any front-end application to consume	ment variables using
SQL REST / Java SQL REST Unix / Java SQL REST Gradle / SpringBoot with Authentication & Messaging  any front-end application to consume  1. Ensure the followed for Operation definition) 2. Secure the defined to define the defined to define the single character of the construction of th	dential storage.
SQL REST / Java SQL REST Unix / Java SQL REST Gradle / SpringBoot with Authentication & Messaging  any front-end application to consume  1. Ensure the followed for Operation definition) 2. Secure the defined to define the defined to define the single character of the construction of th	<u>.</u>
SQL REST Gradle / SpringBoot with Authentication & Messaging  2. Secure the 3. Define a c public and 4. Proper do with Input be docum 5. Ensure the defined to for provide / to rouse ti single cha 7. Ensure the enabled for monitorin dashboard services h 8. Use the re patterns ti services (so  Modularity: 1. If the appl from an e project, tr modules 2. Refactor t on the cur  Logging: 1. Ensure the proper log methods. 2. Ensure the proper log methods. 2. Ensure the proper log methods. 3. Ensure the proper log methods. 4. Proper do public and publi	
Authentication & Messaging  Operation definition) 2. Secure the 3. Define a conception of the public and secure the public and secure the defined to compute the defined the defined to compute the defined the defi	
definition)  2. Secure the 3. Define a compublic and 4. Proper do with Input be docum 5. Ensure the defined to defined to to route it single chance. 7. Ensure the enabled formonitorin dashboard services h 8. Use the respective services (services)  Modularity: 1. If the apple from an emproject, transdules. 2. Refactor to not the cur  Logging: 1. Ensure the proper log methods. 2. Ensure the is configured. 3. Ensure the proper log methods. 4. Proper do with Input be document of the proper log methods. 5. Ensure the is configured.	or API naming, HTTP
2. Secure the 3. Define a c public and 4. Proper do with Input be docum 5. Ensure the defined to 6. Provide / to route the single cha 7. Ensure the enabled for monitorin dashboard services h 8. Use the repatterns to services (services (	and Response (output
3. Define a c public and 4. Proper do with Input be docum 5. Ensure the defined to 6. Provide / to route it single cha 7. Ensure the enabled for monitorin dashboard services h 8. Use the re patterns to services (services (serv	
public and 4. Proper do with Input be docum 5. Ensure th: defined to 6. Provide / to route it single cha 7. Ensure the enabled fe monitorin dashboard services h 8. Use the re patterns t services (v  Modularity: 1. If the appl from an e project, tr modules. 2. Refactor t on the cur  Logging: 1. Ensure the proper log methods. 2. Ensure the is configur files so the	protected APIs
with Input be docum  5. Ensure the defined to 6. Provide / to route the single cha 7. Ensure the enabled for monitorin dashboard services h 8. Use the re patterns to services (or  Modularity: 1. If the appl from an e project, tr modules. 2. Refactor t on the cur  Logging: 1. Ensure the proper log methods. 2. Ensure the proper log methods. 3. Ensure the proper log methods. 4. Ensure the proper log methods. 5. Ensure the proper log methods. 6. Ensure the proper log methods. 7. Ensure the proper log methods. 8. Ensure the proper log methods. 9. Ensure the proper log methods.	ommon URL pattern for secure APIs
be docum  5. Ensure the defined to to route it single cha  7. Ensure the enabled for monitorin dashboare services h  8. Use the re patterns to services (services (ser	cumentation of APIs
5. Ensure the defined to 6. Provide / to route it single cha 7. Ensure the enabled for monitorin dashboard services h 8. Use the repatterns to services (v. Modularity:  1. If the apple from an exproject, try modules.  2. Refactor try on the cur to on the cur to on the cur to on the cur to service the proper log methods.  2. Ensure the proper log methods.  2. Ensure the is configure files so the	and Output Samples to
6. Provide / to route the single change. 7. Ensure the enabled for monitorin dashboard services hand. 8. Use there are patterns to services (sometimes of the services of the	ented at each micro service is
to route the single change.  7. Ensure the enabled for monitoring dashboard services has Use the repatterns to services (services).  Modularity:  1. If the apple from an exproject, try modules.  2. Refactor try on the current on the current proper log methods.  2. Ensure the proper log methods.  2. Ensure the is configurent files so the single change.	do one job.
single cha 7. Ensure the enabled for monitorin dashboard services h 8. Use the re patterns tr services (v  Modularity: 1. If the appl from an e project, tr modules. 2. Refactor tr on the cur  Logging: 1. Ensure the proper log methods. 2. Ensure the is configur files so the	enable to API gateway
7. Ensure the enabled for monitorin dashboard services h 8. Use the repatterns to services (x  Modularity: 1. If the apple from an eproject, the modules. 2. Refactor to on the cur  Logging: 1. Ensure the proper log methods. 2. Ensure the is configurent files so the	ne request through a
enabled for monitorin dashboard services h  8. Use the repatterns to services (v  Modularity:  1. If the appliance of from an eproject, trepodules.  2. Refactor to on the cure on the cure on the cure of the proper log methods.  2. Ensure the proper log methods.  2. Ensure the is configurent files so the files of the cure of the	nnel
monitorin dashboard services h  8. Use the re patterns to services (v  Modularity:  1. If the appl from an e project, tr modules. 2. Refactor t on the cur  Logging: 1. Ensure the proper log methods. 2. Ensure the is configur files so the	e micro services are
dashboard services h  8. Use the repatterns to services (v  Modularity:  1. If the apple from an eproject, tremodules.  2. Refactor to nothe cur  Logging:  1. Ensure the proper logemethods.  2. Ensure the is configurent files so the services (v).	or traceability and
services h  8. Use the repatterns to services (v  Modularity:  1. If the apple from an exproject, to modules.  2. Refactor to on the cur  Logging:  1. Ensure the proper log methods.  2. Ensure the is configurent files so the control of the cur  1. Ensure the proper log methods.	g with appropriate
8. Use the repatterns to services (v  Modularity:  1. If the apple from an exproject, tresponders.  2. Refactor to on the cure.  Logging:  1. Ensure the proper log methods.  2. Ensure the is configurent files so the configurent files f	I to render the micro
patterns to services (v.)  Modularity:  1. If the apple from an exproject, to modules.  2. Refactor to on the current continue to proper log methods.  2. Ensure the proper log methods.  2. Ensure the is configurent files so the continue to the services (v.)	ealth and logging.
Modularity:  1. If the apple from an exproject, tree modules.  2. Refactor tree on the cure to on the cure the proper log methods.  2. Ensure the is configurent files so the cure the proper log methods.	commended design
Modularity:  1. If the apple from an exproject, transcribed in the current on the	o aggregate the micro
1. If the apple from an exproject, transcription on the cur  Logging:  1. Ensure the proper log methods.  2. Ensure the is configuration of the cur  1. Ensure the proper log methods.	wherever is applicable)
1. If the apple from an exproject, transcription on the cur  Logging:  1. Ensure the proper log methods.  2. Ensure the is configuration of the cur  1. Ensure the proper log methods.	
from an e project, tr modules.  2. Refactor t on the cur  Logging:  1. Ensure the proper log methods.  2. Ensure the is configur files so the	ication is enhanced
project, tr modules.  2. Refactor t on the cur  Logging:  1. Ensure the proper log methods.  2. Ensure the is configur files so the	xisting console-based
modules.  2. Refactor ton the cur  Logging:  1. Ensure the proper log methods.  2. Ensure the is configurent files so the	y to reuse the existing
2. Refactor to on the cur  Logging:  1. Ensure the proper log methods.  2. Ensure the is configure files so the	/ to reuse the existing
Logging:  1. Ensure the proper log methods.  2. Ensure the is configure files so the	he existing code based
<ol> <li>Ensure the proper log methods.</li> <li>Ensure the is configuration of the proper log methods.</li> </ol>	rent tech stacks
<ol> <li>Ensure the proper log methods.</li> <li>Ensure the is configuration of the proper log methods.</li> </ol>	
proper log methods.  2. Ensure the is configur files so the	application is using
methods.  2. Ensure the is configure files so that	ging framework and
is configur files so that	
is configur files so that	application's log level
files so that	ed using configuration
	at it can be changed
without d	nanging the code.
	re that the application
	configured to output to
	oned log file.
	e centralized logging
implemen	
implemen	tation

		Testing:  1. Ensure sufficient test cases are written using appropriate testing frameworks.  2. Ensure the code coverage closed to be 80%  Security:
		<ol> <li>Ensure the CORS restriction is applied.</li> <li>SQL Injection thread is taken care</li> <li>Throttling is to be taken care of.</li> <li>Cross site Scripting to be avoided.</li> <li>Ensure that the secrets are stored as environment variables using configuration files or secure credential storage.</li> </ol>
Devops .NET Azure GHA/Devops(CI/CD) Azure Jenkins/Devops(CI/CD) AWS Jenkins	Implementing DevOps Practices and CI/CD Pipelines for Cloud-Based Applications	<ol> <li>Ensure to implement DevOps best practices, emphasizing collaboration, automation, and continuous improvement throughout the software development and deployment lifecycle.</li> <li>Leverage cloud services promptly to meet specific application hosting and deployment needs.</li> <li>Ensure consistent and repeatable resource deployment by applying Infrastructure as Code (IaC) principles to define and provision cloud resources programmatically.</li> <li>Implement cloud-specific security measures to protect applications and data.</li> <li>Create and manage CI/CD pipelines using Jenkins or GitHub Actions to automate building, testing, and deploying applications.</li> <li>Implement pipelines in either scripted or declarative form to streamline the software delivery process.</li> </ol>

# Non-Functional Expectations

- o Application development supposed to follow the Scrum process
- o Application password should be encrypted using appropriate hashing algorithms
- o Applications should use the recommended authentication token such as JWT or other equivalent