Table Of Contents

1. INTODUCTION

1.1 Problem Statement

Simple Ecommerce Cart Application for ordering Clothes which must be responsive and should be a single page application.

1. **Page layout consists of 2 sections header and content.**

2. **Header should occupy full width of screen with height 56px.**

- ✓ Header container has logo container consists of cart logo as icons + text (item count) as shown. Dimension width 80px and height 56px.
- ✓ Logo could be any small image of dimensions width 40px and height 40px.
- ✓ Logo text "cart items count" as shown in above screen shot 1.

3. **Content part has the product list container, width of it is 1320px**

- ✓ Content has 2 sections filter and product list containers.
- ✓ Filter container consists of different size of shirts is displayed.

1.2 GOAL

User-Friendly
Simple
Effective and Effective.

1.3 OBJECTIVE

The user can select the cloths according to the sizes available.
The user can view the ordered items in the cart
User can also be able to see the sum of total in the cart page itself.
All the ordered items are stored in the local storage.

2. SYSTEM REQUIREMENTS SPECIFICATION

2.1 Proposed System

"Shopping Kart" is a website where customers can buy the clothes as per their wish. This application consists of 2 sections one is Header and the other is Content, where the content part consists of 3 sub-division 1. Sizes 2. Products View and 3. Cart. The customers can buy the clothes with various sizes available in the application. Customers have variety of options while purchasing. Since it is a single page application, it is user friendly, customer can easily view the products available according to sizes available and easily add and remove the products into the cart. As soon as the updates the cart, he/she can be able to see the total sum up at the bottom of the cart, so that they can get to know their total purchase amount there itself. Finally at last all the shopping cart data will be stored in local storage for future references.

2.2 Advantages of Proposed System.

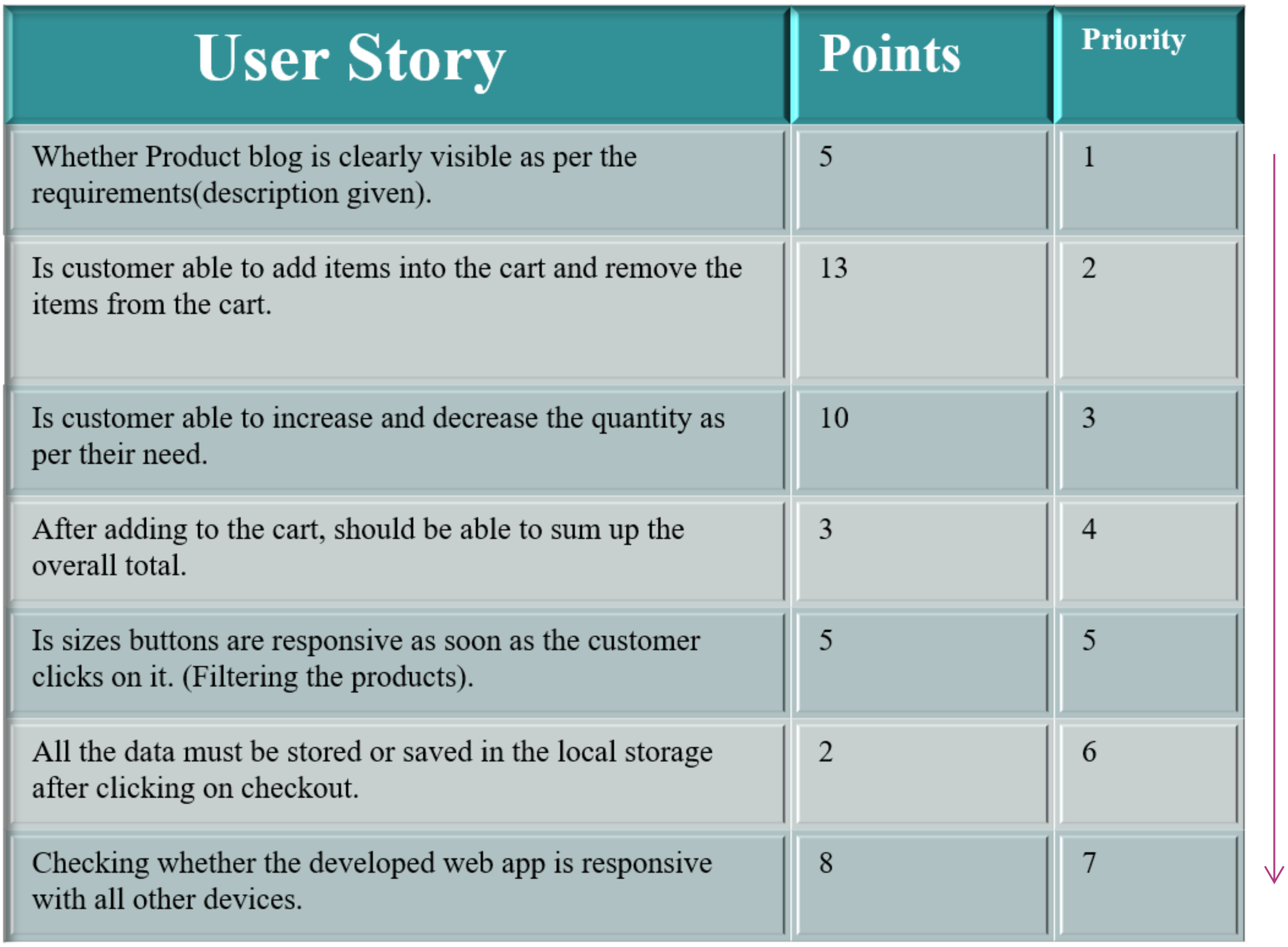
1. Easy and User Friendly to the customer who is using this application.
2. Single page web application.
3. Adding, removing the items, and clearing the cart products are very simple and easy to do.
4. The Sum of total is visible in the cart page itself as soon as the customer updates the quantity.
5. Also, user can get to know the number products added to the cart as soon as the user clicks on Add-to-Cart button.

2.3 Purpose of this Document

The main purpose of the requirement document is to understand the requirements specification or document for the system engineers or the software developers to understand the system developed and for the system customer to specify the requirement and read them to check that they meet necessary needs**.**

2.4 REQUIREMENTS: -

2.4.1 PRODUCT BACKLOG



User Story	Points	Priority
Whether Product blog is clearly visible as per the requirements(description given).	5	1
Is customer able to add items into the cart and remove the items from the cart.	13	2
Is customer able to increase and decrease the quantity as per their need.	10	3
After adding to the cart, should be able to sum up the overall total.	3	4
Is sizes buttons are responsive as soon as the customer clicks on it. (Filtering the products).	5	5
All the data must be stored or saved in the local storage after clicking on checkout.	2	6
Checking whether the developed web app is responsive with all other devices.	8	7

2.4.2 Functional Requirements –

Sl No.	Requirements	Description
1.	**Add to cart**	The user must be able to add the products into the cart.
2.	**Remove**	The user must be able to remove the products from the cart.
3.	**Filter**	Based on the available sizes, the user must be able to view the products available on the screen as soon as the user clicks on the sizes**.**
5.	**Checkout**	As soon as the user clicks on the checkout button, the data must get stored in the local storage database.

2.4.3 Non-Functional Requirements-

Images	Json data

2.4.4 Hardware Requirements:

- Minimum 2GB RAM

2.4.5 Software Requirements:

- HTML
- CSS
- Vuejs
- Vuex
- Local Storage
- Bootstrap
- VS Code

2.4.6US1:
Acceptance Criteria:

When I visit the website

Then I must be able to view the Products/clothes available in the blog.

US2:
Acceptance Criteria:

Given: A list of clothes as given in the documentary

When I click on sizes button

Then I must be able to view the filtered products based on sizes available.

US3:
Acceptance Criteria:

Given: A list of clothes as given in the documentary

When I click on sizes button

Then I must be able to select multiple sizes at a time.

US4:
Acceptance Criteria:

Given: A list of clothes as given in the documentary

When I click on the cart button

Then The products must go to the cart.

US5:
Acceptance Criteria:

Given: A list of clothes as given in the documentary

When I click on the cart

Then I must be able to view the products added into the cart and total summary of shopping.

US6:
Acceptance Criteria:

Given: A list of clothes as given in the documentary

When I click on checkout

Then I must be able to store the shopping data into the local storage.

3. SYSTEM DESIGN

3.1. SYSTEM DESIGN

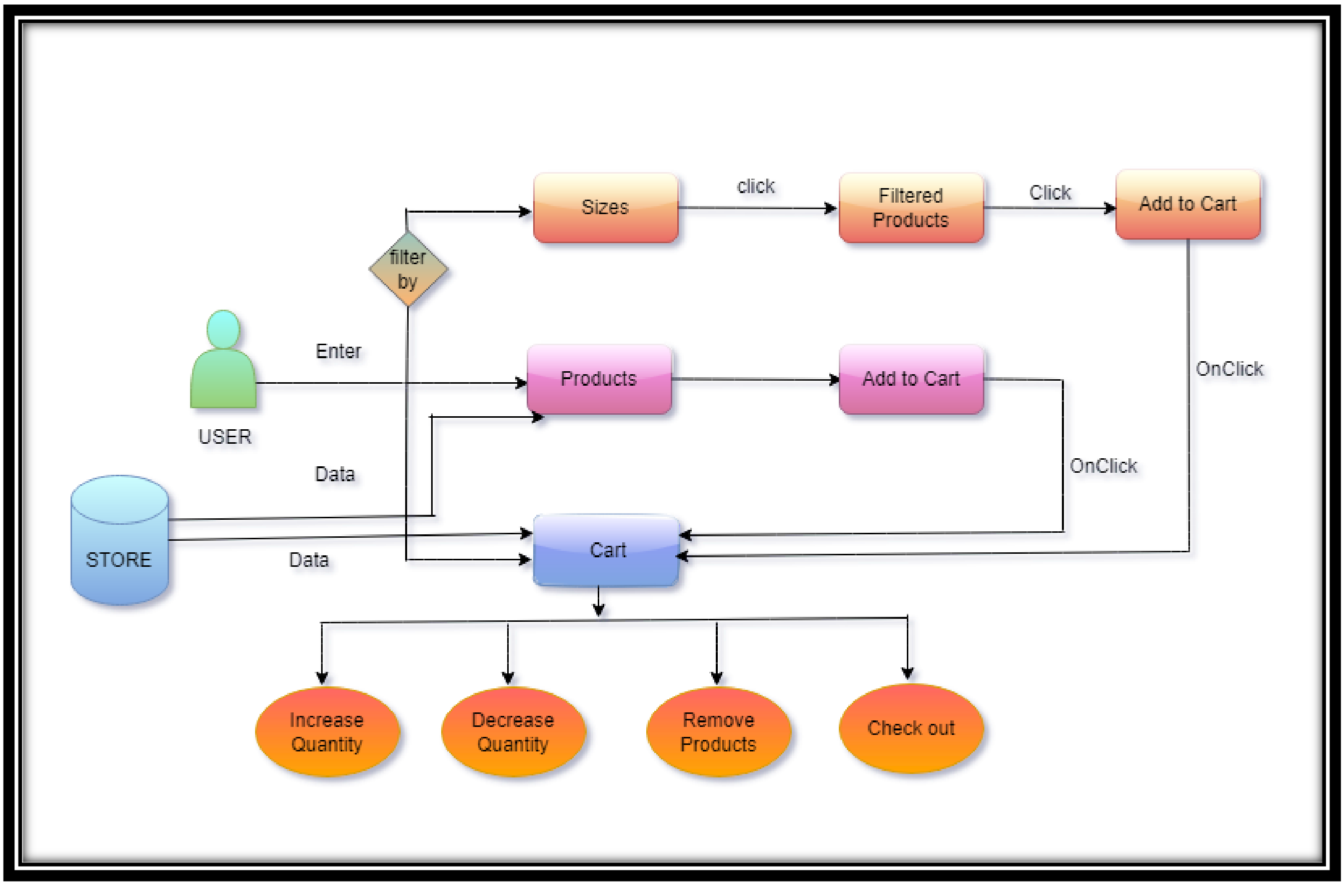


Fig 3.1: **SYSTEM DESIGN**

The above figure is the system design of our application. As soon as the user enters the app, the user can see a single page cart where sizes, product blog and cart are visible to them. If the user wants the products according to the sizes, then they can click on the available sizes button and filter around and can choose their products. If not, they can also directly add to the cart from the products view as soon they enter app. Later, the liked products can be added to cart by clicking on the “add-to-cart” button and all those items goes into the cart. In the cart the user can increase/decrease the quantity of the products or else they can also remove the product if they dislike. In the cart blog itself the user can view their overall total. Later when the user clicks on the checkout button, the whole summary comes and gets stored in the local storage.

3.2 USE CASE DIAGRAM

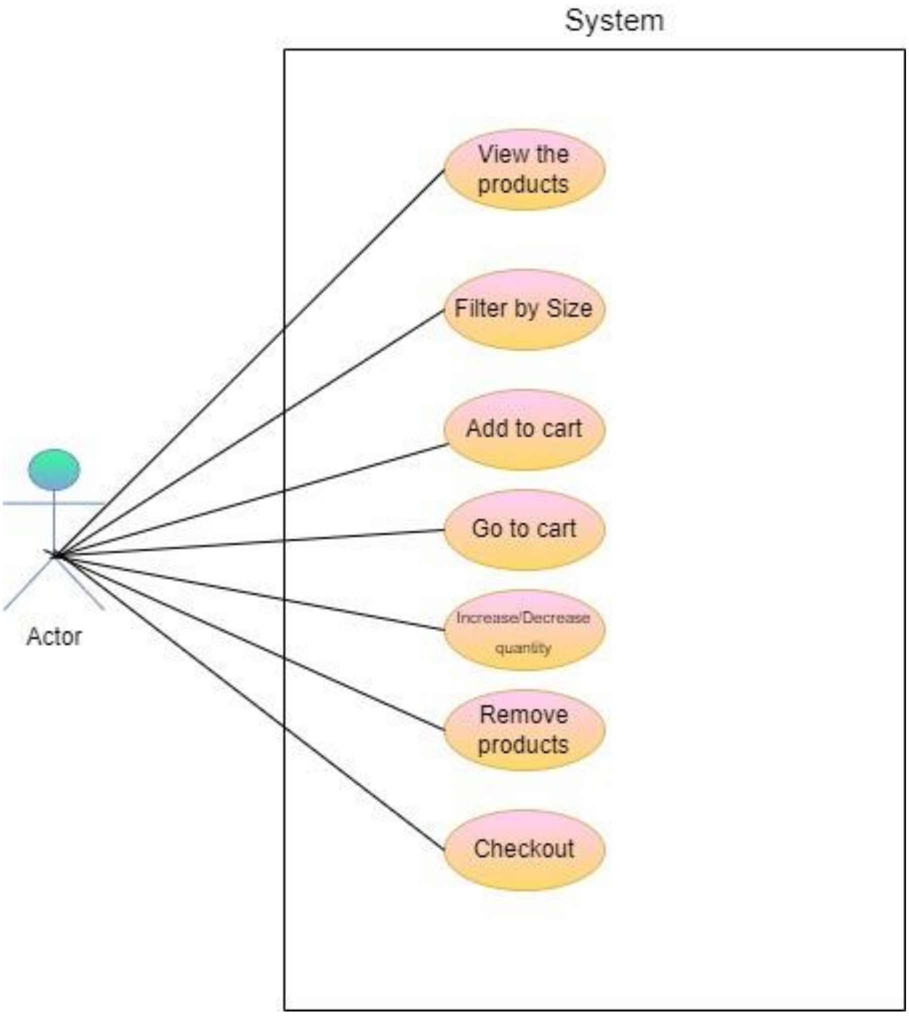


Fig 3.2: **User Case Diagram**

Here in the above User Case Diagram, the user/actor can be able to view the products, can also view the products by filtering the clothes according to the sizes available. User can add the liked products into the cart, later the user can also view the summary of their purchase in the cart and also can increase or decrease the quantity whenever they prefer, if they do not like the products, then they can also remove the products from the cart itself. User can view the total sum of their shopping in the cart page itself, as soon as the user clicks on the checkout button, the whole summary of the shopping gets stored in the local storage for future references.

3.3 FLOW DIAGRAM

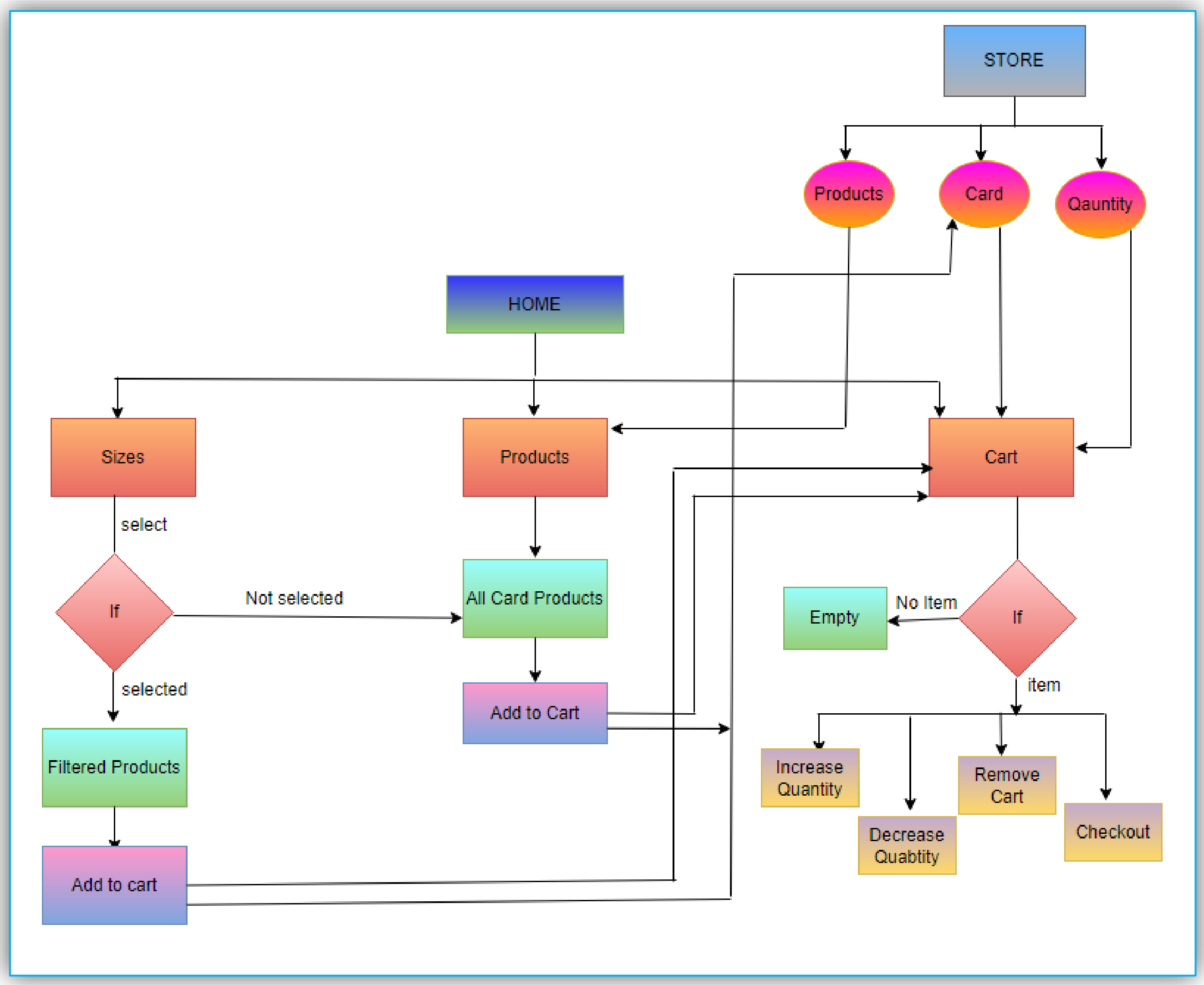
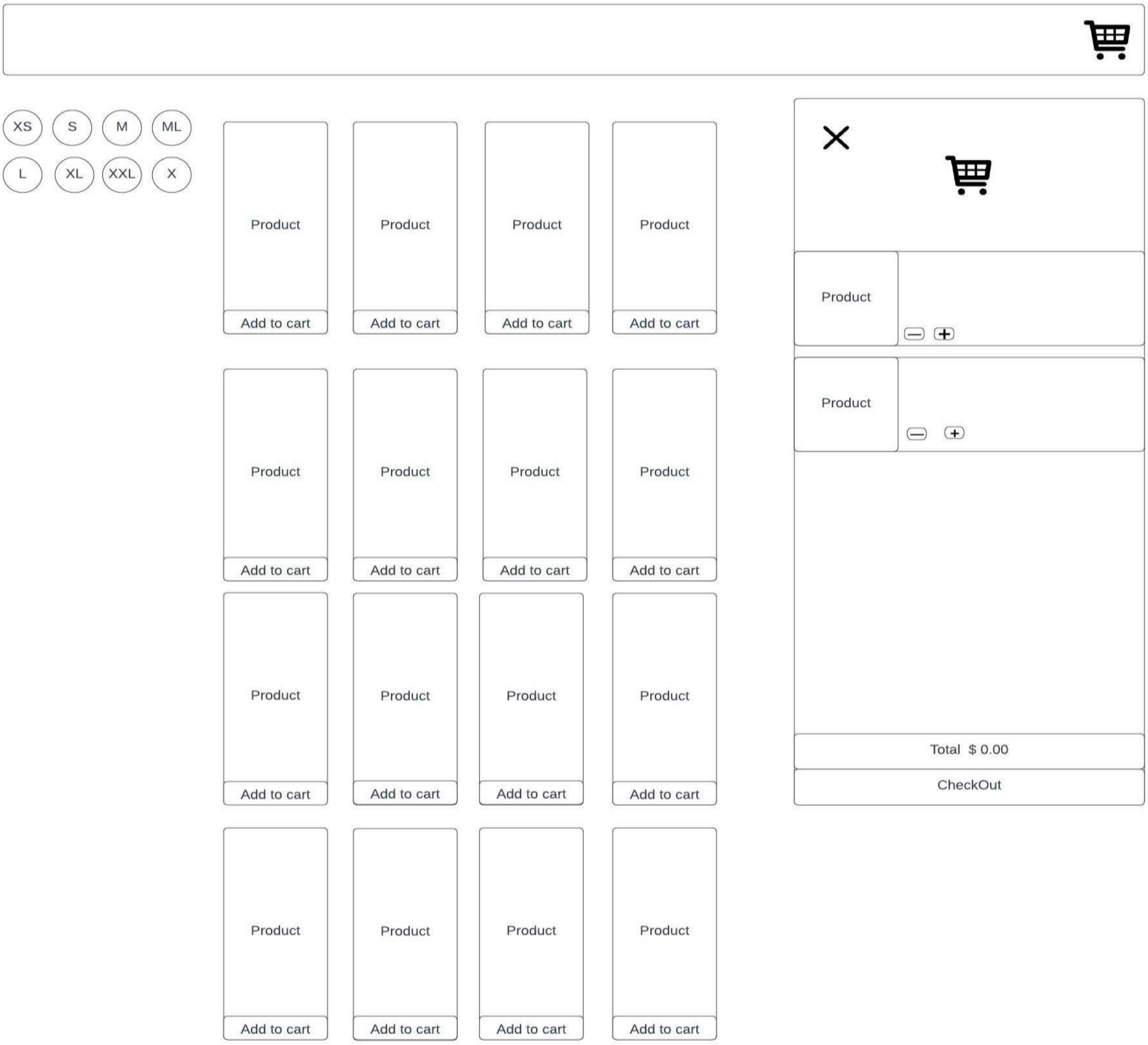
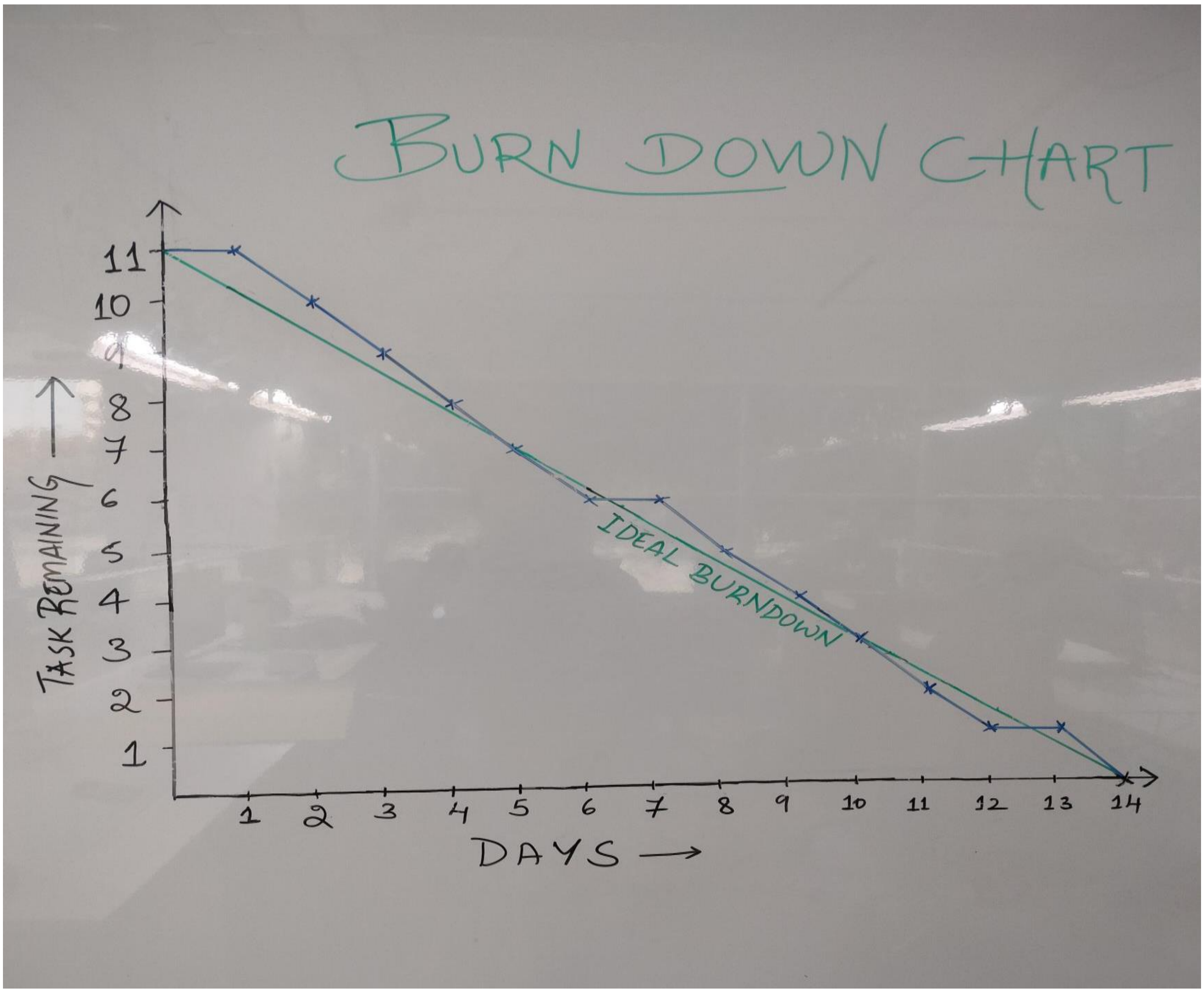


Fig 3.3: **FLOW DIAGRAM**

3.4 User Interface Template

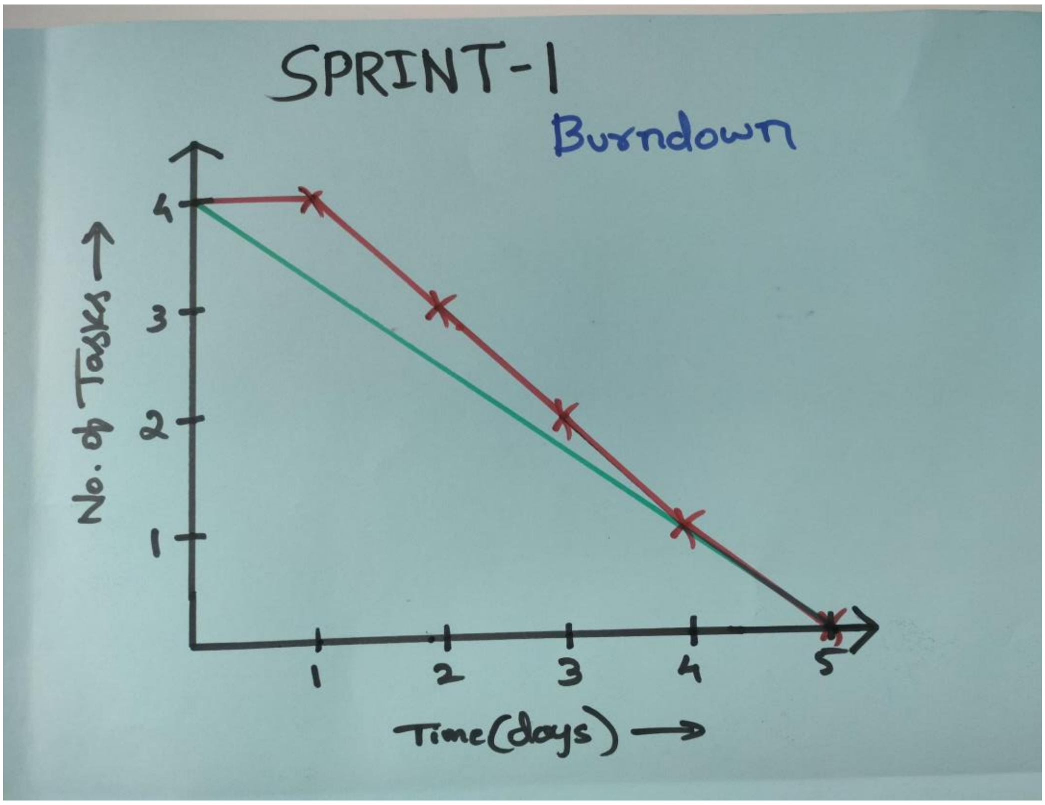


3.5 BURNDOWN CHART



- We planned this project for 3 weeks i.e. (14 days) with 11 tasks.
- There were totally 3 sprints as shown below, sprints 1 & 2 had 5 days with 4 tasks to do and the last sprint had 4 days with 3 tasks only.
- The actual burndown how it went on is captured and pasted here. It went according to that successfully and by end of 14th day our project was ready.

3.5.1 SPRINT-1 BURNDOWN



No. of days – 5

No. of Tasks – 4

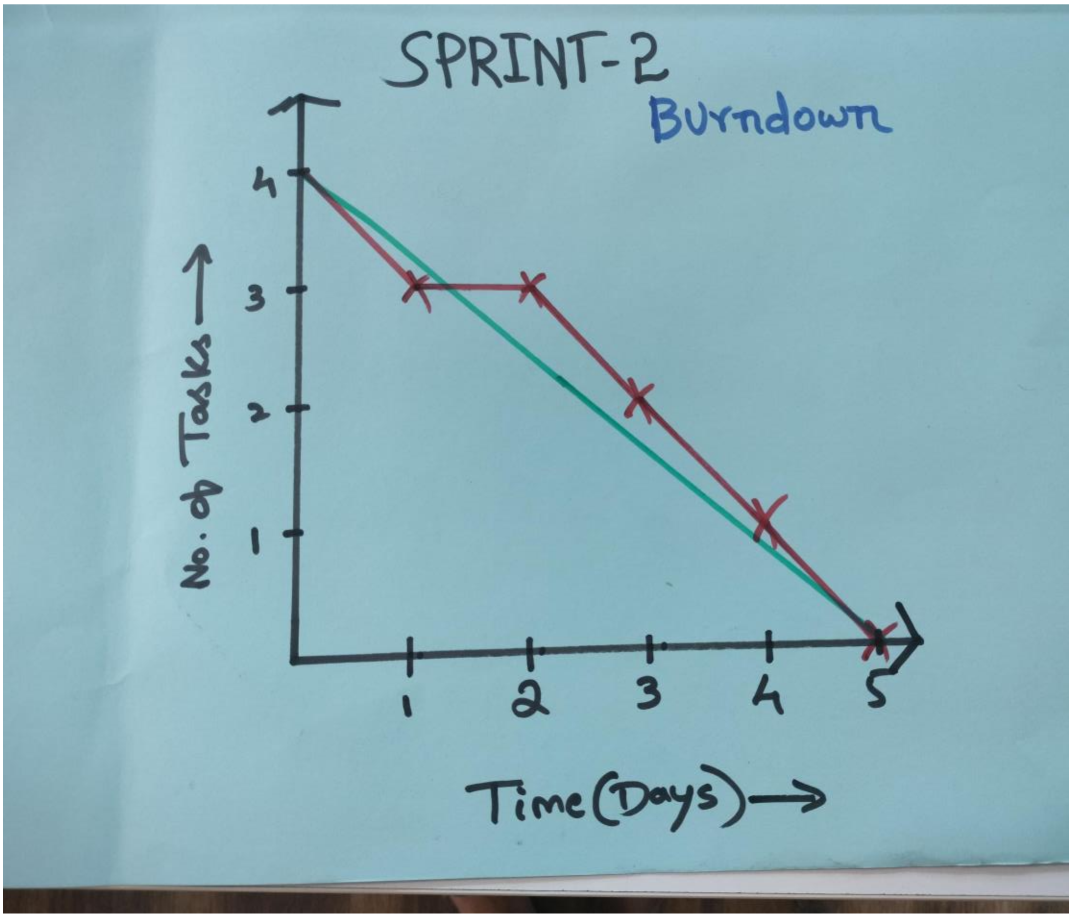
Activities – done with header, product view, cart add and remove.

In the 1st sprint it totally involved 5 days with 4 tasks only. Since the given Technology i.e Vuejs was new to us, we had to go through the documentation from the scratch and then move forward. Later on the work was divided amongst us and we were able to complete the header view, product view along with add to cart and remove products was done (cart page).

Drawback : Technology

We overcame it that by going through the documentation, some references in google**.**

3.5.2 SPRINT-2 BURNDOWN



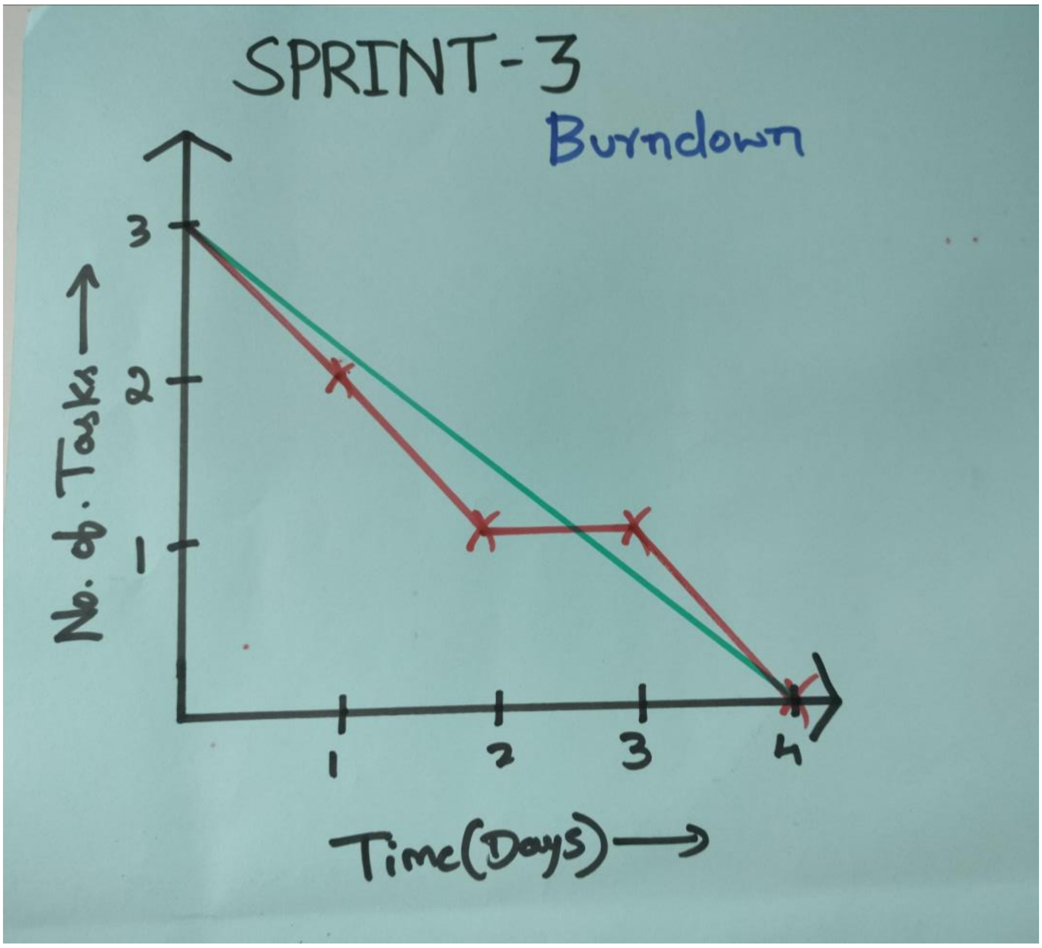
No. of days – 5

No. of Tasks – 4

Activities – done with cart calculation, local storage, sizes button, remove products.

In the 2nd sprint it again involved 5 days with 4 tasks. Now as we were bit aware of the technology, the process went on well, we completed some computations along with total calculations, local storage for storing the final data and we were done with sizes button.

3.5.3 SPRINT-3 BURNDOWN



No. of days – 4

No. of Tasks – 3

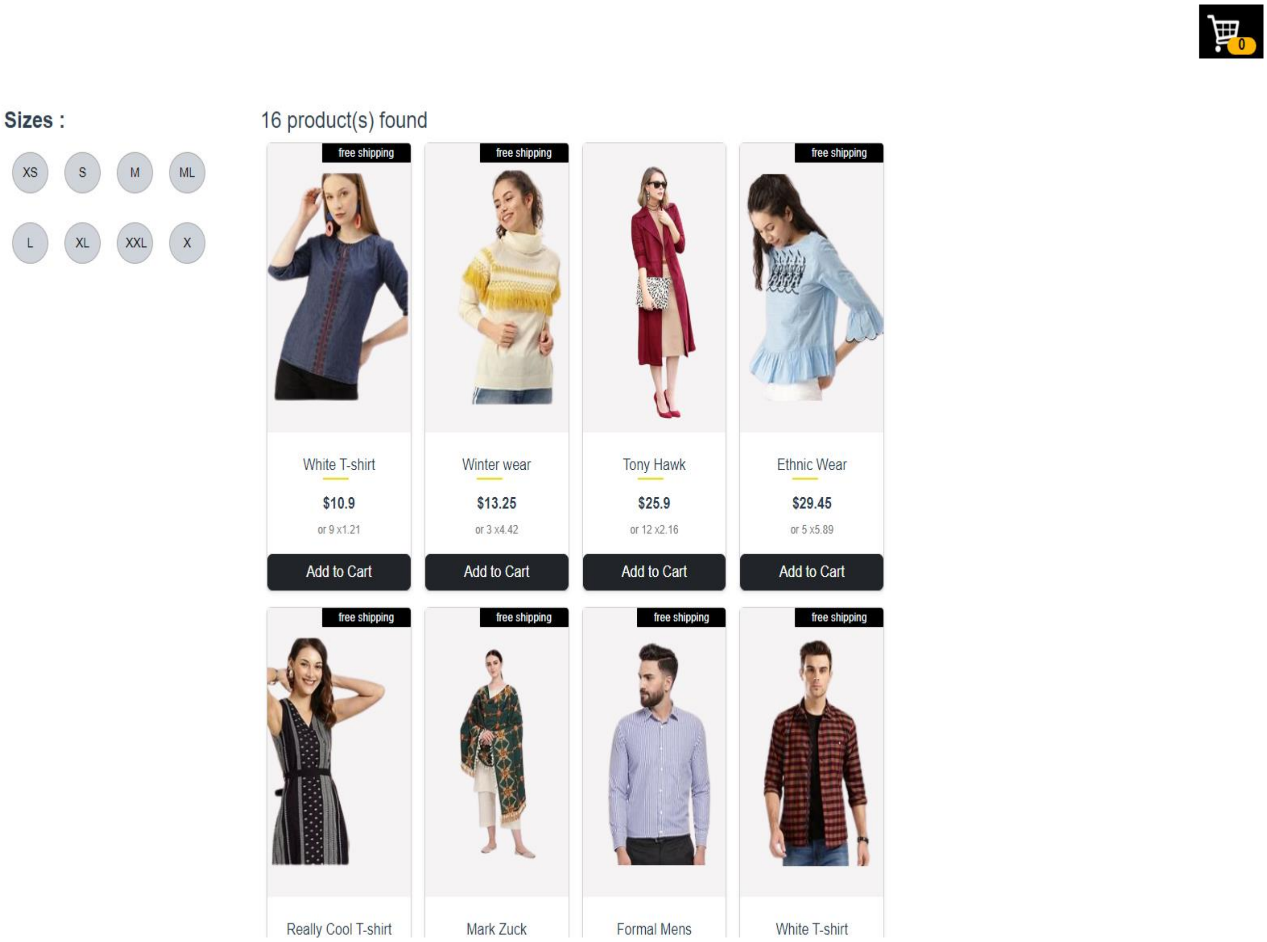
Activities – done with Filtering the products and making it responsive.

In 3rd sprint, it involved 4 days with 3 tasks. This was an actually an permission for extension of the deadline. Here we faced a major problem in Filtering the products according to the sizes available. We were unable to figure out what was going wrong. Then we contacted the SME(Asif), they helped us in figuring in out the problem and helped us. Then some how we were able do the filtering. At the end we made the whole project to look into responsive. Now the project can be deployed/used in whichever devices you prefer.

Major Problem: Filtering the products.

Solution: Asif helped us.

5. SNAPSHOTS:



16 product(s) found

White T-shirt
$10.9
or 9 x1.21

Winter wear
$13.25
or 3 x4.42

Tony Hawk
$25.9
or 12 x2.16

Ethnic Wear
$29.45
or 5 x5.89

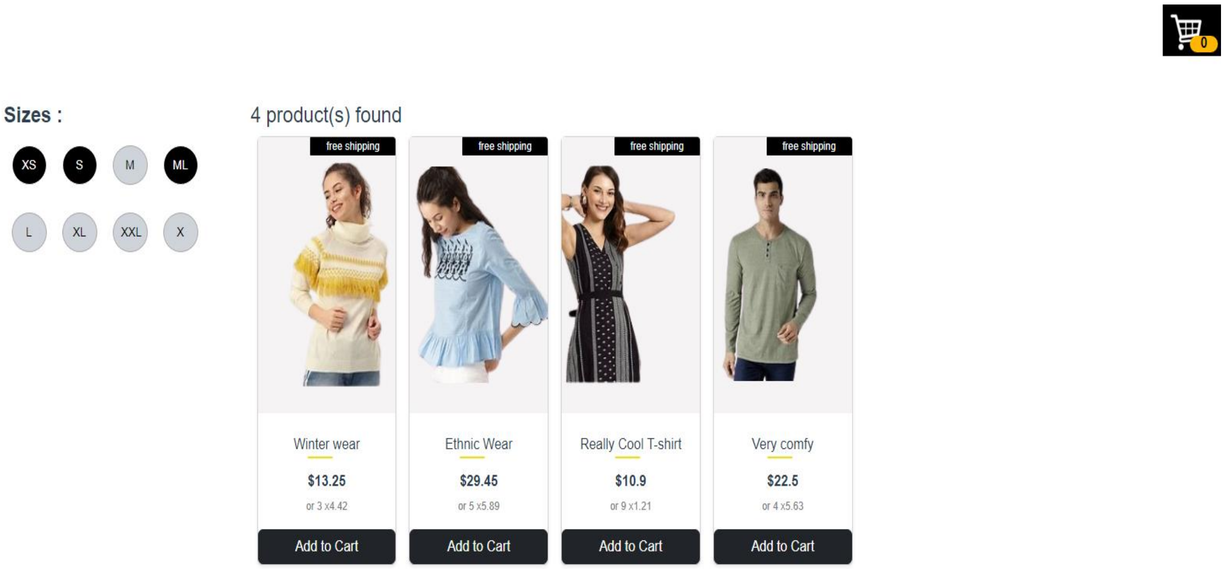
Really Cool T-shirt

Mark Zuck

Formal Mens

White T-shirt

Fig 5.1: Home Page



Monday, December

Fig 5.2: Filtering View

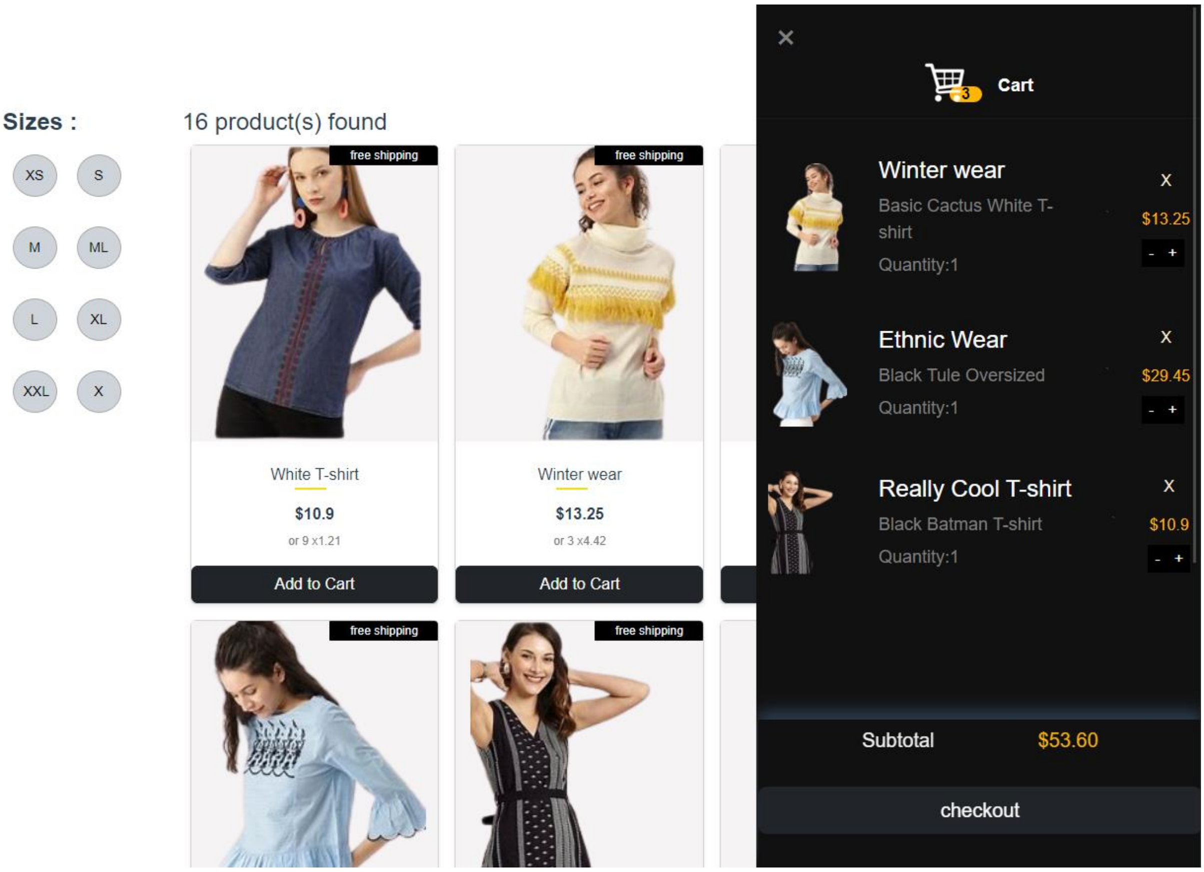


Fig 5.3: Cart View

6. TESTING

6.1 TEST CASES

SI.NO.	Module Name	Test Case No.	Test Case Description	Expected result
1.	Responsive	TC1	The user can get different screen resolutions accordingly.	All the expected resolutions of the screen are working.
2.	Add to cart	TC2	The user can add items to the cart.	Items are successfully added in the cart.
3.	Remove items	TC3	The user can remove items from the cart	Items are successfully removed from the cart.
4.	Checkout	TC4	The user can check out their products	Items are successfully checked out and details are stored in local storage.
5.	Go to cart	TC5	The user can go to the cart to view the order details.	The order details from the cart are displayed.
6.	Filtering	TC6	The user can filter the products according to the sizes available in the store.	The users can see the filtered products as soon as they select the sizes of their choices.
7.	Calculation	TC7	The sum of all the items.	Accurate and expected total is displayed as per the calculations.

Table 6.1

6.2 TEST REPORTS

SI.NO.	Test Case	Test status	Test Report
1.	TC1	The user can get different screen resolutions accordingly.	Pass
2.	TC2	The user can add items to the cart.	Pass
3.	TC3	The user can remove items from the cart	Pass
4.	TC5	The user can see the selected products in the cart.	Pass
5.	TC6	The user can view the filtered products according to the sizes available.	Pass

Table 6.2

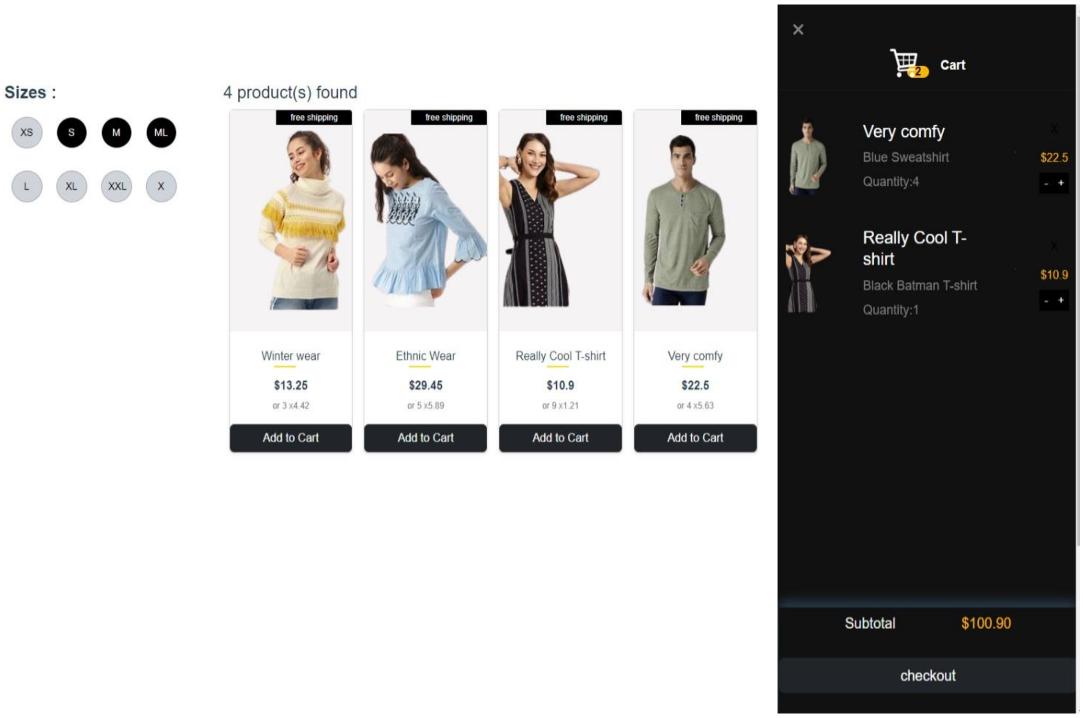
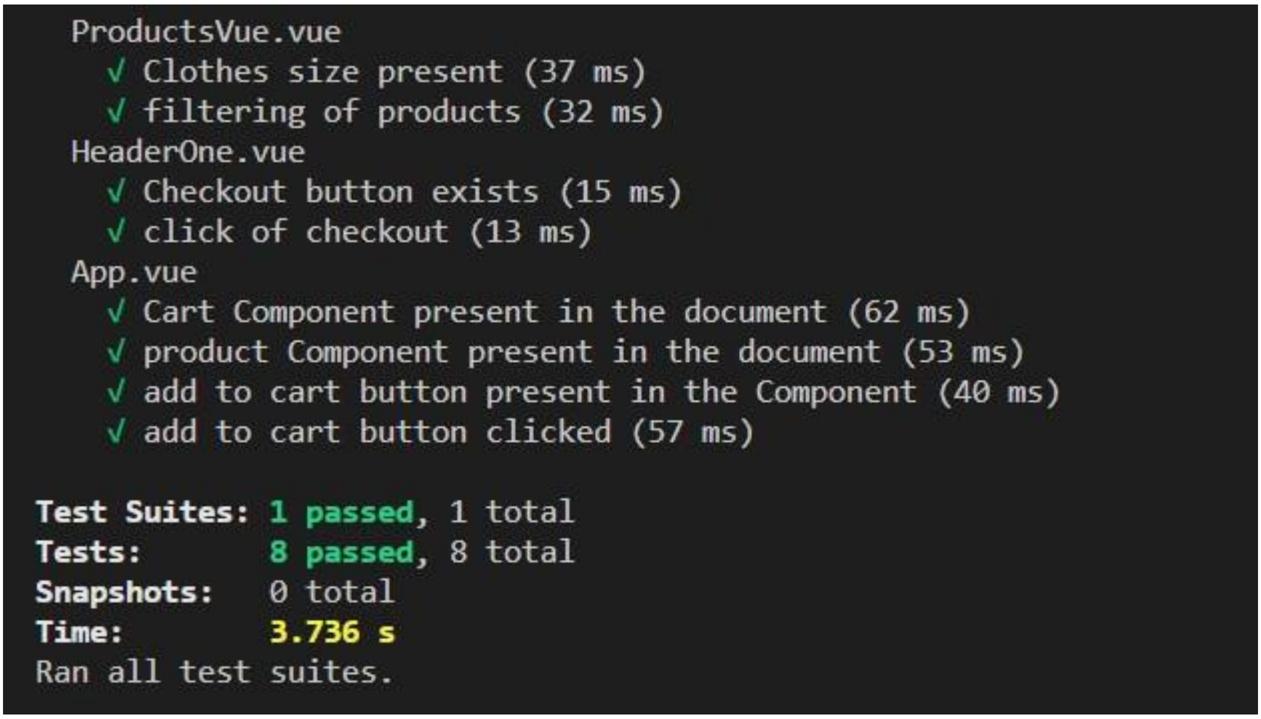


Fig 6.1



```
  ProductsVue.vue
    √ Clothes size present (37 ms)
    √ filtering of products (32 ms)
  HeaderOne.vue
    √ Checkout button exists (15 ms)
    √ click of checkout (13 ms)
  App.vue
    √ Cart Component present in the document (62 ms)
    √ product Component present in the document (53 ms)
    √ add to cart button present in the Component (40 ms)
    √ add to cart button clicked (57 ms)

Test Suites: 1 passed, 1 total
Tests:       8 passed, 8 total
Snapshots:   0 total
Time:        3.736 s
Ran all test suites.
```

Fig 6.2

7. DEPLOYMENT

7.1 STEPS FOR DEPLOYMENT: -

1.Build the current project for production using the command "npm run build"

2.this creates a folder named "Dist" which is a minified version of the project and ready for deployment

3.Create a local git repository using the command 'git init'

4.add all the project files (from current working directory) to this repository using the command 'git add.

5.commit the changes in repository using the command 'git commit -m "message"'

6.create a remote repository in GitHub where the project will be stored

7.add the local repository to the remote repository using the following commands

"Git remote add origin 'the gilt repo link'. git"

"Git branch -M main"

"Git push -uf origin main"

8.deploy the project in netlify www.netlify.com

9.sign up with the GitHub in which the project is stored.

10.to deploy into netlify provide the repository in which the project is stored using the option "import from git".

11.provide the build command and built file name and click on deploy.
12.the link for the deployed project is ready Shopping Kart (shopphub.netlify.app)

8. RETROSPECTIVE FINDINGS

What went on well?

The workflow went on according to our planning itself. Teamwork was good. Each and every one was helping each other, if the other person in the group facing the problem.

In the 1st sprint, we found the given technology was very challenging, since we were not that much aware of the given technology, so the work progress went very slowly. We had to go through the documentation and some research. But then also we managed to complete 4 tasks. With the help of Bootstrap, CSS and JavaScript, UI part went very well.

In 2nd sprint, the work went on normally, but then also we found some difficulties in computation and calculation part. We all went through the documentation and somehow, we cracked it.

In 3rd sprint we faced a major issue in filtering the products according to the sizes, then we asked for an extension and some help from the SME, then Asif helped us what was going wrong and where, and then we were able to proceed further with our project.

Totally, each member worked in a group and if anyone in the facing problem, all others would come and help them in solving that problem.