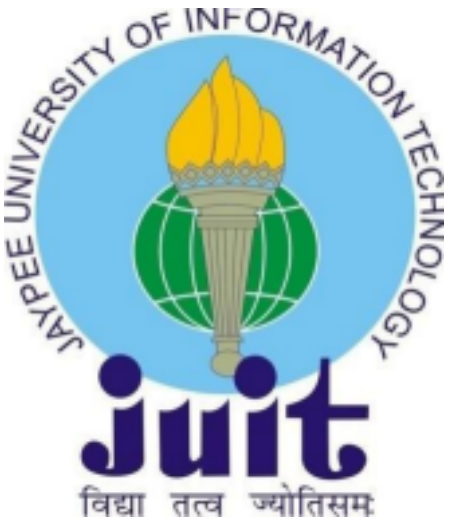
E-commerce Web API Testing and Automation

Minor project report submitted in partial fulfillment of
the requirement for the degree of Bachelor of
Technology in
Computer Science and Engineering

By
Jaspreet Singh 181477
Avirukh Dahiya 181462

UNDER THE GREAT SUPERVISION OF

Dr. Amit Kumar



Department of Computer Science & Engineering and
Information Technology

**Jaypee University of Information Technology, Waknaghat,
173234, Himachal Pradesh, INDIA**

TABLE OF CONTENT

Title Page No.

DECLARATION

We hereby declare that this project has been done by me under the supervision of Dr. Amit Kumar of the Jaypee University of Information Technology. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:

Dr. Amit Kumar
Associate Professor
Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology

Submitted by:

Jaspreet Singh
181477
Avirukh Dahiya
181462
Computer Science & Engineering Department
Jaypee University of Information Technology

CERTIFICATE

This is to certify that the work which is being presented in the project report titled "**Ecommerce Web API Testing and Automation** " in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science And Engineering and submitted to the Department of Computer Science And Engineering, Jaypee University of Information Technology, Waknaghat is an authentic record of work carried out by **Jaspreet Singh(181477),Avirukh Dahiya(181462)** during the period from January 2022 to May 2022 under the supervision of Dr. Amit Kumar, Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat.

The above statement made is correct to the best of our knowledge.

Dr. Amit Kumar
Assistant Professor(SG)
Computer Science & Engineering and Information Technology Jaypee University of Information Technology, Waknaghat

ACKNOWLEDGEMENT

Firstly, We express our heartiest thanks and gratefulness to Almighty God for His divine blessing that makes it possible to complete the project work successfully.

We are grateful and wish our profound indebtedness to Supervisor Dr. Amit Kumar, Associate Professor, Department of CSE Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of our supervisor in the field of Deep Learning carrying out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

We would like to express my heartiest gratitude to Dr. Amit Kumar, Department of CSE, for his kind help to finish our project.

We would also generously welcome each one of those individuals who have helped us straightforwardly or in a roundabout way in making this project a win. In this unique situation, We might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, We must acknowledge with due respect the constant support and patience of our parents.

Avirukh Dahiya
Jaspreet Singh

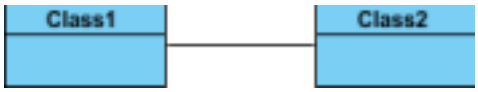
NOTATION

1) USE-CASE DIAGRAM

Actor: Someone interacts with use case with one another.

Include: The stereotype "<<include>>" identifies the relationship as an include relationship.

2) CLASS DIAGRAM



Simple Association: A structurallink between two peer classes.

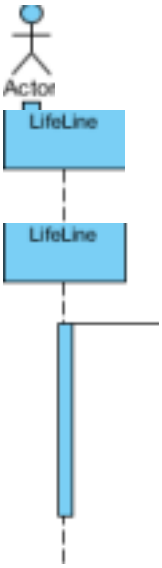
Composition: Objects of Class2 liveand die with Class1.

Use Case: System function

Dependency: Class1 depends on Class2.

Communication Link: Indicating that the actor andthe use case can communicate

3) SEQUENCE DIAGRAM

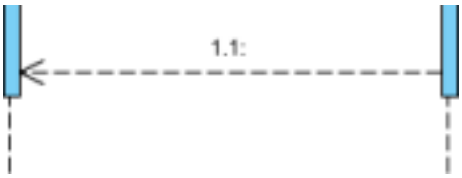


Actor: represent roles played by human users, external hardware, or other subjects.

Lifeline: A lifeline represents an individual participant in the Interaction.

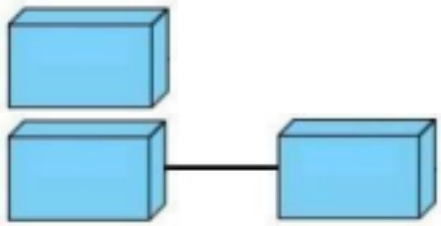
Activations: A thin rectangle on a lifeline) represents the period during which an element is performing an operation.

Call Message: Call message is a kind of message that represents an invocation of operation of target lifeline.



Return Message: Return message is a kind of message that represents the pass of information back to the caller of a corresponded former message.

4) DEPLOYMENT DIAGRAM



Nodes: 3-D box represents a node,either software or hardware.

Association: Indicates interaction between nodes.

TRAINING TAKEN AT INDUSTRY

OBJECTIVE AND SCOPE OF TRAINING

- The key objective of the training was to take an experience of how industry works.
- I learned various training tools and concepts in Java.
- I learned how to write clean code with the appropriate docs which works in the industry.

DURATION AND SCHEDULE OF TRAINING

The duration of training for tools and concepts of Java was 15-16 weeks.
The training started from 7th February 2022.

SOFTWARE TRAINING WORK UNDERTAKEN

Tools Training : Git, GitHub, Intellij, Docker, PostgreSQL
Concept Training : OOP, DBMS, Java, Spring Boot

CONCLUSION AND FEEDBACK

- The training of tools and concepts was very interesting and helpful in respect to industry. ● I learned how to write the clean and tidy code with appropriate unit test cases as well as the best practices for making a Rest API/Service. I learned how to write docs for the code written so it can be readable.
- I also learned the version control tools like Git and its commands, also learned GitHub and was used to it.

1. INTRODUCTION

1.1 PROJECT DETAILS

Project provides an API for the E-commerce Site which has functionality for viewing products, adding a product to their cart. API has functionality of registration & login for Admin as well as Customers. API provides functionality of Admin side where he can add/update/delete the products.

1.2 PURPOSE

This Web API serves the purpose of fetching as well as adding data to the Database from UI or any API testing platform.

1.3 SCOPE

Currently this Web API has some sort of Security mechanism of Login but it can be more secured. And also fields used by the big E-commerce platforms are missing in Web API.

1.4 OBJECTIVE

This Web API allows Customer/Client to Login, view products, add/remove the products to/from the cart.

For Admin, Web API has Login, view products, view categories, add/update/delete a product/category and view users in Site.

2. Context Diagram

Context diagram

Context diagram

3. SCOPE STATEMENT

The Ecommerce Web API is an API which easily creates/updates/delete products as well as User Authentication for safety.

In everyday life there are many difficulties we are facing. In this difficulties one of the difficulty is to manage store/ecommerce site as well as in this hectic life schedule we don’t have time to go for getting the products from the shops.

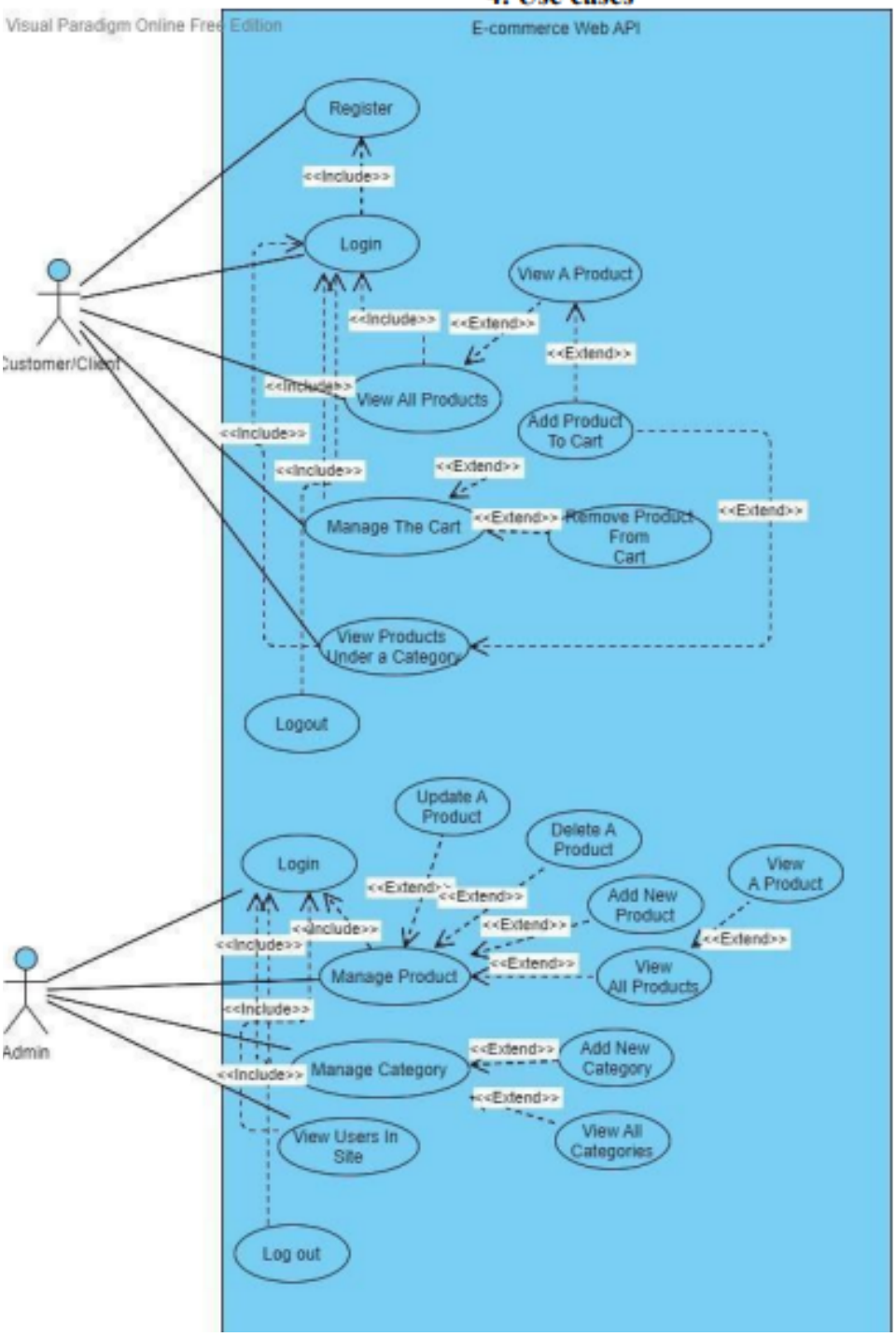
In my Web API, I have created a backend of the E-commerce site which has Admin – who can manage the site, who can manage the products & users & Users – who can view products as well as add them to the cart. Now this Web API may work with any sort of UI without any prior knowledge of building/structuring of the API.

Apart from this Web API has UI friendly responses which can be easily utilized at the time of creation of the UI as well as it is also user friendly for users when used the API for testing in the postman. Also Web API is secure with basic sort of Authentication.

Currently this Web API has some sort of Security mechanism of Login but it can be more secure. And also fields used by the big E-commerce platforms are missing in Web API.

4. Use cases

4. Use cases



1) Customer/User/Client:

R1: Login:

a) Description : User will have to register for login and see the products and use

add to cart functionality

b) Pre condition : The user coming for first time after login
c) Input : Email, Password
d) Processing : Confirms password and email given by user.
e) Output : User can access the above functionality.

R2: Register:

a) Description : User have to register with required details.
b) Input : Email, username, password.
c) Processing : Creates new user and add to database.
d) Output : Success if created.

R3: View Products:

R3.1: View all Products

a) Description : If user login so he/she can view all the products present in database.
b) Pre condition : The user must be login.
c) Processing : Gets all product present in database.
d) Output : List of products.

R3.2: View Products under a Category

a) Description: if user login then can view products under a category.
b) Precondition : the user must be login.
c) Input : category name/id.
d) Processing: gets all product under one category.
e) Output : List of products under one

category. R4: Cart Management:

R4.1: Add Product to Cart

a) Description : if user login then can add a product to a cart.
b) Pre condition : the user must be login.
c) Input : product to be inserted in cart.
d) Processing : saves product in cart of that user.
e) Output : product added to

cart R4.2: View Cart

a) Description : Shows products in the cart.

b) Processing : Gets the list of products in the cart.
c) Out Put : Shows products present in user's cart.

R4.3: Remove Product From Cart

a) Description : if user login then can remove a product from the cart.
b) Pre condition : the user must be login.
c) Input : product to be removed from cart.
d) Processing : deletes product from cart of that user & decreases total price & quantity if more than one product is added.
e) Output : product removed from cart

2) Admin:

R5: Login:

a) Description : Admin login to get access of admin.
b) Input : Email, password.
c) Processing : Verify email and password in database.
d) Post Condition : Access to admin functionality.

R6: View Users:

a) Description : View Users connected to the site.
b) Processing : Gets list of users present in database.
c) Output : list of users.

R7: Product Management:

R7.1: Add new Product:

a) Description : Adds a new product with details of that products.
b) Input : Product's id, name, price, category, description, quantity.
c) Processing : saves the new product with the above details

R7.2: View a Product:

a) Description : View a particular product by its id.
b) Input : Product's Id.
c) Processing : gets the product with the id.
d) Output : Product with the input id.

R7.3: Update Product:

a) Description : Update the product with its id.
b) Input : Product's details to be updated.

c) Processing : updates the product and saves that to the database.
d) Output : success if saved.

R7.4: Delete Product:

a) Description : Delete the product with its id.
b) Input : Product's id to be deleted.
c) Processing : removes that product from the database.

R8: Category Management:

R8.1: Add new Category:

a) Description : Adds new Category to the database.
b) Input : Category's Id, name.
c) Processing : saves the category in database.
d) Output : Success if saved.

R8.2: View All Categories:

a) Description : View list of all the categories in the site. b) Processing : Gets list of all categories present in database. c) Output : list of categories. R8.3:

View Products Under categories:

a) Description : can view products under a category.
b) Input : category name/id.
c) Processing : gets all product under one category.
d) Output : List of products under one category.

5. SYSTEM DESIGN

FEASIBILITY STUDY

Technical Feasibility:

- A technical feasibility study evaluates the details of how you intend to build a system or solution. Technical analysis evaluates technical merits of the system and at the same time collects additional information about performances, reliability, maintainability and productivity. We have analyzed feasibility studies of our project in this phase.
- Basically, this study is to ensure that proposed system software is not a loss or burden to the user. It assesses the practicality of the proposed project. For this study, core knowledge of the system is required.

Time Schedule Feasibility:

- The Project has simple working and basic requirement can be satisfied with the allotted time period.

Operational Feasibility:

- The proposed system will increase the operational efficiency of all users. Therefore, the throughput of the system and reduction of time is as desired.
- This system will ensure and satisfy the requirements identified by the requirement analysis phase of the system.
- The API will be user-friendly and will work with any frontend system for the same definition i.e. it will be very useful and easy to operate.

Implementation Feasibility:

- User can directly access API with the Postman or any API testing software or also can create the frontend and API can be useful for the same.
- Implementation will result in all the important information about all the available professionals and customers can hire any of them for their personal activities/needs.

PROJECT PLANNING

Project Development Approach & Justification:

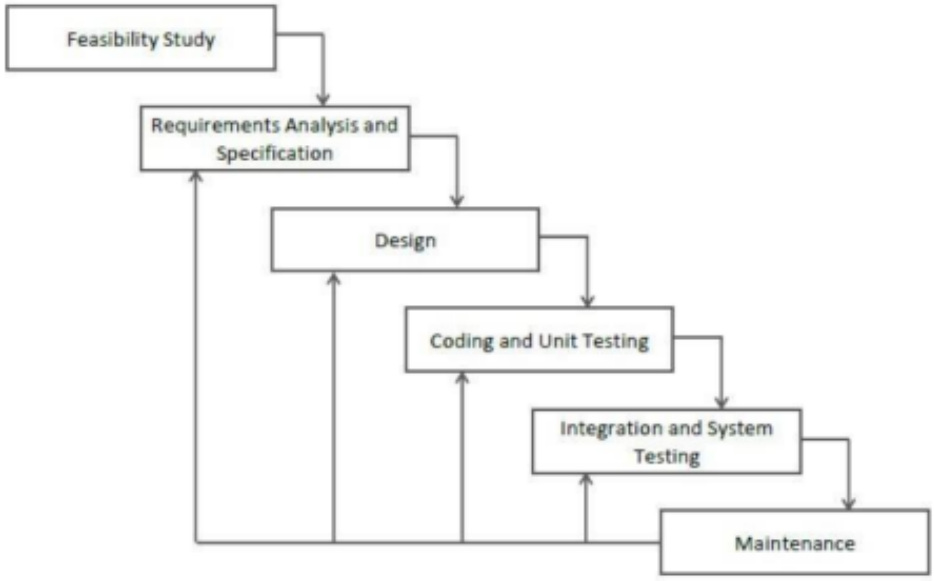
- For project improvement, the Iterative cascade model is utilized. It is a specific execution of a product improvement life cycle that centers around an underlying, improved on execution, which then, at that point, continuously acquires intricacy and a more extensive list of capabilities until the last framework is finished.

Iterative Waterfall:

The Iterative water fall model methodology conquers the issues related with the cascade model methodology. Assuming that any trouble or issue experience in any stage might require returning to the past stage and playing out the expected alterations and continues consecutively. This backtracking permits altering any adjustments or changes expected in the past stage.

This model partitions the cycle into the stages referenced underneath:

1. Feasibility Study
2. Requirement analysis and specification
3. Design
4. Coding and unit testing
5. Integration and system testing
6. Maintenance



Iterative waterfall model

Iterative waterfall model

Justification:

After feasibility study the functional requirements are almost clear, but in some cases like implementing, designing as well as in testing time error can occur. Here we have to decompose the system into modules. That is why we
decide to use an iterative waterfall model which is most suitable model here i.e. if we find any difficulty in coding and testing a modification in design can be done easily.

Milestones and Deliverables

– Got Project Definition
– Learned Appropriate Technology/Libraries.
– Complete Requirement Gathering and requirement specifications.
– Start with the Implementation phase.
– Complete with Implementation and test the API as well as trying to add some new/additional functionalities.
– Complete Documentation of Project.

Roles and Responsibilities

Roles and Responsibilities of Group Members

Name	Role			
	Analysis	**Prototype Designing**	**Development of system**	**Testing Documentation**
Jaspreet Singh Avirukh Dahiya	✔	✔	✔	✔ ✔

Group Dependencies:

The dependencies among the tasks include the following:

- Analysis or System Requirement Study (SRS) is independent of all, yet will be started after completion of feasibility study and project planning.
- Designing of prototypes can be done simultaneously with system analysis.
- Development of the project is preceded by the designing of prototype and system analysis.
- Testing can be only done once the development of some major functionalities are completed and are ready to be tested.
- Documentation is independent of all the tasks and can be done as the other tasks proceed.

Project Scheduling

Project Scheduling

ID	Task Duration
1	System Requirements, Analysis & Project Planning 2 Week
2	System Design Layout 3 Week
3	Design 4 Week
4	Coding 4 Week

5	Testing 3 Week
6	Documentation 1 Week

STUDY OF CURRENT SYSTEM

In the current system the Admin has to manually add/update/remove a product from his shop. The admin will have to put all the products in some categories. And for customers, they have to go to shop for the shopping and get the product they want.

USER CHARACTERISTICS

User Characteristics

Sr. no.	Role	Features
1	User/Client	Authentication, cart management, product viewing.
2	Admin	Authentication, product management, categories management.

HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements:

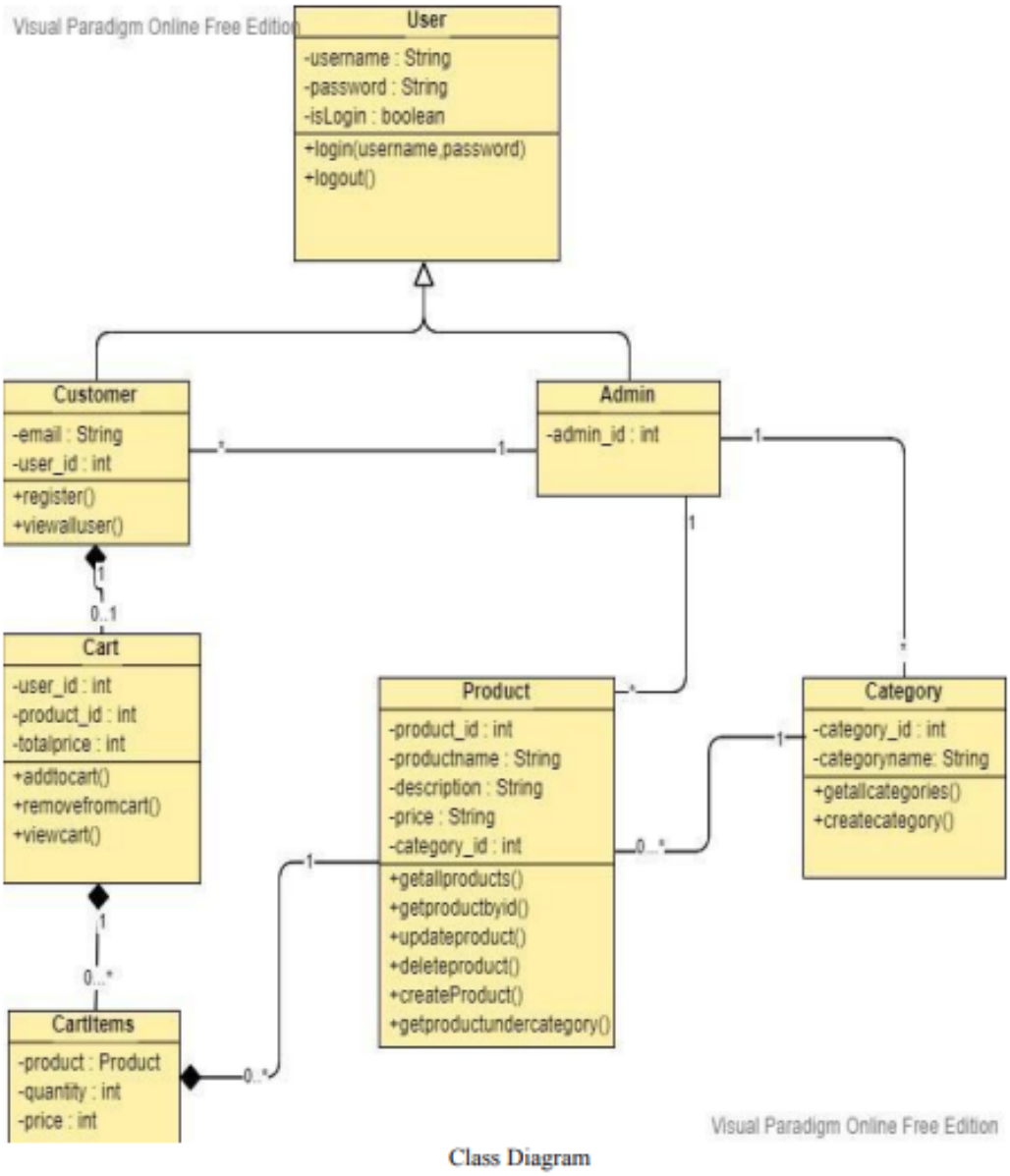
- Microsoft® Windows® 7/8/10.
- 2 GB RAM minimum, 8 GB RAM recommended.
- 1280 x 800 minimum screen resolution.

Software Requirements:

- Docker Desktop/CLI
- Java (JDK8 or more)
- Spring Boot
- PostgreSQL
- Git/GitHub
- IDE : Intellij IDE

SYSTEM ARCHITECTURE DESIGN

Class Diagram:



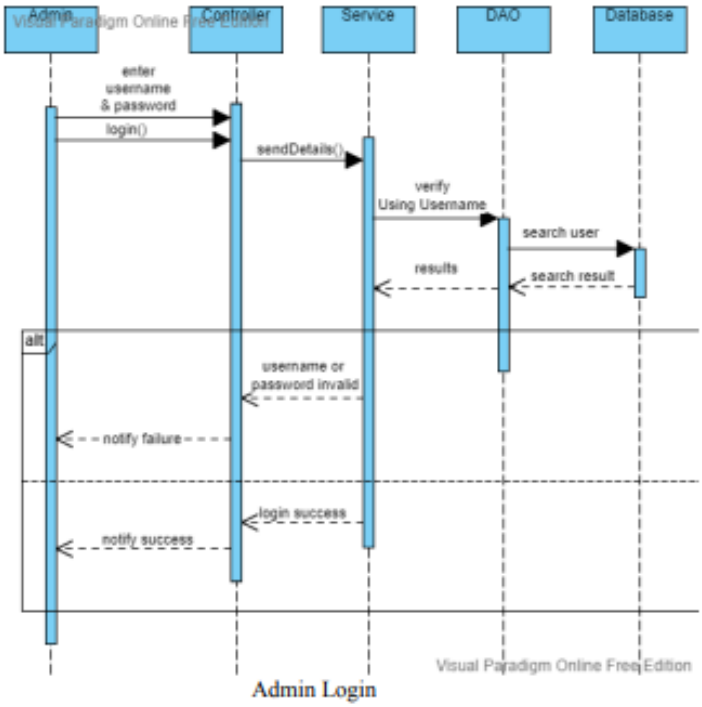
Class Diagram

Class Diagram

Sequence Diagram:

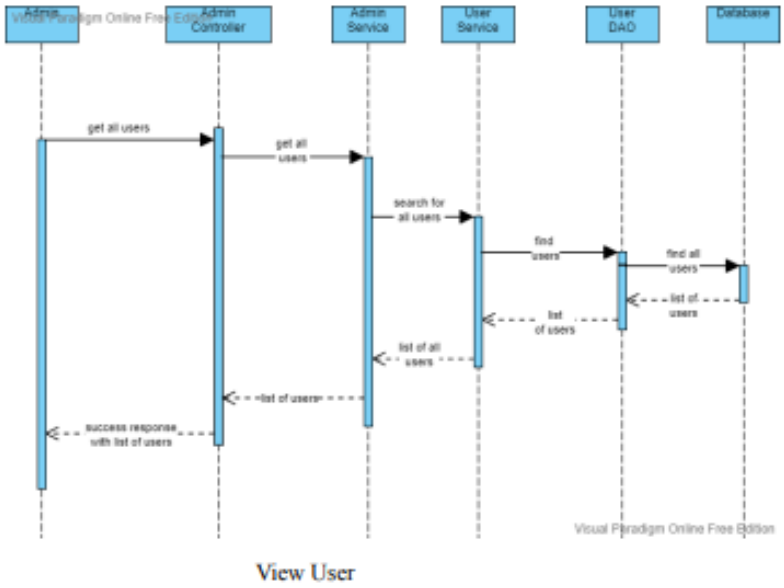
1) Admin:

a) Login:



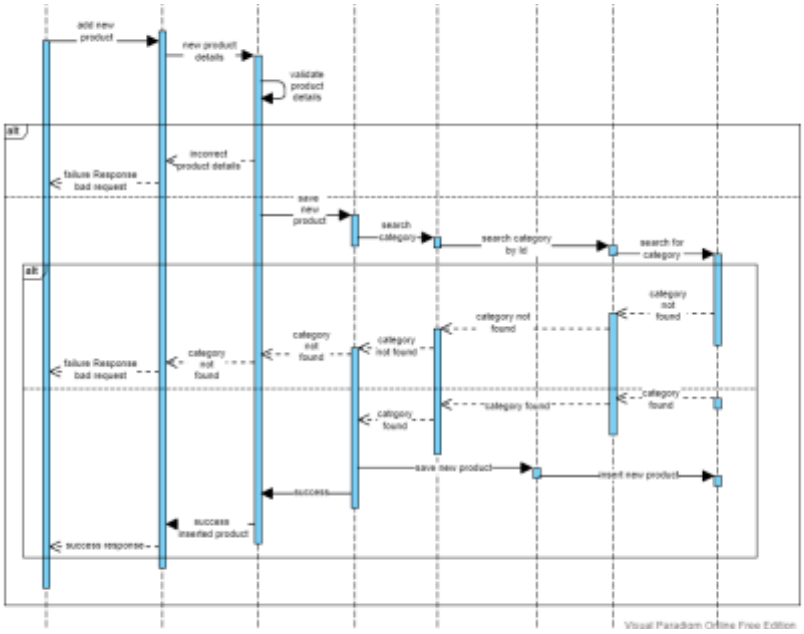
Admin Login

b) View Users:



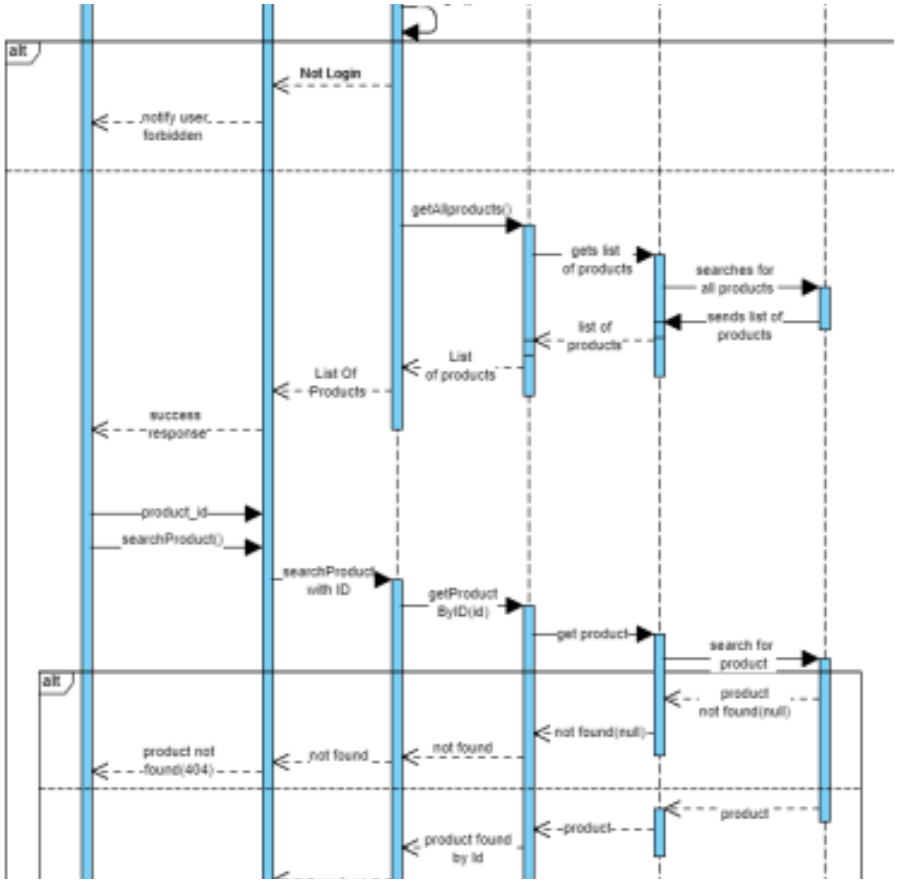
View User

c) View Categories:



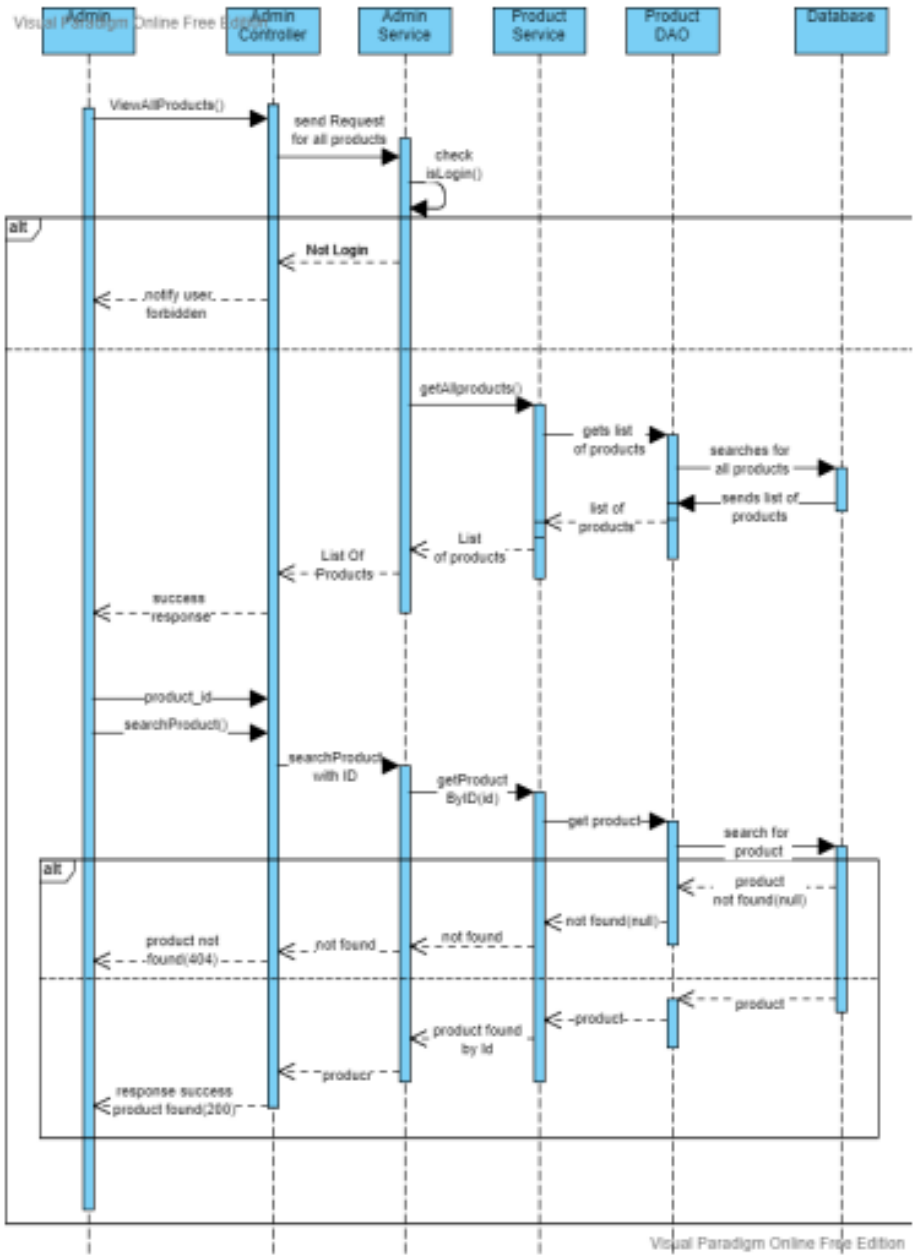
View categories

d) Add a Product:



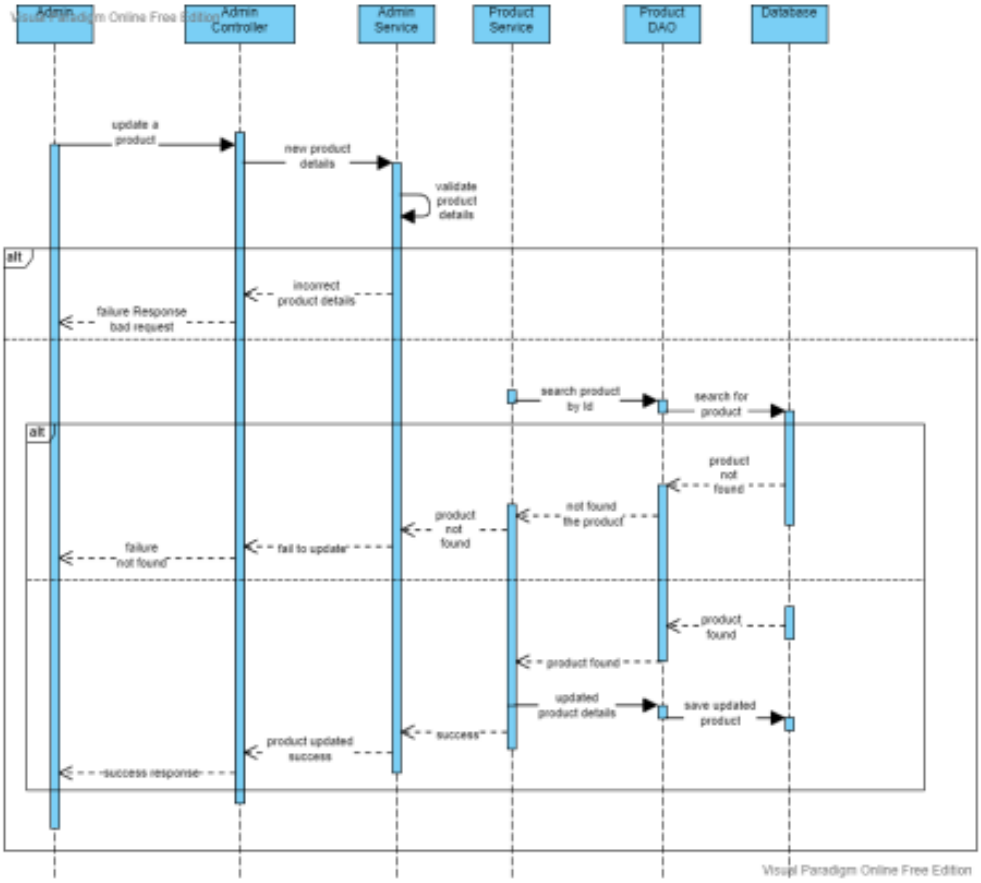
Add Product

e) View Product:



View Product

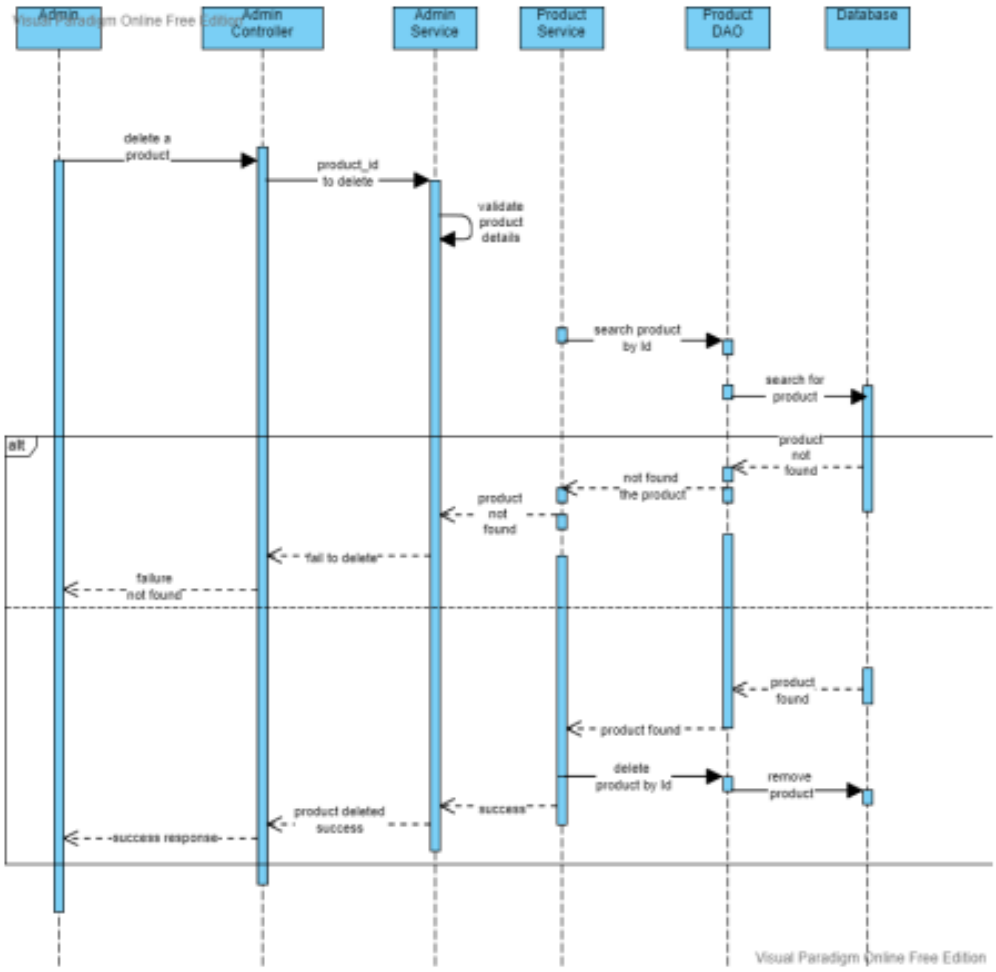
f) Update Product:



Update Product

Update Product

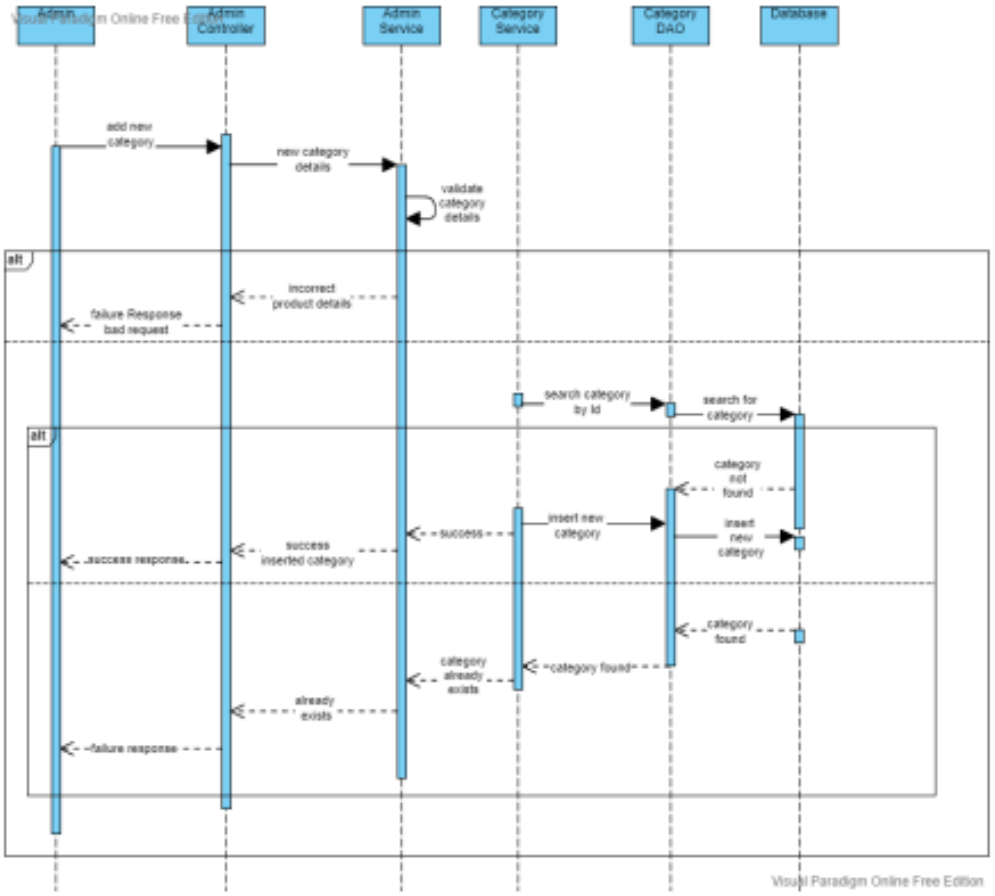
g) Delete Product:



Delete Product

Delete Product

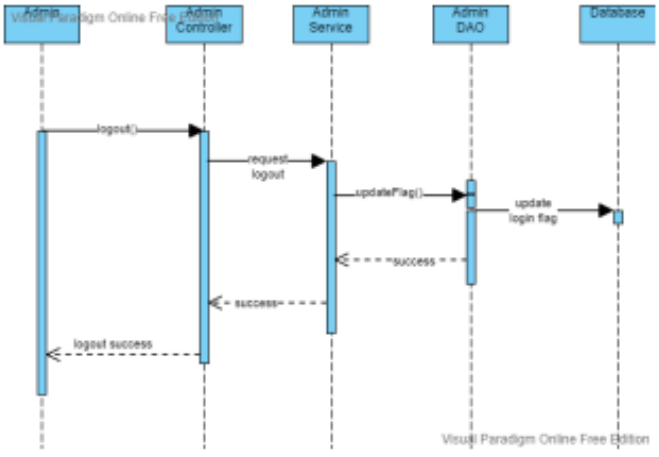
h) Add category:



Add Category

Add Category

i) Logout:

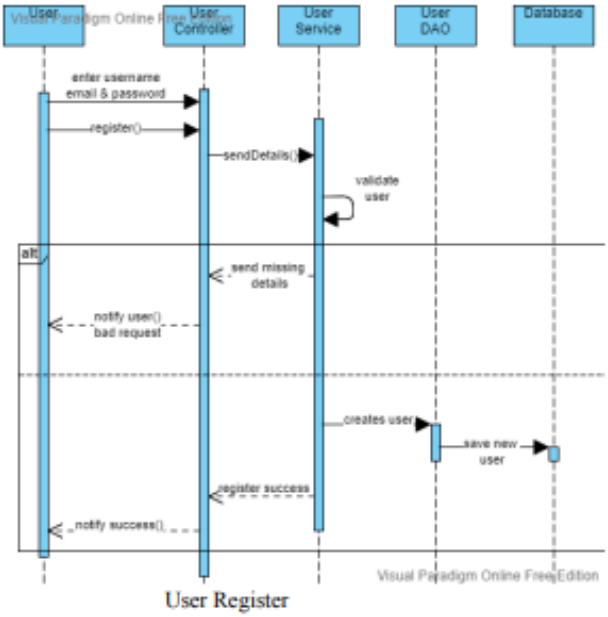


Admin Logout

Admin Logout

2) User/Customer/Client:

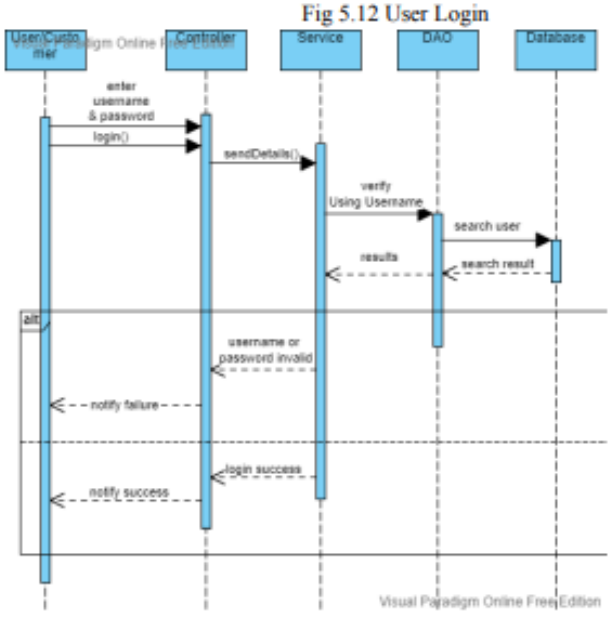
a) Register:



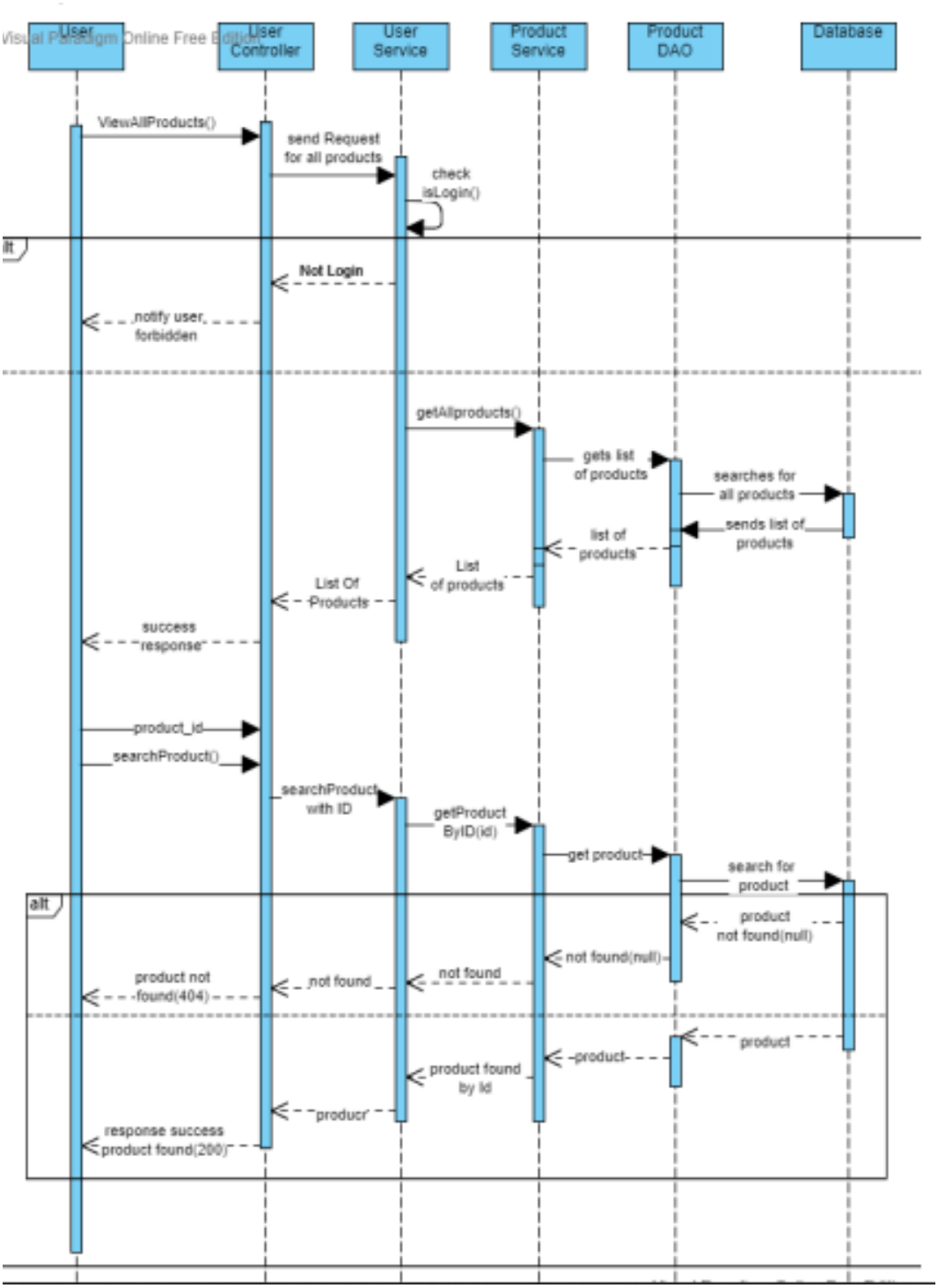
User Register

b)Login

Fig5.12UserLogin

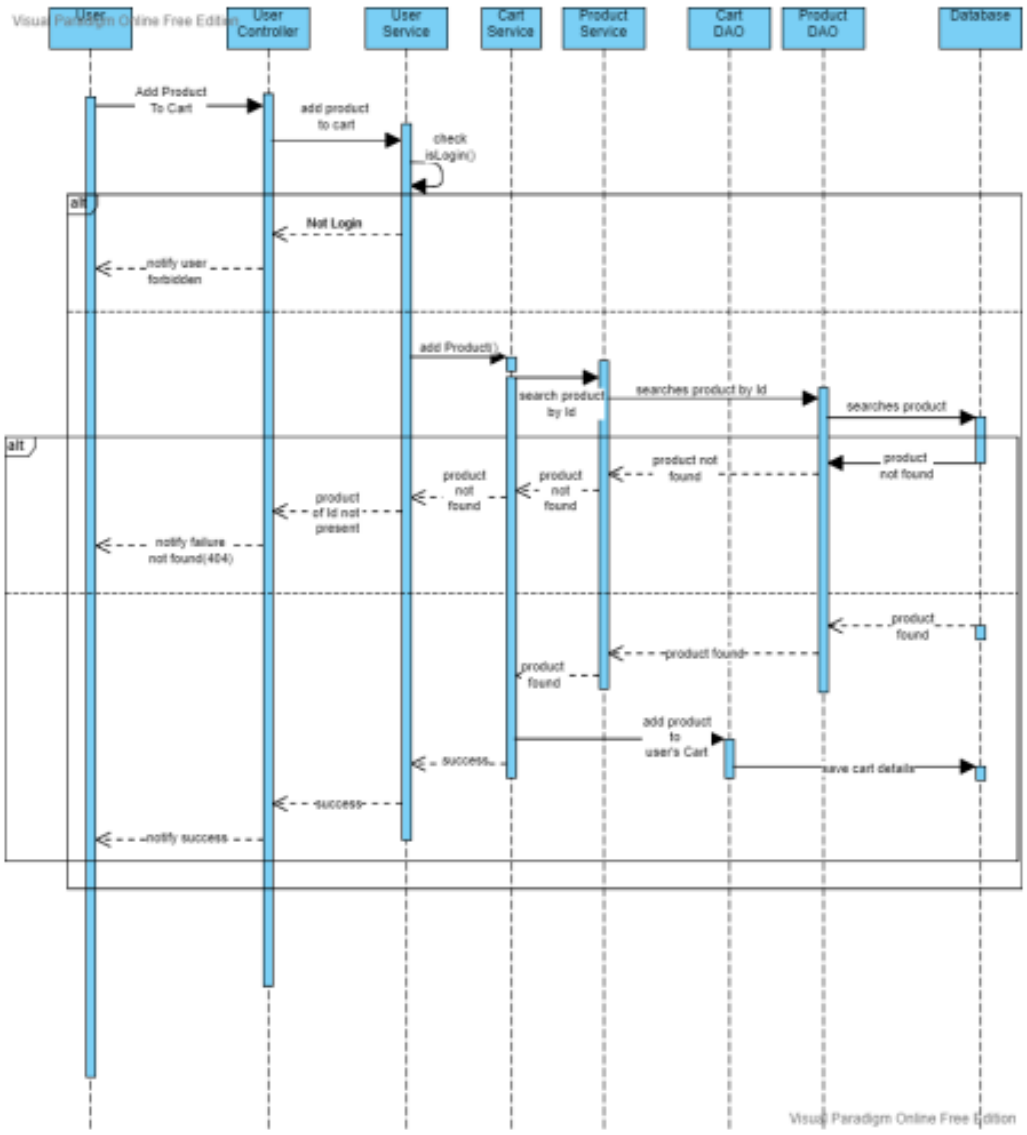


c) Get Products:



Get Products

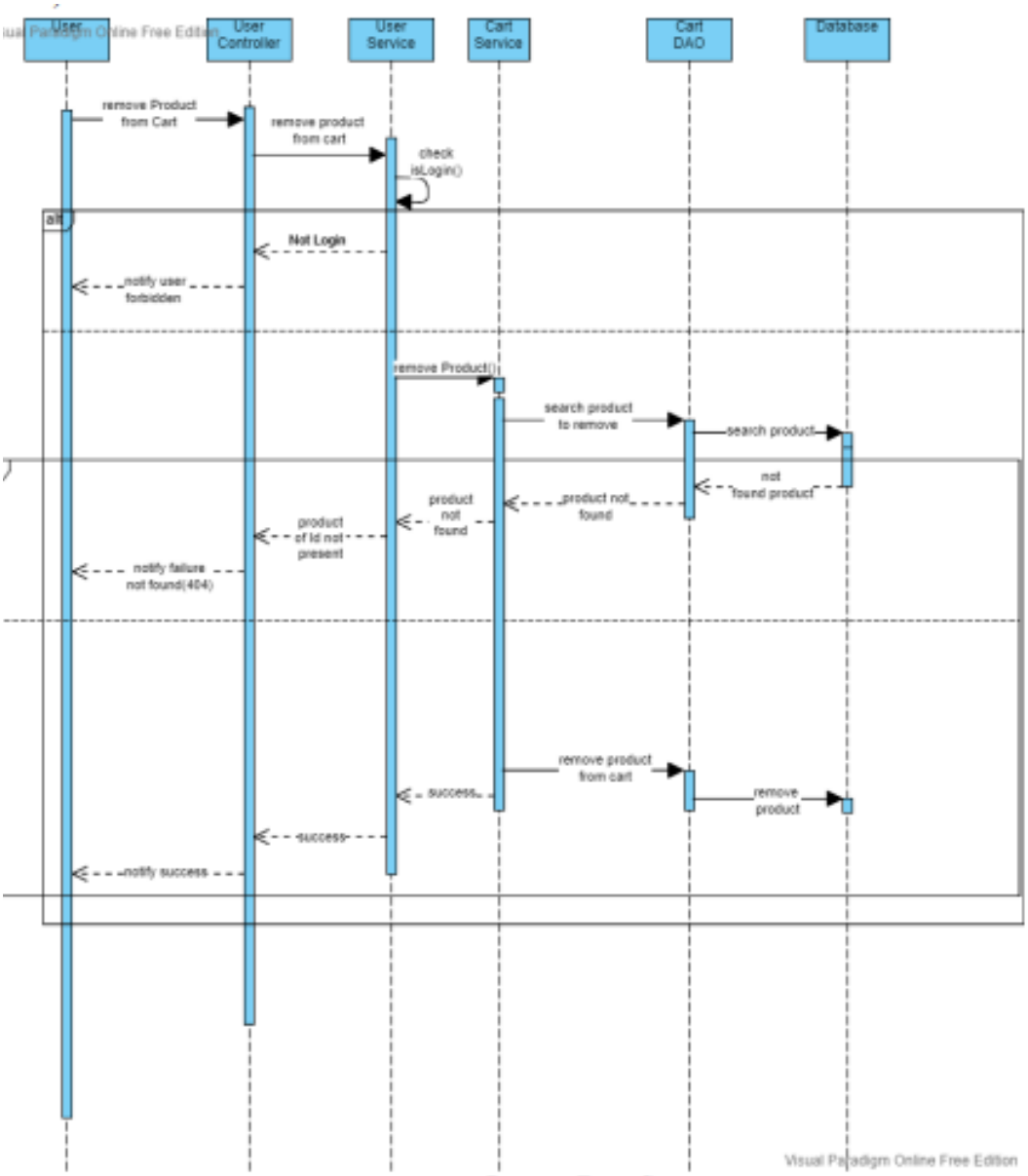
d) Add to Cart:



Add to Cart

Add to Cart

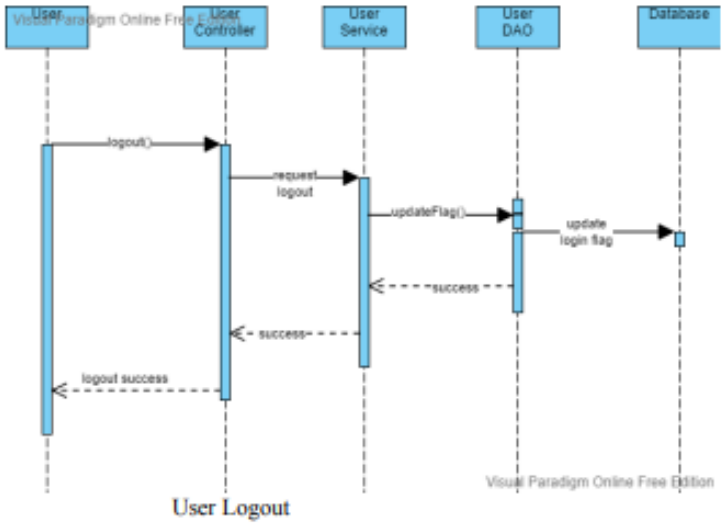
e) Remove From Cart:



Remove From Cart

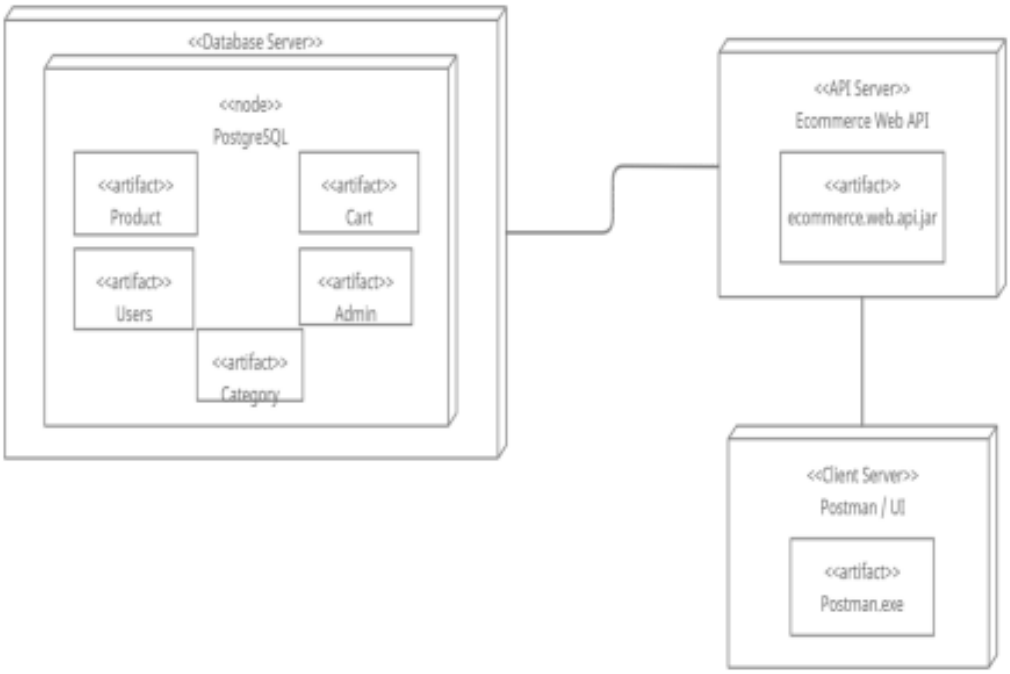
Remove From Cart

f) Logout:



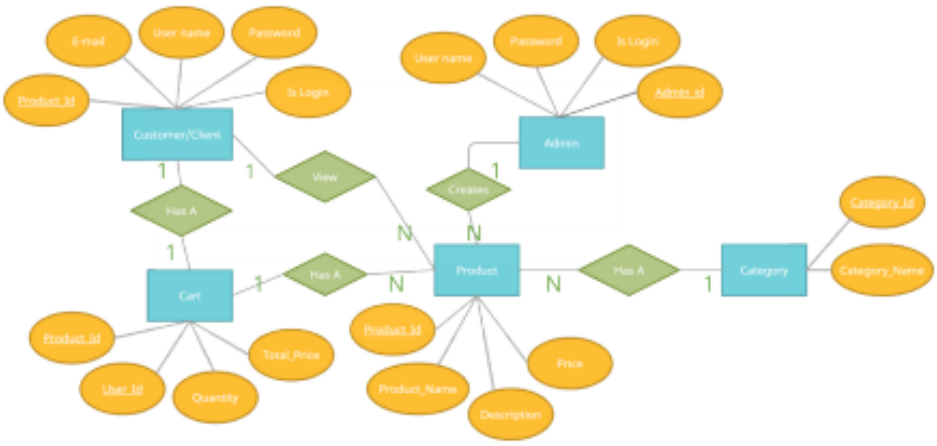
User Logout

5.1.1 Deployment Diagram:



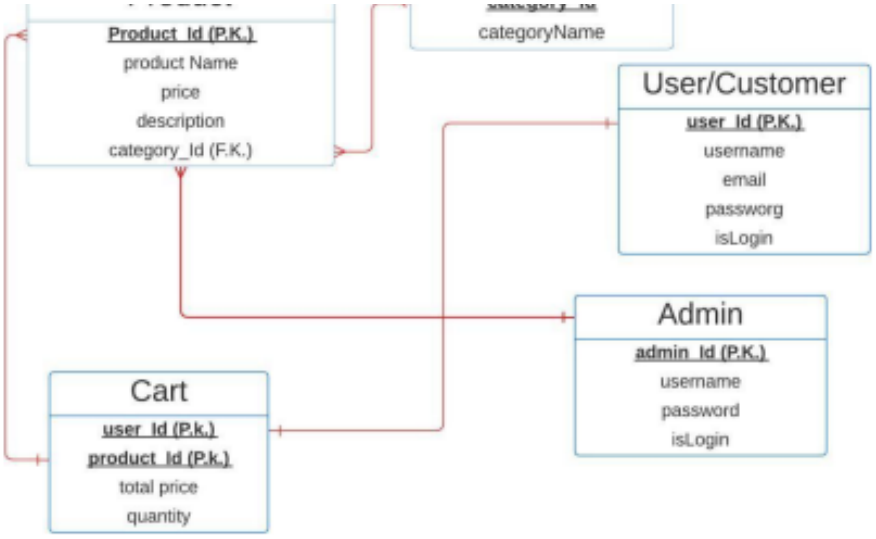
Deployment Diagram

5.1.3 ER DIagram



5.2 DATABASE DESIGN

5.2.1 Table and Relationship:



Schema Diagram

5.2.2 Data Dictionary:

a) Product

Sr. No.	Name	Data type	Not Null Primary Key
1	product_id	int	No Yes

2	product_name	varchar	No
3	category_id	int	No
4	price	int	No
5	decription	varchar	Yes
6	category_name	varchar	No

Product

b) Category:

Sr. No.	Name	Data type	Not Null Primary Key
1	category_id	Int	No Yes
2	category_name	Varchar	No

Category

c) Users:

Sr. No.	Name	Data type	Not Null Primary Key
1	user_id	Int	No Yes
2	username	Varchar	No
3	email	Varchar	No
4	password	Varchar	No
5	isLogin	Boolean	Yes

Users

d) Admin:

Sr. No.	Name	Data type	Not Null Primary Key
1	admin_id	Int	No Yes

2	username	Varchar	No
3	password	Varchar	No
4	isLogin	Boolean	Yes

Admin

e) Cart:

Sr. No.	Name	Data type	Not Null Primary Key
1	product_id	int	No Yes
2	user_id	int	No Yes
3	total_price	int	No
4	quantity	int	No

Cart

6. Implementation

6.1 IMPLEMENTATION ENVIRONMENT

During the total execution I have dealt with Intellij IDEA IDE made by JetBrains for advancement of Java Web/Console Application. Highlights incorporates investigating, linguistic structure featuring, astute code finish, pieces, code calculating and inserted Git. Client can change the topics, symbols and introduce augmentations that add extra usefulness. Assembling and conveying the Spring Boot Application is done effectively with the assistance of Intellij IDEA.

6.2 MODULES SPECIFICATION

There are two modules in the code:

1) Admin

Having Admin Side functionalities.

2) Customer

Having User Side functionalities.

6.3 CODING STANDARDS

I used the Restful approach for creating the Web API.
While writing the code I took utmost care to follow the basic coding standards while writing a Java code like:

- Limited use of global variables
- Proper exception handling
- Proper Indentation/formatting of code.
- Proper error responses for errors.
- Following proper naming conventions for global/local variables, classes, exceptions, interfaces.
- Adding JavaDocs/comments for better understanding.

6.4 EXAMPLE CODING

Add New Product (Admin Side):

1) Controller Code:

```
@PostMapping("/{adminid}/product")
public ResponseEntity<?> saveProduct(@RequestBody Product product,
@PathVariable("adminid") Integer adminId) { try {
      if (adminService.saveProduct(product, adminId) != null) { SuccessResponse
          successResponse = new SuccessResponse(200,
"Inserted Product");
          return ResponseEntity.ok().body(successResponse);
      } else {
          ErrorResponse errorResponse = new ErrorResponse(500, "Internal Server
Error");
          return ResponseEntity.status(500).body(errorResponse);
      }
   } catch (UnauthorizedAccessException unauthorizedAccessException) {
      unauthorizedAccessException.printStackTrace();
      ErrorResponse response = new ErrorResponse(403,
unauthorizedAccessException.getMessage());
      return ResponseEntity.status(403).body(response);
   } catch (IllegalArgumentException illegalArgumentException) {
      illegalArgumentException.printStackTrace();
      ErrorResponse response = new ErrorResponse(400,
illegalArgumentException.getMessage());
      return ResponseEntity.status(400).body(response);
   }
```

```
}
```

2) Service Code:

```
@Override
public Product saveProduct(Product product) throws
IllegalArgumentException {
   if (product.getProductName() == null ||
product.getProductName().isEmpty()) {
      throw new IllegalArgumentException("Product Name is Null/Empty");
   } else if (product.getPrice() == null || product.getPrice() == 0) {
      throw new IllegalArgumentException("Product Price is Null/Empty");
   } else if(product.getCategory() == null){
      throw new IllegalArgumentException("Category is Null/Empty");
   }
   return productDao.save(product);
}
```

3) Model Code:

```
@Entity
@Data
@AllArgsConstructor
@NoArgsConstructor
@Table(name = "products")
public class Product {

   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY) private
   Integer product_id;

   private String productName;

   private Integer price;

   private String description;

   @ManyToOne(targetEntity = Category.class,cascade =
CascadeType.PERSIST)
   @JoinColumn( name = "category_id" , referencedColumnName = "category_id")
   private Category category;

}
```

4) Response Code:

```
@Data
@AllArgsConstructor
public class ProductResponse {
   private Product product; private
   Integer StatusCode; private String
   message;
}
```

7 Tools, Technologies, APIs and Libraries

Development Tool (Intellij) :

- **IntelliJ IDEA** is a coordinated improvement climate (IDE) written in Java for creating PC programming. It is created by JetBrains (previously known as IntelliJ), and is accessible as an Apache 2 Licensed people group release, and in a restrictive business version. Both can be utilized for business advancement.

Java :

- Java is an undeniable level, class-based, object-situated programming language that is intended to have as scarcely any execution conditions as could be expected. It is a broadly useful programming language expected to allow developers to compose once, run anyplace (WORA),[17] implying that ordered Java code can run on all stages that help Java without the need to recompile.

Example Code:

```
class HelloWorld {

   public static void main(String[] args) {

      System.out.println("Hello, World!");

   }

}
```

Spring Boot :

Spring Boot makes it simple to make independent, creation grade Spring based Applications that you can "recently run".

Features:-

Make your own Spring apps. Embed Tomcat, Jetty, or Undertow natively (no WAR files required). To simplify your build settings, provide opinionated'starter' dependencies.Automatically configure Spring and 3rd party libraries whenever

possible

Provide features that are ready for production, such as metrics, health checks, and externalised configuration. There is no need for code creation or XML setup.

Example Code :-

```
package com.javatpoint.springbootexample;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SpringBootExampleApplication
{
    public static void main(String[] args)
    {
            SpringApplication.run(SpringBootExampleApplication.class, a
        rgs);
    }
}
```

Postgresql :

PostgreSQL is a strong, open source object-social data set framework that purposes and broadens the SQL language joined with many elements that securely store and scale the most muddled information jobs. The starting points of PostgreSQL date back to 1986 as a component of the College of California at Berkeley and has over 30 years of dynamic advancement on the center stage.

PostgreSQL

Version Control (Git/GitHub) :

Git is a variant control device (programming) to follow the progressions in the source code. GitHub is an online cloud administration to have your source code(Git vaults).

Git & GitHub

FLYWAY

Flyway is an open-source apparatus, authorized under Apache License 2.0, that assists you with carrying out robotized and form based data set movements. It permits you to characterize the necessary update tasks in a SQL script or as Java code. You can then run the relocation from an order line client or consequently as a feature of your fabricate interaction or incorporated into your Java application. The beneficial thing about this cycle is that Flyway recognizes the necessary update activities and executes them. Thus, you don't have to know which SQL update articulations should be performed to refresh your ongoing information base. You and your associates simply characterize the update activities to relocate the information base starting with one form then onto the next. Furthermore, Flyway identifies the ongoing form and plays out the important update

tasks to get the data set to the most recent variant. To have the option to do that, Flyway utilizes a metadata table to record the ongoing data set form and every single executed update. Of course, this table is called SCHEMA_VERSION.

DOCKER

Docker is an open source containerization stage. It empowers designers to bundle applications into holders — normalized executable parts joining application source code with the working framework (OS) libraries and conditions expected to run that code in any climate. Holders work on conveyance of disseminated applications, and have become progressively famous as associations shift to cloud-local turn of events and mixture multicloud conditions. Engineers can make holders without Docker, yet the stage makes it more straightforward, less complex, and more secure to assemble, convey and oversee compartments. Docker is basically a tool stash that empowers designers to fabricate, convey, run, update, and stop holders utilizing straightforward orders and work-saving robotization through a solitary API. Highlights of Docker can diminish the size of improvement by giving a more modest impression of the working framework by means of holders. With holders, it becomes simpler for groups across various units, for example, improvement, QA and Operations to work consistently across applications. You can convey Docker compartments anyplace, on any physical and virtual machines and, surprisingly, on the cloud. Since Docker holders are lightweight, they are effectively adaptable. Parts of docker . Docker for Linux – It permits one to run Docker holders on the Linux OS. Docker Engine – It is utilized for building Docker pictures and making Docker compartments. Docker Hub – This is the library which is utilized to have different Docker pictures. Docker Compose – This is utilized to characterize applications

utilizing various Docker holders.

POSTMAN

Postman is an API client that makes it easy for developers to create, share, test and document APIs. This is done by allowing users to create and save simple and complex HTTP/s requests,
as well as read their responses, the result - more efficient and less tedious work. Postman is very convenient when it comes to executing APIs. Once you've entered and saved them, you can simply use them over and over again, without having to remember the exact endpoint, headers, API keys, etc.

SWAGGER

Strut is utilized for API documentation. Strut permits you to depict the construction of your APIs that machines can understand them. The capacity of APIs to portray their own design is the foundation of all wonder in Swagger. For what reason is it so amazing? All things considered, by perusing your API's design, we can consequently assemble lovely and intuitive API documentation. We can likewise naturally create client libraries for your API in numerous dialects and investigate different potential outcomes like robotized testing. Strut does this by requesting that your API return a YAML or JSON that contains an itemized depiction of your whole API. This document is basically an asset posting of your API which sticks to Open API Specification. The particular requests that you incorporate data like: What are the tasks that your API upholds? What are your API's boundaries and what does it return? Does your API require some approval? And, surprisingly, fun things like terms, contact data and permit to utilize the API. You can compose a Swagger spec for your API physically, or have it created naturally from explanations in your source code.

GRADLE

Gradle is an open source construct mechanization apparatus that depends on the idea of Apache Maven and Apache Ant. It is fit for building practically any sort of programming. It is intended for the multi-project fabricate, which can be very huge. It presents a Java and Groovy-based DSL (Domain Specific Language) rather than XML (Extensible Markup Language) for pronouncing the venture design. It utilizes a DAG (Directed Acyclic Graph) to characterize the request for executing the errand. Gradle offers a versatile model that can help the improvement lifecycle from arranging and bundling code for web and portable applications. It offers help for the structure, testing, and sending programming on various stages. It has been produced for building mechanization on numerous dialects and stages, including Java, Scala, Android, C/C ++, and Groovy. Gradle furnishes coordination with a few improvement instruments and servers, including Eclipse, IntelliJ, Jenkins, and Android Studio.

8 TESTING

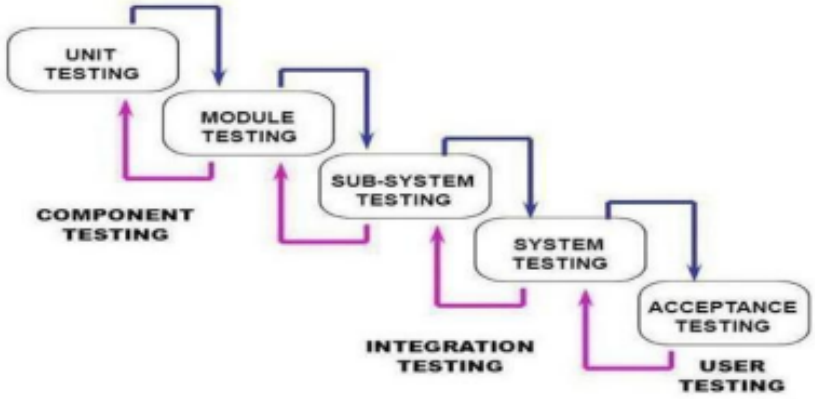
Whenever code has been created, programming should be tried to uncover whatever number mistakes as could reasonably be expected before conveyance to client. You want to plan a progression of experiments that have a high probability of tracking down blunders. Programming testing methods give deliberate direction to planning tests that (1) practice the inward rationale of programming parts, and (2) practice the data sources and results spaces of the program to reveal mistakes in program capacity, conduct and execution.

Testing Objectives:

• Testing is a course of executing a program determined to track down a blunder.

• A decent experiment is one that has a high likelihood of tracking down an at this point unseen blunder.

• A fruitful test is one that uncover an at this point unseen mistake.



Testing Strategy

Testing Strategy

Unit Testing:

Unit testing is a product improvement process in which the littlest testable piece of an application, called units, are exclusively examined for legitimate activity. Unit testing is frequently mechanized however it should likewise be possible physically. This testing mode is a part of Extreme Programming (XP), a down to earth strategy for programming improvement that adopts a fastidious strategy to building an item through ceaseless testing and update. Unit testing includes just those attributes that are
fundamental to the presentation of the unit under test. This urges engineer to alter the source code without prompt worries about what such changes could mean for the working of the units or the program in general. Once of entire of the units in a program have been viewed as working in the most productive and blunder free way conceivable, bigger parts of the program can be assessed through reconciliation testing. I tried each single piece of the whole application. I tried every single module separately.

Sub System Testing:

In the wake of testing every unit, we continue on toward bigger units called sub framework. In

subsystem testing I tried the entire client side as one framework. On the client side every one of the modules like dashboard, API, and so on were tried together to check whether there was any blunder or bug found.

System Testing:

In the wake of testing all the sub-framework, the time has come to test the entire framework. Framework testing of programming is trying directed on a total, incorporated framework to assess the framework's consistence with its predefined prerequisites. While testing the entire framework I found numerous mistakes like the information mining postpones prompting hardships in inputs. I addressed it by rolling out fitting improvements in the span of information mining as well as changed its properties. I chipped away at every blunder and special case that I got while testing and a large portion of them are eliminated or made such revision that it won't reoccur.

- Recuperation Testing: It is a framework test that powers the product to flop in an assortment of ways and checks that recuperation is appropriately performed.

- Security Testing: It endeavors to confirm that assurance components incorporate into a framework will, as a matter of fact, shield it from ill-advised infiltration.

- Execution Testing: It is intended to test the run-time execution of programming inside the setting of a coordinated framework execution testing happens all through all means in the testing system.

Acceptance Testing:

Acknowledgment testing can be associated toward the end client, client, or client to approve the decision about whether to acknowledge the item. Acknowledgment testing might be proceeded as a feature of the hand-off process between any two periods of improvement. The acknowledgment test suite is run again the provided input information or utilizing an acknowledgment test content to coordinate the analyzer. Then, at that point, the outcomes acquired are contrasted and the normal outcomes. On the off chance that there is a right counterpart for each case, the test suite is said to pass.

TESTING METHODS:

The check exercises fall into the class of static testing. During static testing, you have an agenda to check whether the work you are doing is going according to the set principles of the association. These guidelines can be for coding, incorporating and arrangement. Surveys, Inspections and Walkthroughs are static trying philosophy. Dynamic testing includes working with the product giving information esteems and checking in the event that the result is true to form. These are the approval exercises. Unit test, reconciliation test, System and acknowledgment tests are not many of the powerful testing strategies.

Alpha and beta testing: the alpha test is directed at the designer's site by a client. The product is utilized in a characteristic setting with the engineer "investigating shoulder" of the client and recording blunders and utilization issues. Alpha test is led in a controlled climate. The beta testing is led at least one client site toward the end-client of the product. Dissimilar to alpha testing, the engineer is for the most part not present. In this way, the beta test is a "live" use of the product in a climate that can't be constrained by the designer.

Black Box Testing:

Otherwise called useful testing. A product testing procedure where by the inward working of the thing being tried are not known by the analyzer. For instance, in a black box test on programming plan the analyzer just knows the data sources and what the normal results ought to be and not the way that the program shows up at those results. The analyzer never examines the programming code and needn't bother with any further information on the program other than its particular.
The benefits of this kind of testing include:

● The test is unprejudiced as the fashioner and the analyzer are indepAlso known as practical testing. A product testing procedure where by the interior working of the thing being tried are not known by the analyzer. For instance, in a black box test on programming plan the analyzer just knows the data sources and what the normal results ought to be and not the way in which the program shows up at those results. The analyzer never examines the programming code and needn't bother with any further information on the program other than its determination.
The benefits of this sort of testing include:

● The test is fair-minded as the originator and the analyzer are autonomous of one another. ● The analyzer needn't bother with information on a particular programming dialects.

● The test is done according to the perspective of the client, not the creator.

● Experiments can be planned when the particulars are finished.
The burdens of this kind of testing include:
The test can be excess in the event that the computer programmer has proactively run an experiment.
The experiments are challenging to plan.
Testing each conceivable information stream is ridiculous in light of the fact that it would take an excessive measure of time: thus many program ways will go untested.
endent of one another.

● The analyzer needn't bother with information on a particular programming dialects.

● The test is done according to the perspective of the client, not the fashioner.

● Experiments can be planned when the determinations are finished.
The burdens of this kind of testing include:
The test can be excess in the event that the programmer has previously run an experiment.

The experiments are challenging to plan.

Testing each conceivable info stream is ridiculous on the grounds that it would take an unreasonable measure of time: consequently many program ways will go untested.

White Box Testing:

Otherwise called glass box, underlying, clear box and open box testing. A product testing method by which unequivocal information on the inward operations of the thing being tried are utilized to choose the test information. Not at all like black box testing, white box testing utilizes explicit information on programming code to inspect yields. The test is exact provided that the analyzer knows what the program should do. The individual in question can than check whether the program wanders from its expected objective.

Design of Test Cases:

A wide range of test design methodologies for software have emerged to reduce the amount of faults in software. These approaches allow the developer to test in a methodical manner. More importantly, techniques give a methodology for ensuring the thoroughness of tests and increasing the possibility of finding software faults.

One of two strategies can be utilized to test a designing item:

1) With information on the item's planned capacity, tests might be embraced to affirm that each capacity is totally utilitarian while likewise searching for deserts in each capacity.

2) By understanding an item's inner operations, testing might be completed to ensure that "all stuff network," that is, inside mistreatment is done by details and all interior parts are fittingly worked out. The experiments for our application are recorded beneath.

TEST CASES:

Test Cases

Sr. No.	Purpose	Input	State	Expected Output / Actual Output / Output Response Code	Test Result
1	User Login	Username, passwor	logout	200 Login As Success	Pass

		d		expected Response	
2	User Login	Invalid Username or Password	logout	400 Error As Response expected for Login	Pass

3	Admin Login	Password	Password logout 200 Login Success Response	logout 400 Error Response for Login As expected	As expected	Pass
4	Admin Login Username,	Invalid Username or				Pass

5	User Register	New username, email &	logout	200 Register As expected Success Response	Pass

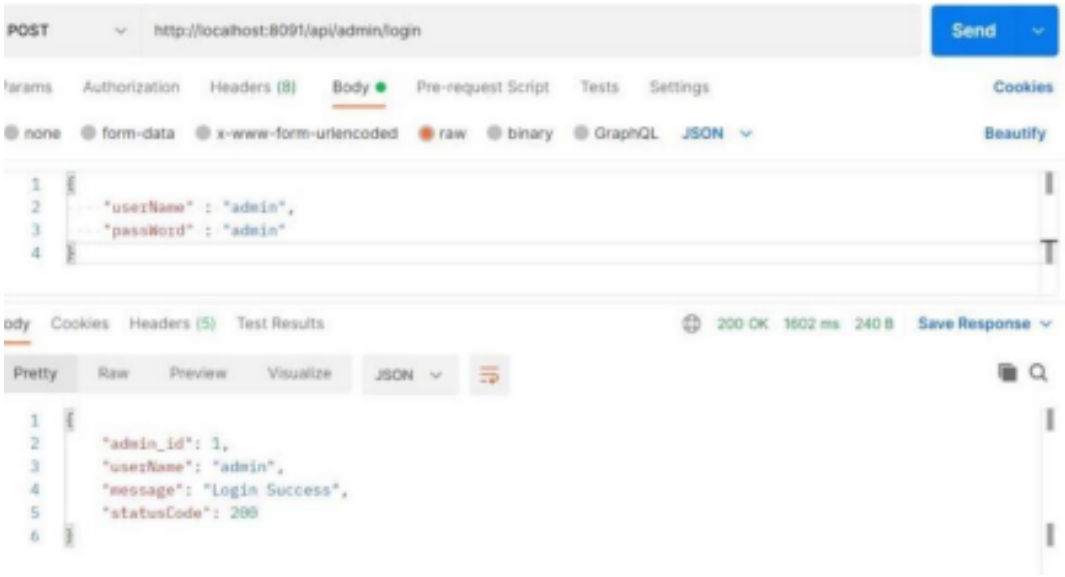
		password			
6	User Register	Already used Username, email or Password is less than 8 characters	logout	400 Error As Response expected for Register	Pass
7	Admin add product	Name, category, price, description , product_id	login	200 Product As Added expected Success Response	Pass

8	Admin add product	Name null or empty or price 0 or category empty	login	400 Error As Response expected for Product Addition	Pass
9	Admin delete product	Product id to be deleted	login	200 Deletion As Success expected Response	Pass
10	Admin delete Product	Invalid product id	login	404 Error As Response expected for Deletion	Pass
11	Admin view all Users	-	login	200 List of As Users in expected System	Pass
12	User view all Products	-	login	200 List of As Products expected in System	Pass
13	User add produc t to cart	Product id to be added	login	200 Cart As Addition expected Success Response	Pass
14	User add produc	Invalid product Id	login	404 Error As Response	Pass

	t to cart			expected for Cart Addition	
15	User remove product from cart	Product id	Login	200 Remove As Product expected Success Response	Pass

16	Admin add new Produ ct	Name, category, price, descripti on , product_id	logout	403 Error As Response expected for Auth	Pass
17	User add produc t to cart	Product id	logout	403 Error As Response expected for Auth	Pass
18	User View Cart	-	login	200 List of As expected products in Cart	Pass
19	User View Cart	-	login	404 Empty As Cart expected Response	Pass
20	Admin View Product	Product id	login	200 Product As Response expected related to ID	Pass

User manual
ADMIN:

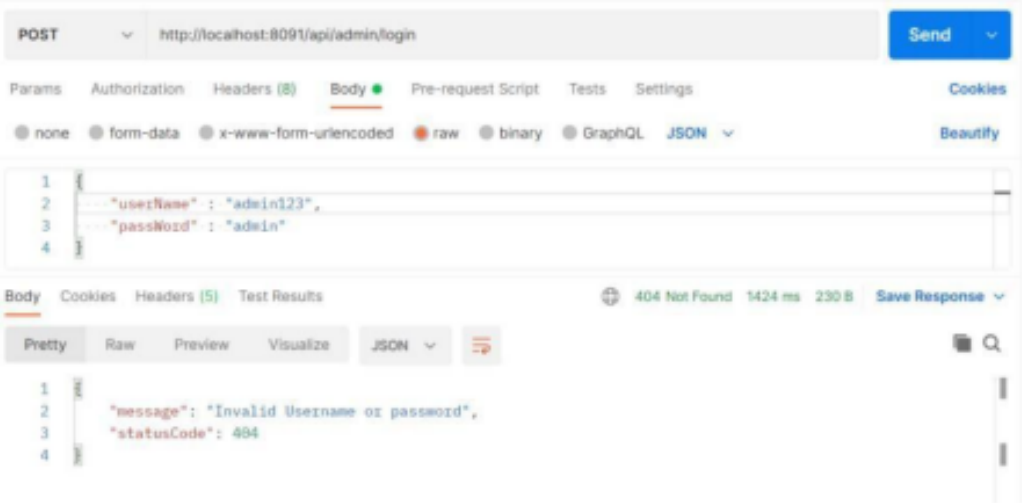


Login

Input: username & password

Processing: Authentication Success if Credential correct Output: Login Success Response

Login Fails



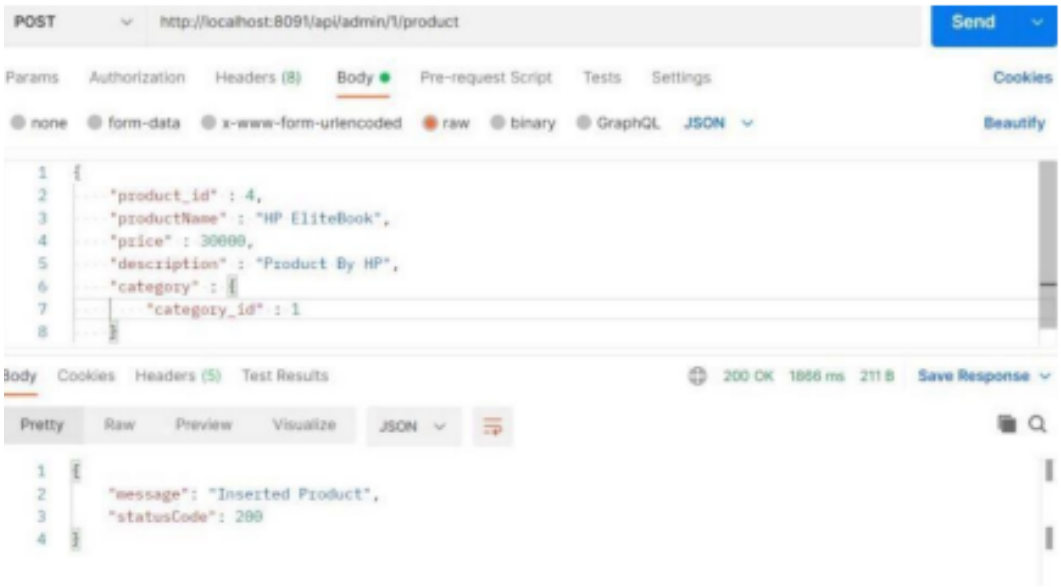
Failed Login

Failed Login

Input: invalid username or password

Processing: Authentication Failed Output: Error Response for Login Failed.

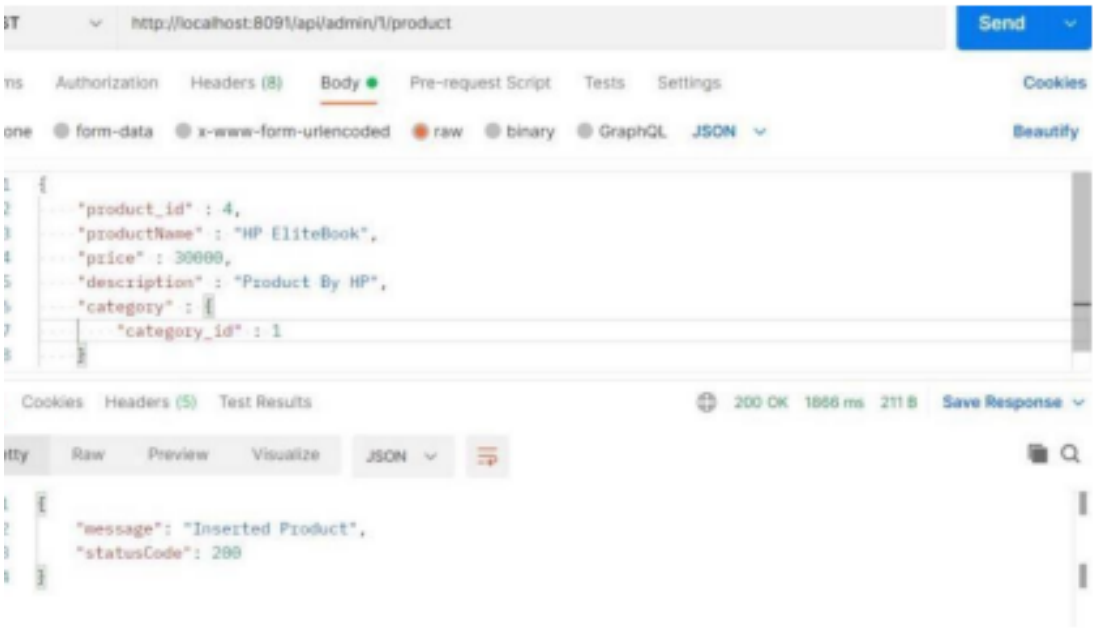
Add New Product:



Add New Product

Input: product_id, productName, price, description, category_id
Processing: Insert Product Success
Output: Success Response of Insertion

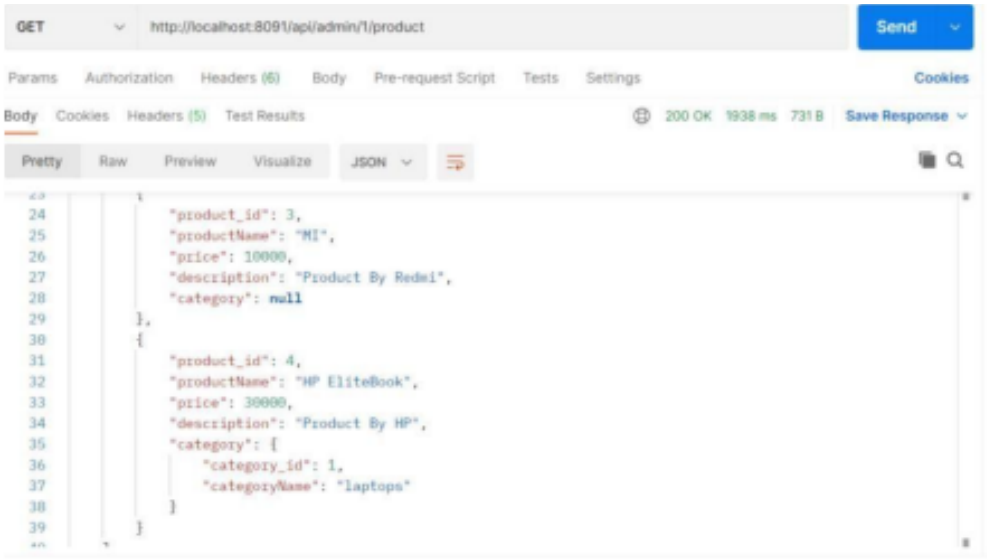
View all Products:



View Products

Processing: Get All Products
Output: List of all products

View Product by Id:

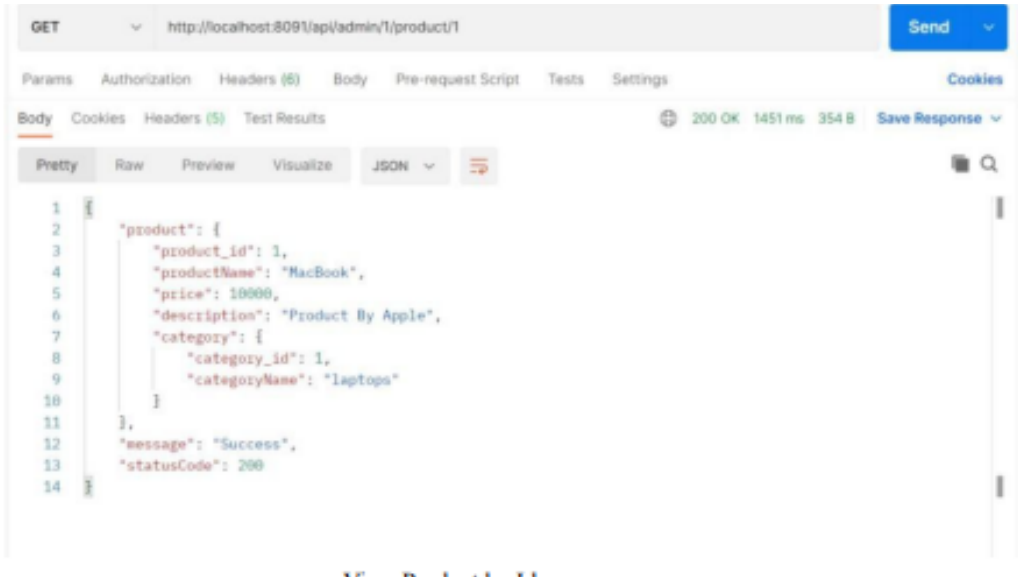


View Products

View Product by Id

Input: product_id to be searched
Processing: gets product by its id if correct Id is passed Output:
Success Response with Product with product_id

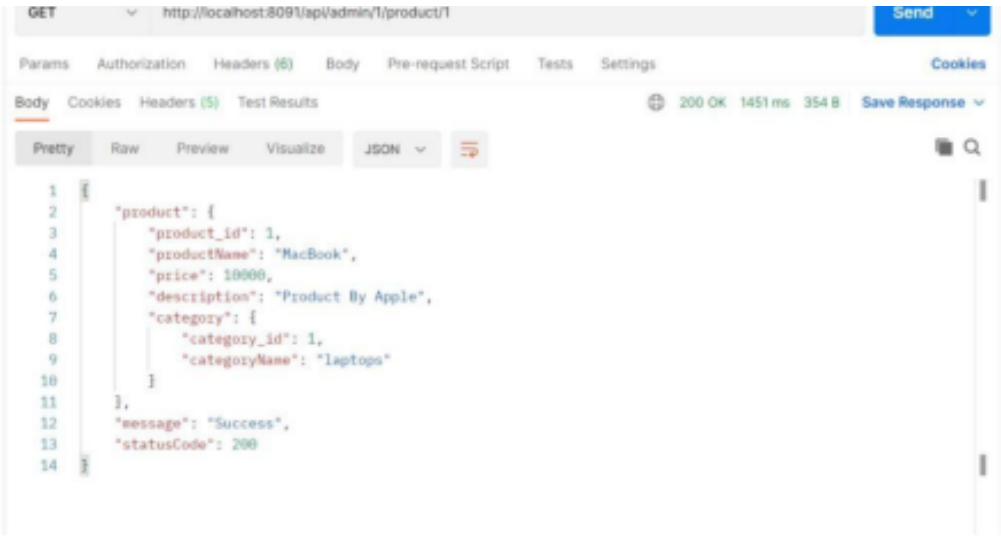
View Categories:



View Categories

Processing: Gets the categories present
Output: Success Response with list of categories.

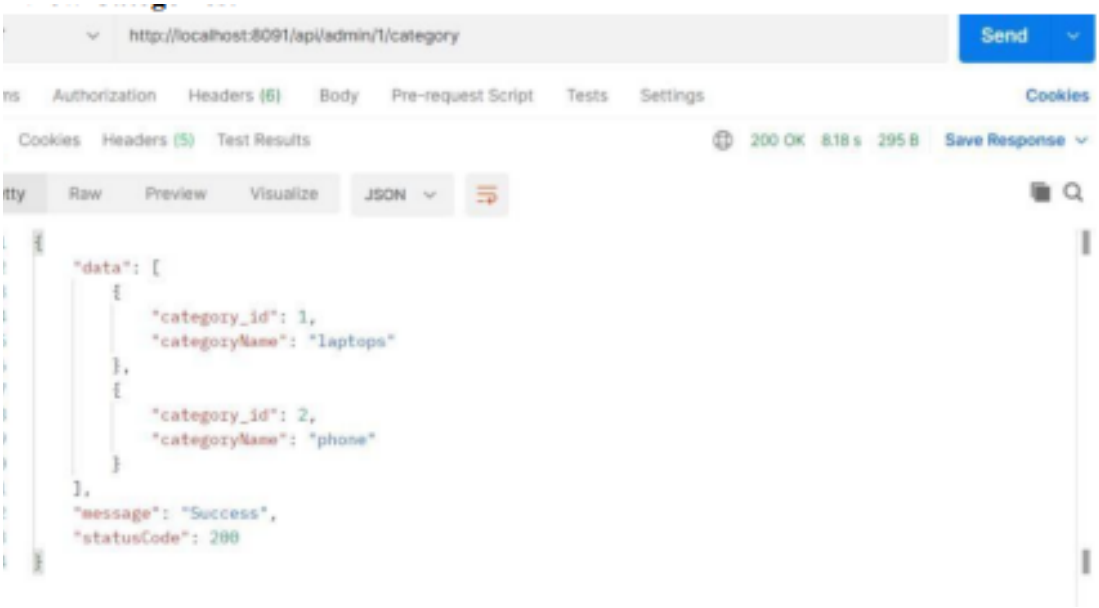
Delete Product by its ID:



Delete Product

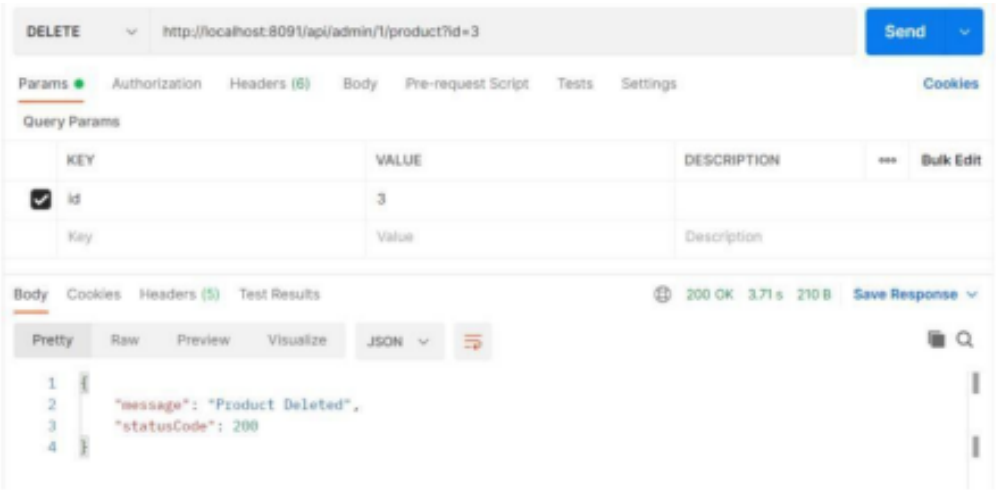
Input: product_id to be deleted

Processing: deletes product by its id if correct id is passed Output: Success response of Product Deletion



Product List After Deletion

Delete Product with Invalid product_id:

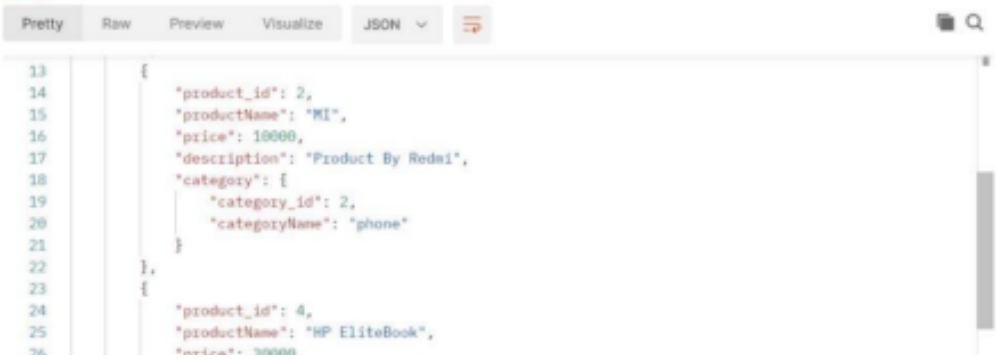


Delete Product

Delete Product with Invalid Id

Input: Invalid product_id
Processing: checks product is present or not Output:
Error Response for Product Deletion

Logout:

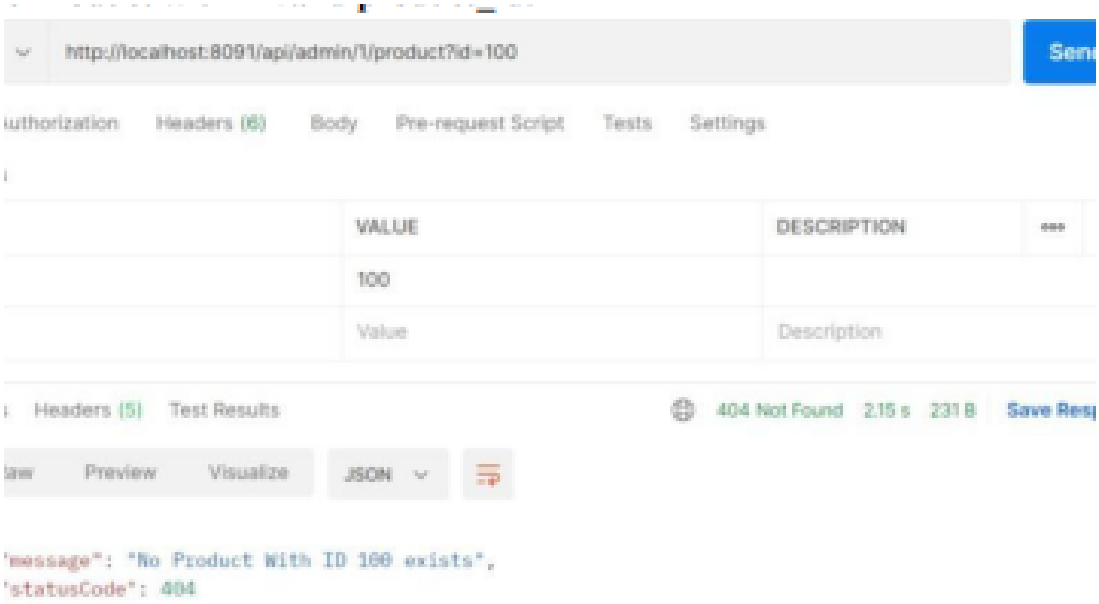


Logout

Processing: Changes isLogin flag to false
Output: Success Response for Logout

USER/CUSTOMER:

Register:

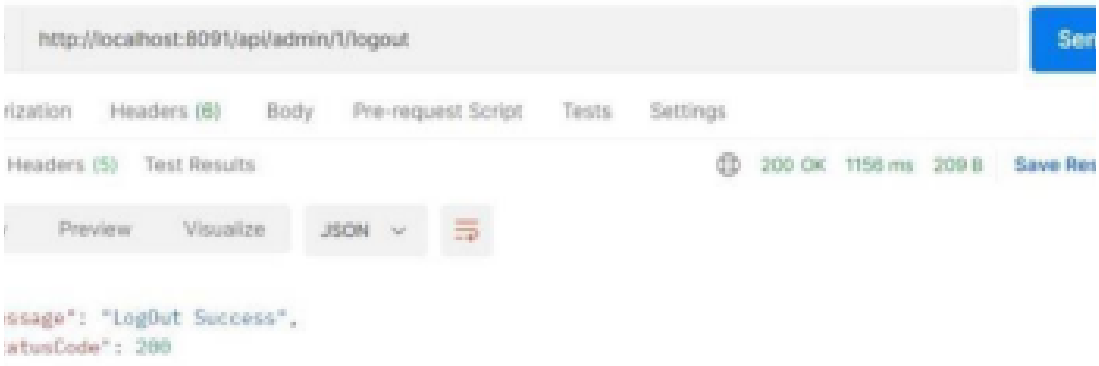


Delete Product with Invalid Id

User Register

Input: username, password, email
Processing: registers new user
Output: Success Response with User's access Id

Register Failed:

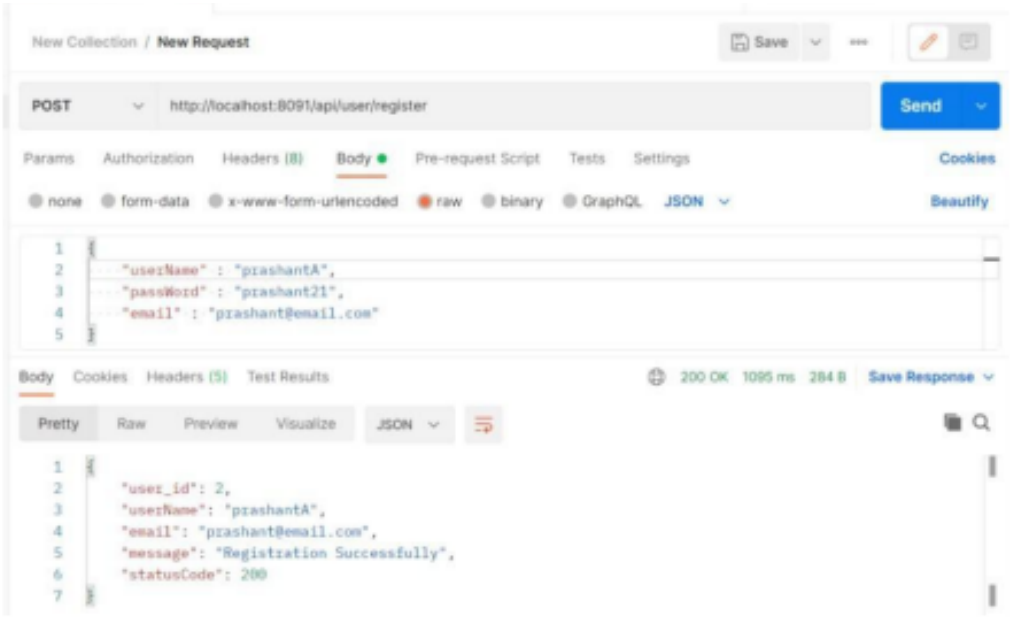


Logout

Register Failed

Input: invalid password or non-unique username
Processing: check with database for username or checks password is greater than 8 chars
Output: Error Response with correct Error message.

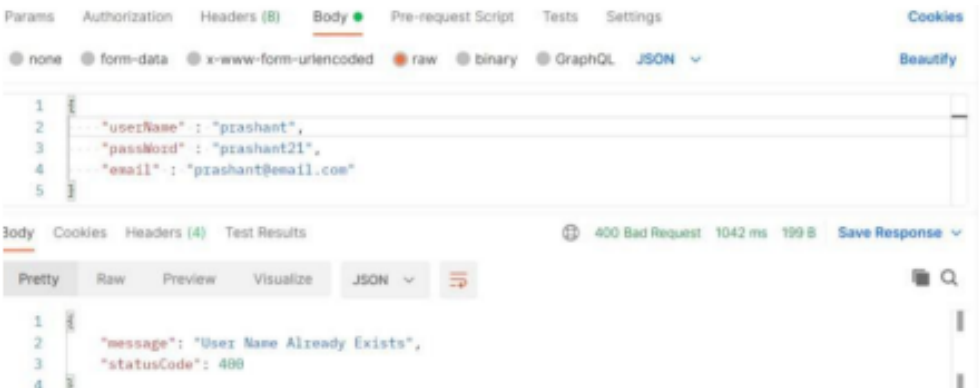
User view Products:



User Views products

Processing: gets all products
Output: Success Response with Product List

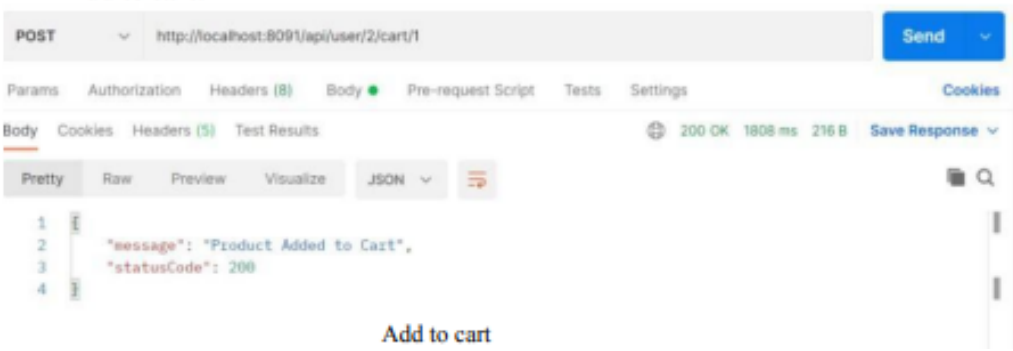
Add to cart:



Add to cart

Input: Product id to be added to cart
Processing: product added to cart Output:
Success Response for addition

Add to cart with invalid product's id:



Add to cart

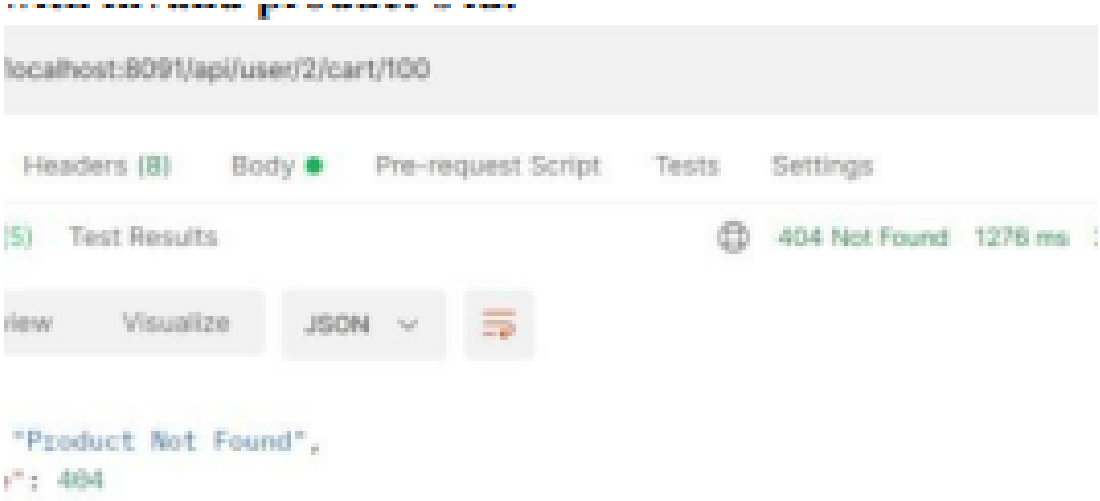
Add to cart(invalid Id)

Input: Invalid product id

Processing: checks for product with the same Id

Output: Error Response of Not Found

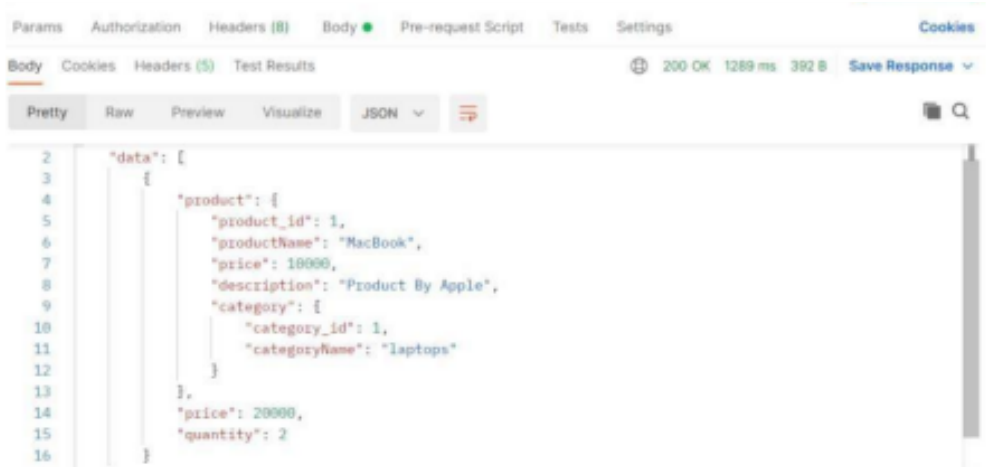
View Cart



view cart

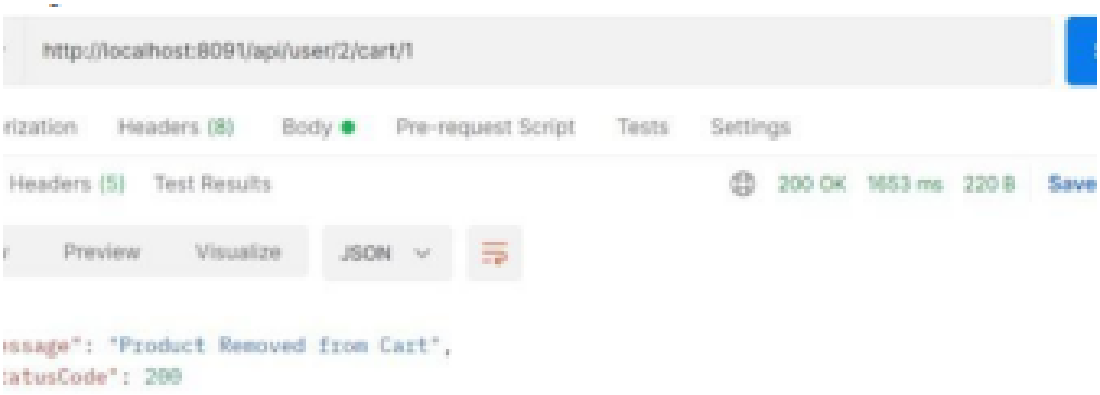
Processing: gets products list associated with user's cart Output: List of Products with total price & quantity

Remove product from cart:



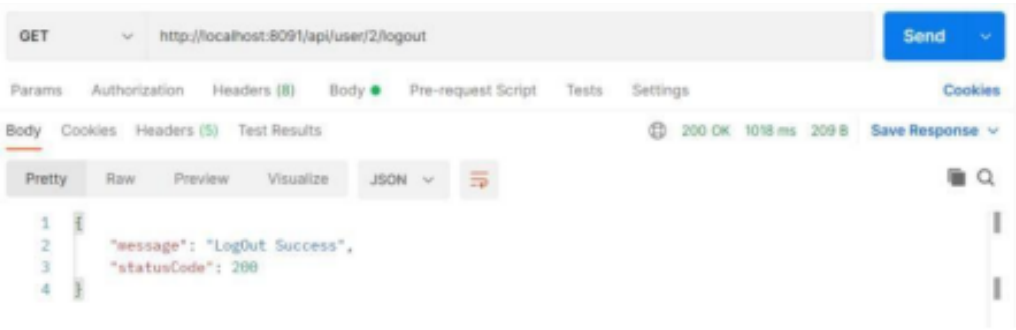
Remove product from cart

Input: product to be removed
Processing: Removes product from cart
Output: Success Response for remove the product



View cart of Product removal

User Logout:

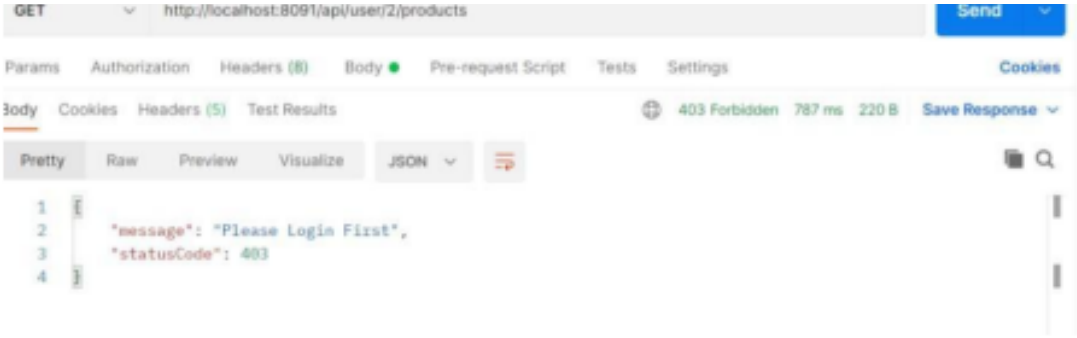


Logout

Logout

Processing: Changes isLogin flag to false
Output: Success Response for Logout

Access without Login



Access without Login

Access without Login

LIMITATION AND FUTURE ENHANCEMENT

LIMITATION

1) There are limited numbers of filters for getting the Product.

2) Limited number of errors/exception is handled.

3) Limited number of data is there for API.

4) Authentication in API is not much secure yet.

FUTURE ENHANCEMENT

1) Increase the efficiency of the API.

2) New fields to be added to the API data dictionary.

3) Authentication technique can be enhanced.

4) Functionality like Checkout, manage orders can be added in future.

5) Structuring of API Responses can be enhanced.

CONCLUSION AND DISCUSSION

CONCLUSION

This project has been implemented from what we learned in our internship and training. The scope of this project is subjective to the type of application to be built. However it was developed by keeping in mind the goal to keep it as generic as possible.

Also, enhancement which we can do in future is improving speed and accuracy, add more functionalities, extending & enhancing the current functionalities and add secure Authentication methods to API.

DISCUSSION

Self Analysis of the Project:

In my opinion, this project serves the goal that I set at the beginning for the project. It provides functionalities like cart management & authentication to the user and for admin it provides crud operations on product etc. Besides this functionality, there are numerous other improvements as well as enhancements to this API. New functionalities can also be added easily to the project.

References

1) Best Practices for Web API: https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-design
2) JavaTpoint : https://www.javatpoint.com/
3) Spring.io: https://spring.io/
4) Stackoverflow: https://stackoverflow.com/
5) TutorialsPoint: https://www.tutorialspoint.com/index.htm