|  |  |  |
| --- | --- | --- |
| VILNIAUS KOLEGIJA  UNIVERSITY OF APPLIED SCIENCES  FACULTY OF ELECTRONICS AND INFORMATICS  https://screenshotscdn.firefoxusercontent.com/images/eaf3f7f3-2952-4801-af5c-4f20e8ae8b88.png | | |
|  | | |
| **ECOMMERCE WEBSITE PROJECT** | | |
|  | | |
| GROUP WORK  6531BX028 PI18E | | |
| STUDENTS | (SIGNATURE) | EDITA KOMAROVA    DŽIUGAS PEČIULEVIČIUS |
| 2021-11 |
| (SIGNATURE) |
| 2021-11 |
|  |
|  |
|  |
| LECTURER | (SIGNATURE) | JŪRATE VAIČIULYTĖ | |
|  | 2021-11 |  |
|  |  |  |
| 2020 | | |

Table of Contents

[1 SOFTWARE ARCHITECTURE DIAGRAM 3](#_Toc57376500)

[2 SOFTWARE DESIGN MODEL 7](#_Toc57376501)

[2.1 Functional requirements traceability table 7](#_Toc57376502)

[2.2 Entity-relationship diagram 8](#_Toc57376503)

[2.3 Structure diagram 8](#_Toc57376504)

[2.4 System behavior diagrams 9](#_Toc57376505)

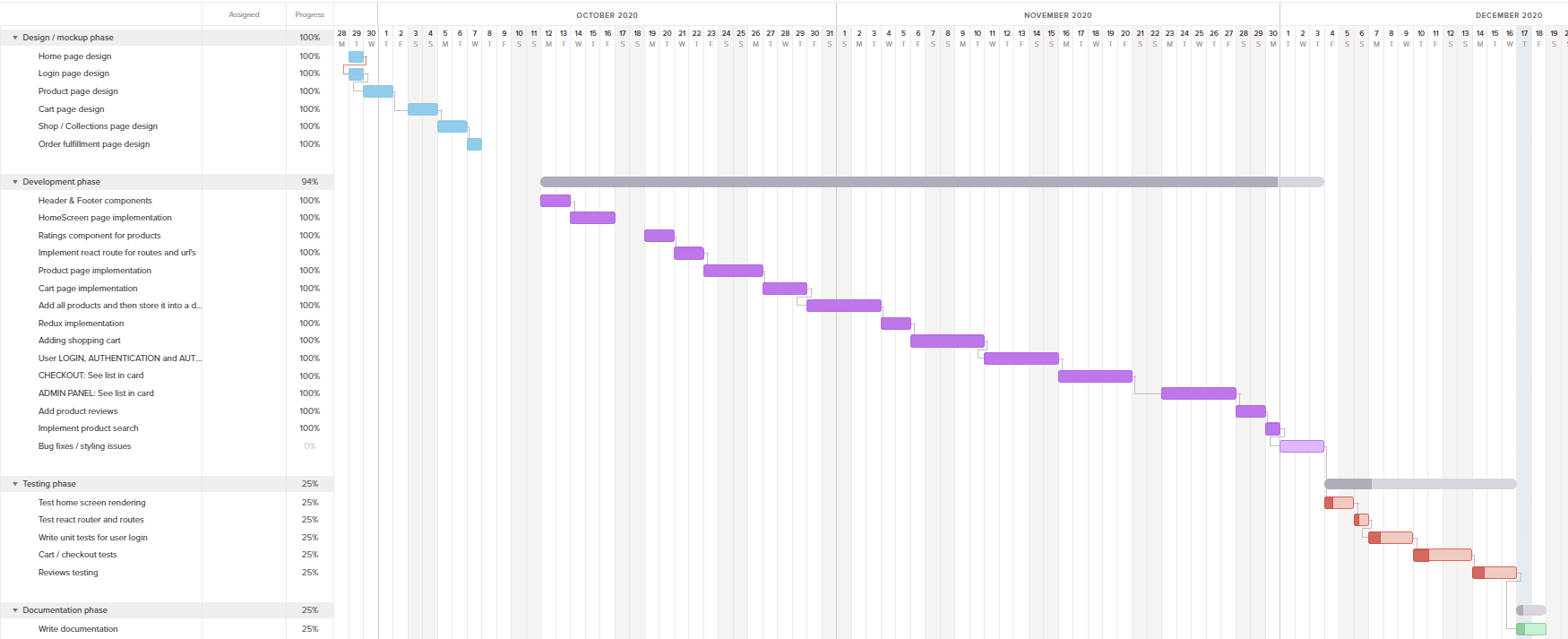
[3 UPDATED PROJECT SCHEDULE 16](#_Toc57376506)

# INTRODUCTION

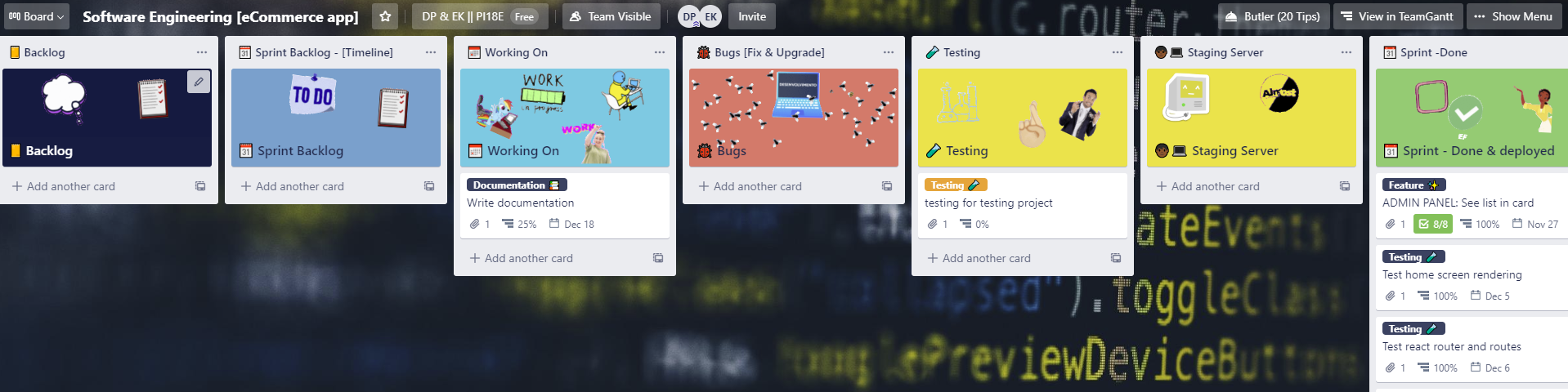
Ecommerce application which was built from scratch, it allows any user to find his/her products, use all cart features, make orders, leave reviews and lots more.

# DESCRIPTION OF THE PROJECT PLAN

This is our project plan binded to our Trello board.



Picture 1 - teamgantt



Picture 2 - Trello board

# FUNCTIONAL AND NON-FUNCTIONAL SOFWARE REQUIREMENTS

## Functional requirements

Diagram

Description automatically generated

Picture 3 - Use-case diagram for functional requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **FUNCTIONAL REQUIREMENT** | **USER REQUIREMENT** | **USE-CASE** | **PRIORITY** |
| FR1, FR 7 | View | User, Admin | UC01 | 1 |
| FR2 | Add Item to the Cart | User | UC02 | 2 |
| FR3 | Login | User | UC03 | 3 |
| FR4 | Buy Product | User | UC04 | 4 |
| FR5 | Search Products | User | UC05 | 11 |
| FR6 | Write Reviews | User | UC06 | 10 |
| FR7 | Login | Admin | UC | 5 |
| FR8 | View Orders | Admin | UC07 | 6 |
| FR9 | Update Order | Admin | UC08 | 7 |
| FR10 | Edit/Delete Products | Admin | UC09 | 8 |
| FR11 | Add Products | Admin | UC10 | 9 |

Make sure that the functional requirements are formulated correctly: they are easy to understand, clear, unambiguous, implementable, verifiable.

Describe the use cases.

|  |  |
| --- | --- |
| **Function** | View |
| **ID** | UC01 |
| **Description** | User/Admin can look at the products |
| **User roles** | User, Admin |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | Click View Products |
| **Outputs** | Product List |
| **Action** | 1. The user enters the website 2. User click Browse Products |
| **Post-conditions** | After clicking View Products the user should be able to see the products. |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | Add Item to the Cart |
| **ID** | UC02 |
| **Description** | Customer can put item into the shopping cart |
| **User roles** | User |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | The product that customer desires. |
| **Outputs** | Desired product in the shopping cart. |
| **Action** | 1. The user enters website.  2. The user click Browse Products.  3. The user click the desired product.  4. In the Product Description user click Add item to the Cart.  5. The product appears in the cart. |
| **Post-conditions** | The product appears in the cart. |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | Login |
| **ID** | UC03 |
| **Description** | Customer can login into the system |
| **User roles** | User |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | Email and Password |
| **Outputs** | User succesfully logins in the system. |
| **Action** | 1. The user enters website.  2. The user click Login.  3. The user click enters their Email and Password.  4. User sees that sees succesfully log in. |
| **Post-conditions** | User is succesfuly log in |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | Buy Product |
| **ID** | UC04 |
| **Description** | Customer can buy the desired product |
| **User roles** | Registered User |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | Address, Bank information. |
| **Outputs** | Confirmation of the order window. |
| **Action** | 1. The user checks out item in the shopping cart.  2. The user enter their customers information.  3. The user click buy.  4. The user gets a window of order confirmation.  5. User‘s Order entered in the database. |
| **Post-conditions** | The user‘s order is entered into database |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | Search Product |
| **ID** | UC05 |
| **Description** | Customer can search for the desired Product |
| **User roles** | User |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | Keyword |
| **Outputs** | List of product of related products |
| **Action** | 1. The user enters website.  2. The user click the search bar.  3. The user enters their keyword.  4. The system shows a products related to the keyword. |
| **Post-conditions** | System shows products related to the keyword |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | Write Reviews |
| **ID** | UC06 |
| **Description** | Customer can leave reviews |
| **User roles** | Registered user |
| **Prie-conditions** | There is an internet connection, The user should be logged in. |
| **Inputs** | Message that user wants to leave, star rating |
| **Outputs** | Message appears in the reviews section. |
| **Action** | 1. User enters the website 2. User clicks Browse products. 3. User click on a chosen product. 4. User click to write review. 5. User leaves message and star rating. 6. The review appears in the product page. |
| **Post-conditions** | Review appears in the review section of the product |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | View Orders |
| **ID** | UC07 |
| **Description** | Admin can view orders |
| **User roles** | Admin |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | Click on orders page |
| **Outputs** | System outputs orders |
| **Action** | 1. Admin enter the website. 2. Admin logs-in 3. Admin clicks on orders 4. All orders appear on the screen |
| **Post-conditions** |  |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | Update Order Status |
| **ID** | UC08 |
| **Description** | Admin can update order status |
| **User roles** | Admin |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | Click on specific order |
| **Outputs** | Show order information |
| **Action** | 1. Admin enter the website. 2. Admin logs-in 3. Admin clicks on orders 4. All orders appear on the screen 5. Click on the order details 6. Click on mark as delivered. |
| **Post-conditions** | The system now shows that the order is delivered. |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | Edit/Delete Product listings |
| **ID** | UC09 |
| **Description** | Admin can edit or delete the product. |
| **User roles** | Admin |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | Click on products page |
| **Outputs** |  |
| **Action** | 1. Admin enter the website. 2. Admin logs-in 3. Admin clicks on products 4. All existing products appear on the page 5. Select the delete or edit button on the product |
| **Post-conditions** | The admin is able to edit or delete the product |
| **Side effects** | None. |

|  |  |
| --- | --- |
| **Function** | Add Product |
| **ID** | UC10 |
| **Description** | Admin can add product to the system. |
| **User roles** | Admin |
| **Prie-conditions** | There is an internet connection |
| **Inputs** | Product information : Product name, Description,Quantity,Price. |
| **Outputs** |  |
| **Action** | 1. Admin enter the website. 2. Admin logs-in 3. Admin clicks on products 4. All existing products appear on the page 5. Admin clicks „create product“ 6. Enters product information |
| **Post-conditions** | The admin is able to add product to the database. |
| **Side effects** | None. |

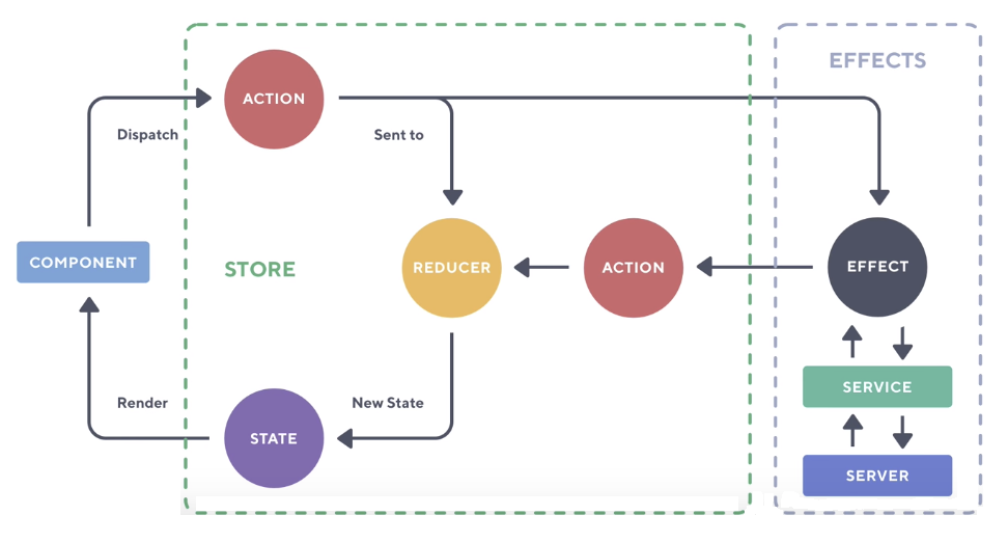
## Non-Functional requirements

1. Application should be written in JavaScript.
2. Application should use React framework.
3. The application should be hosted on Heroku.
4. The application (shop) data should be stored on Firebase or MongoDB.
5. The system should allow users to pay for their goods using Stripe or PayPal.

# SOFTWARE ARCHITECTURE DIAGRAM

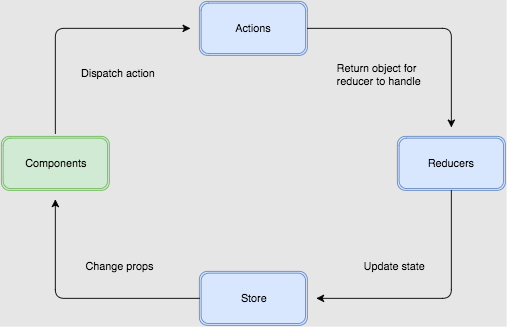
Software architecture diagram and its description. Basically react is not based on any architecture like MVC. Used Redux state management in this project.

Once we click a button, we trigger (dispatch) an action. An action in redux returns an object and gives it to a reducer to handle it. Those reducers are put into a store in one big combined reducer. And then we re-render our components depending on the state. There are also effects that are used when we have to make requests and fetch information from our backend.



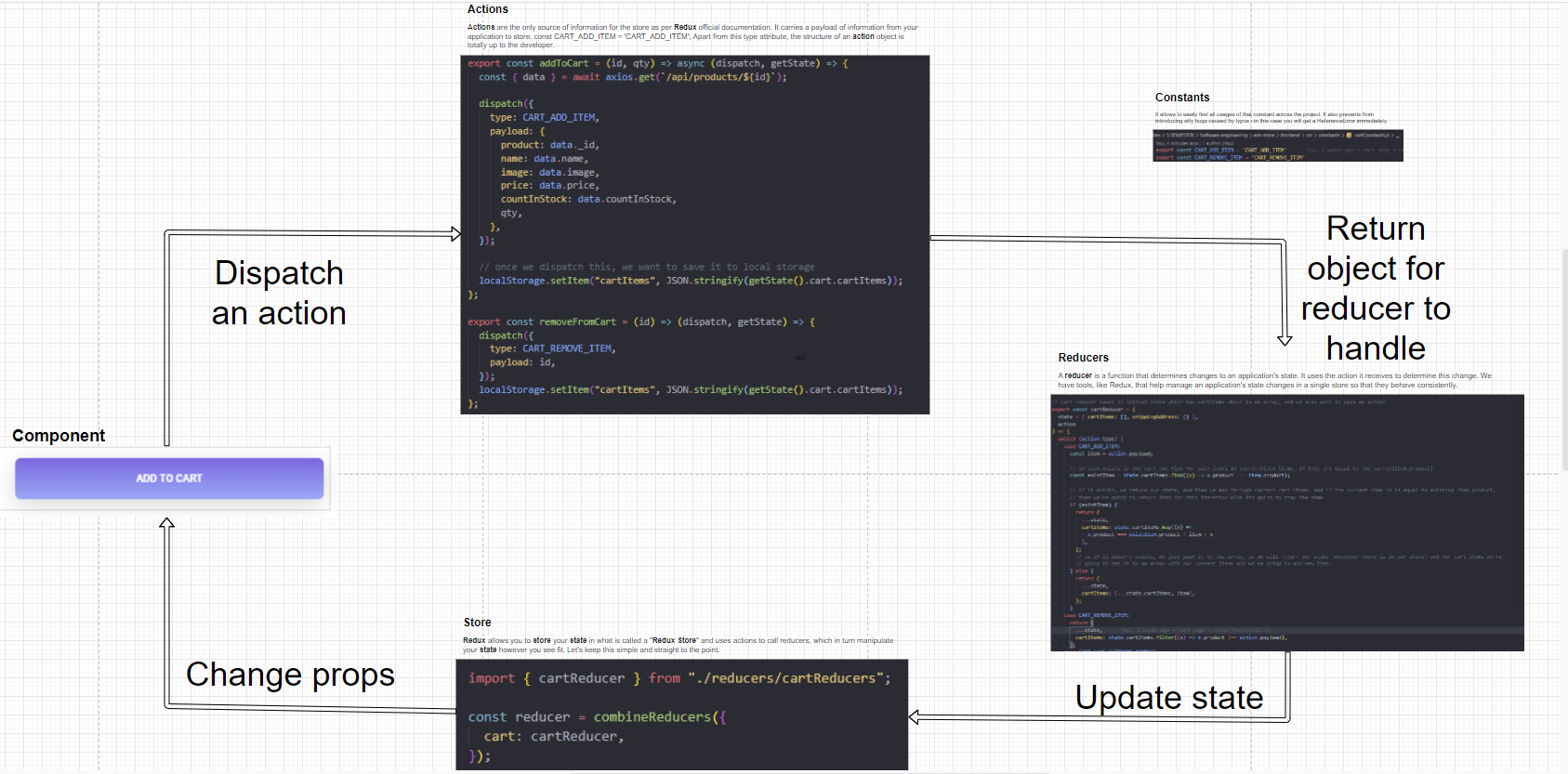
Picture 4 - Redux example 1

Here’s a simpler graph without any react effects.



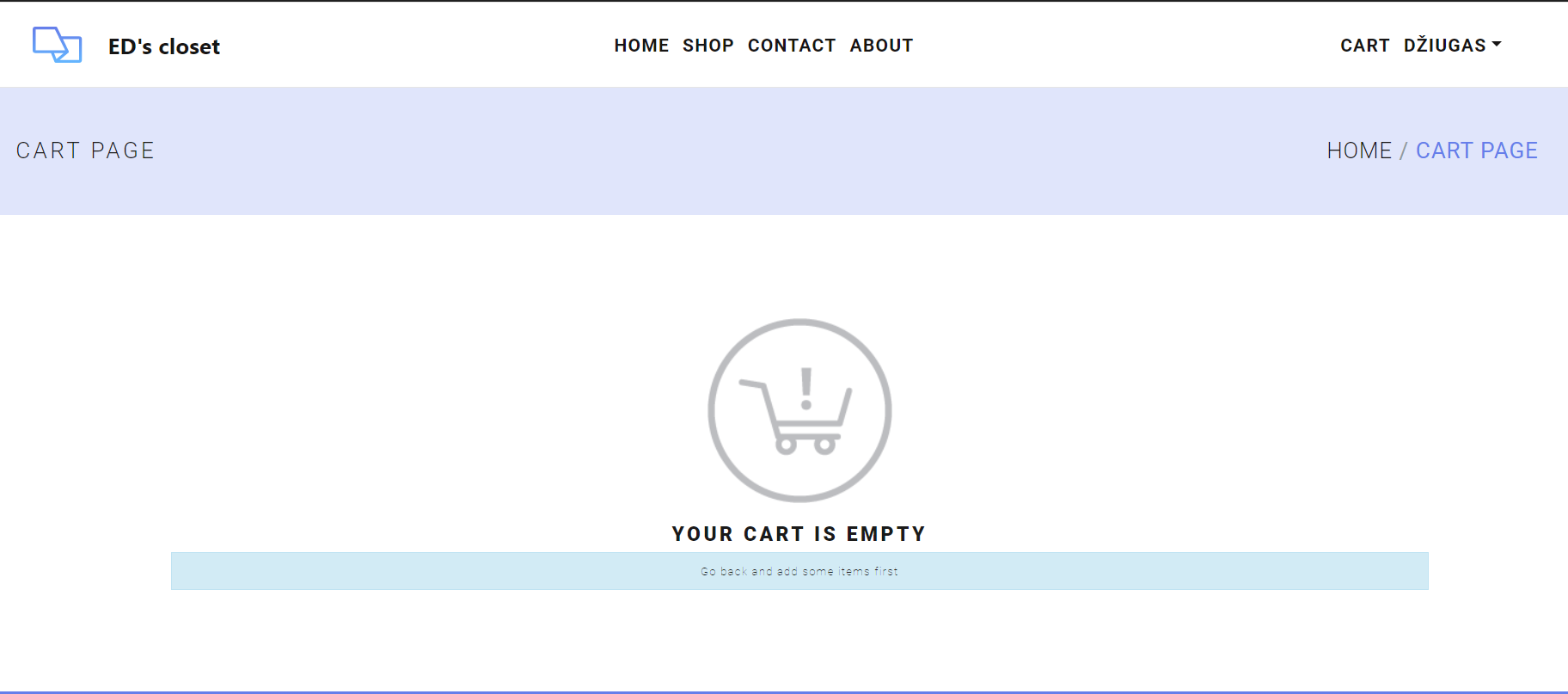
Picture 5 - Redux example 2

This is how add to cart flow looks like.

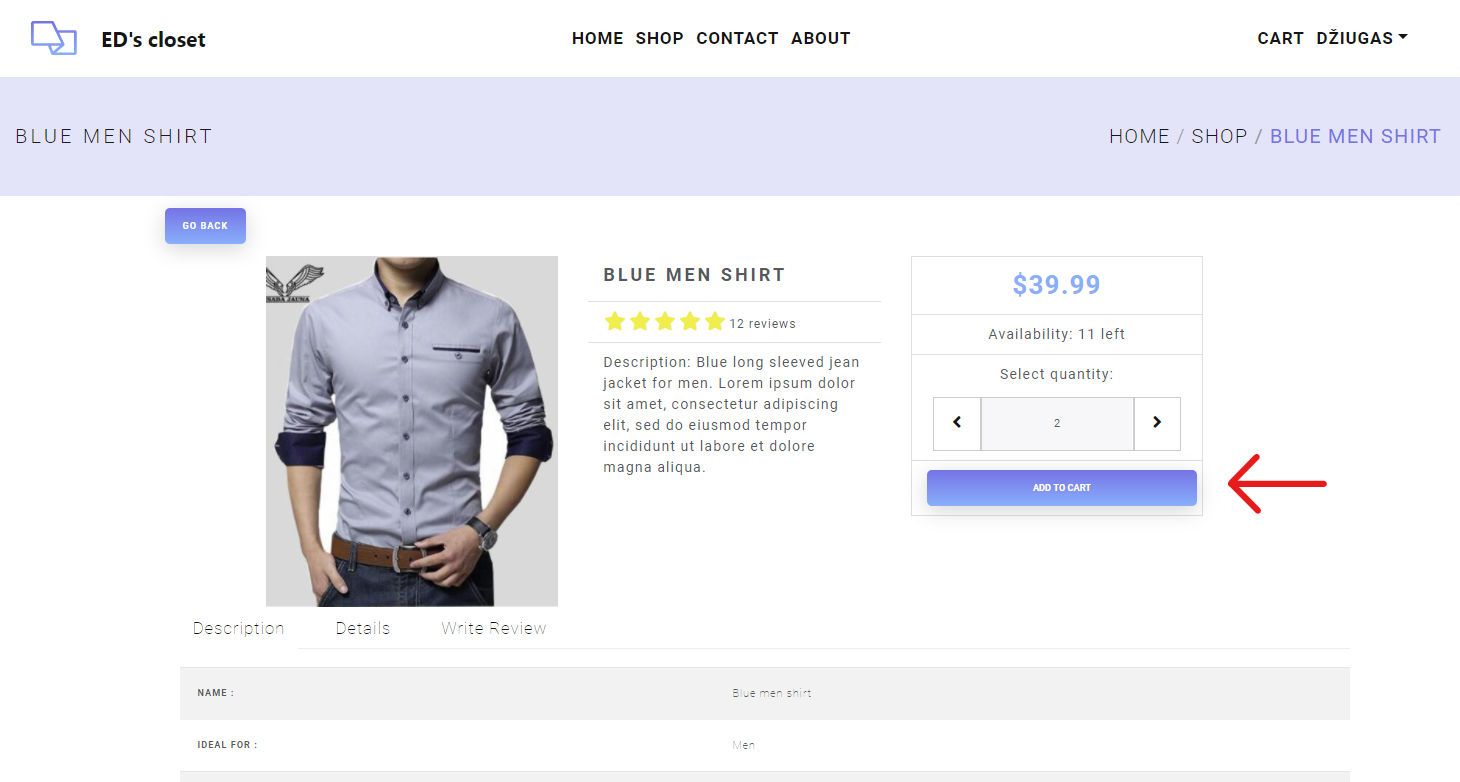


Picture 6 - Flow of redux for cart

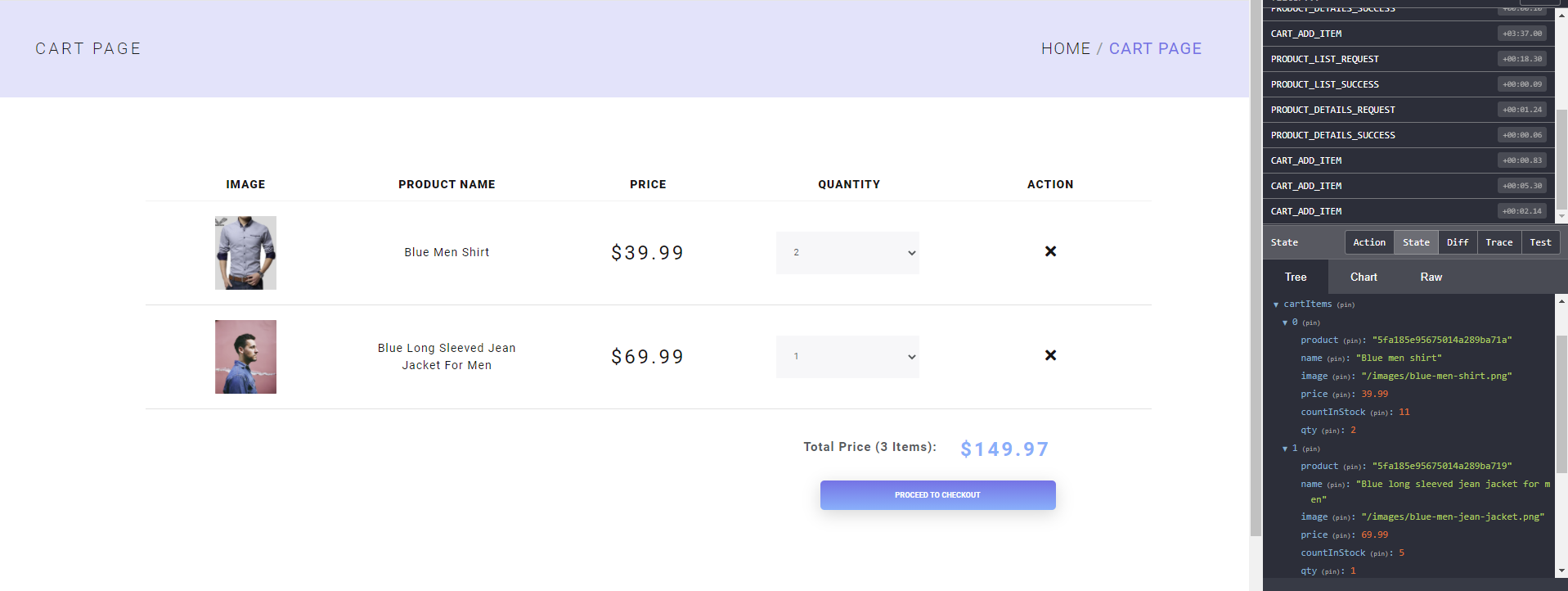
And here how it looks on the user end



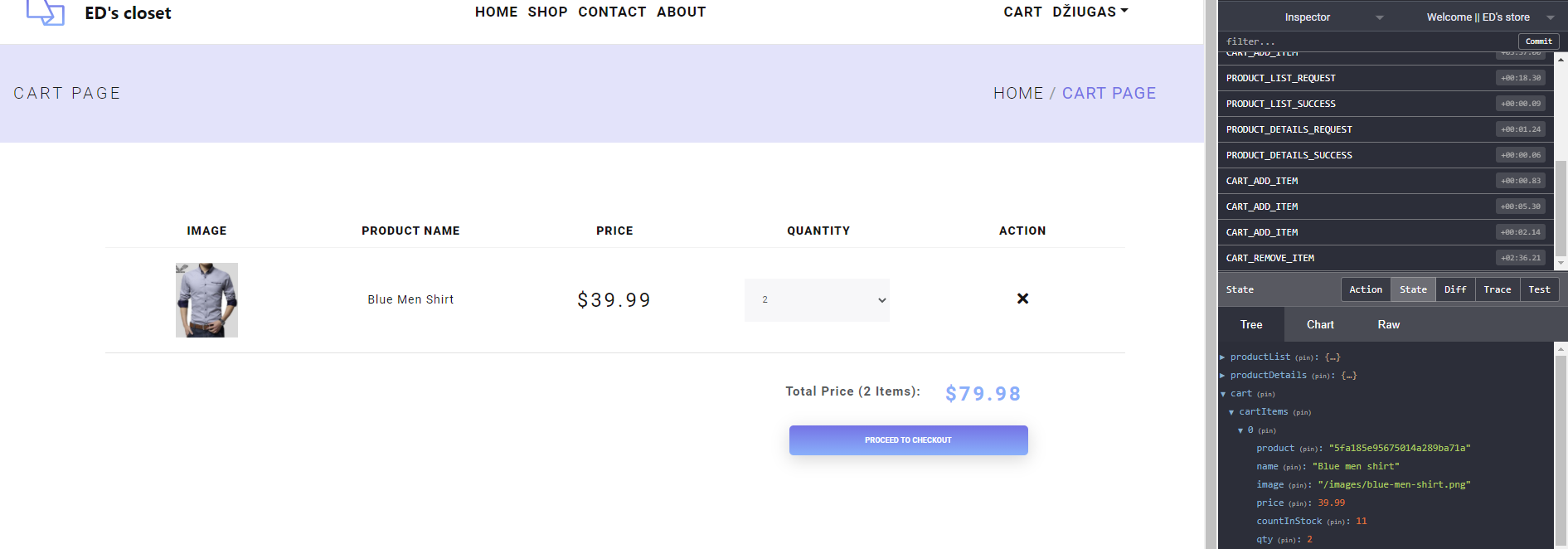
Picture 7 - Cart redux functionality example 1



Picture 8 - Cart redux functionality example 2



Picture 9 - Cart redux functionality example 3



Picture 10 - Cart redux functionality example 4

# SOFTWARE DESIGN MODEL

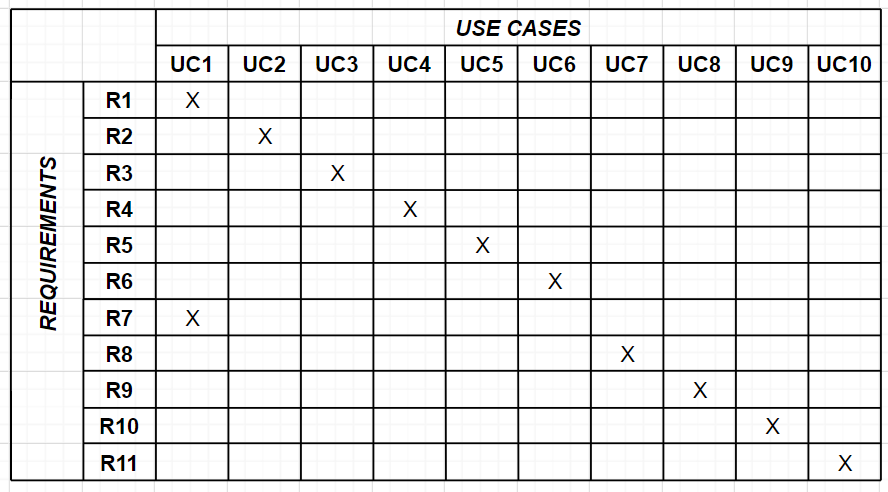
## Functional requirements traceability table

Functional Requirements:

1. The system should allow users to view product collections.
2. The system should allow users to add item to shopping cart.
3. The system should have Sign-in window where user can login.
4. The system should allow users to check out and buy an item.
5. The system should allow users to search for the product.
6. The system should allow users to read and write reviews.
7. Admin should be able to view products in the system.
8. Admin should be able to view orders in the system.
9. Admin should be able to update the order.
10. Admin should be able to edit/delete products.
11. Admin should be able to add products into the system.

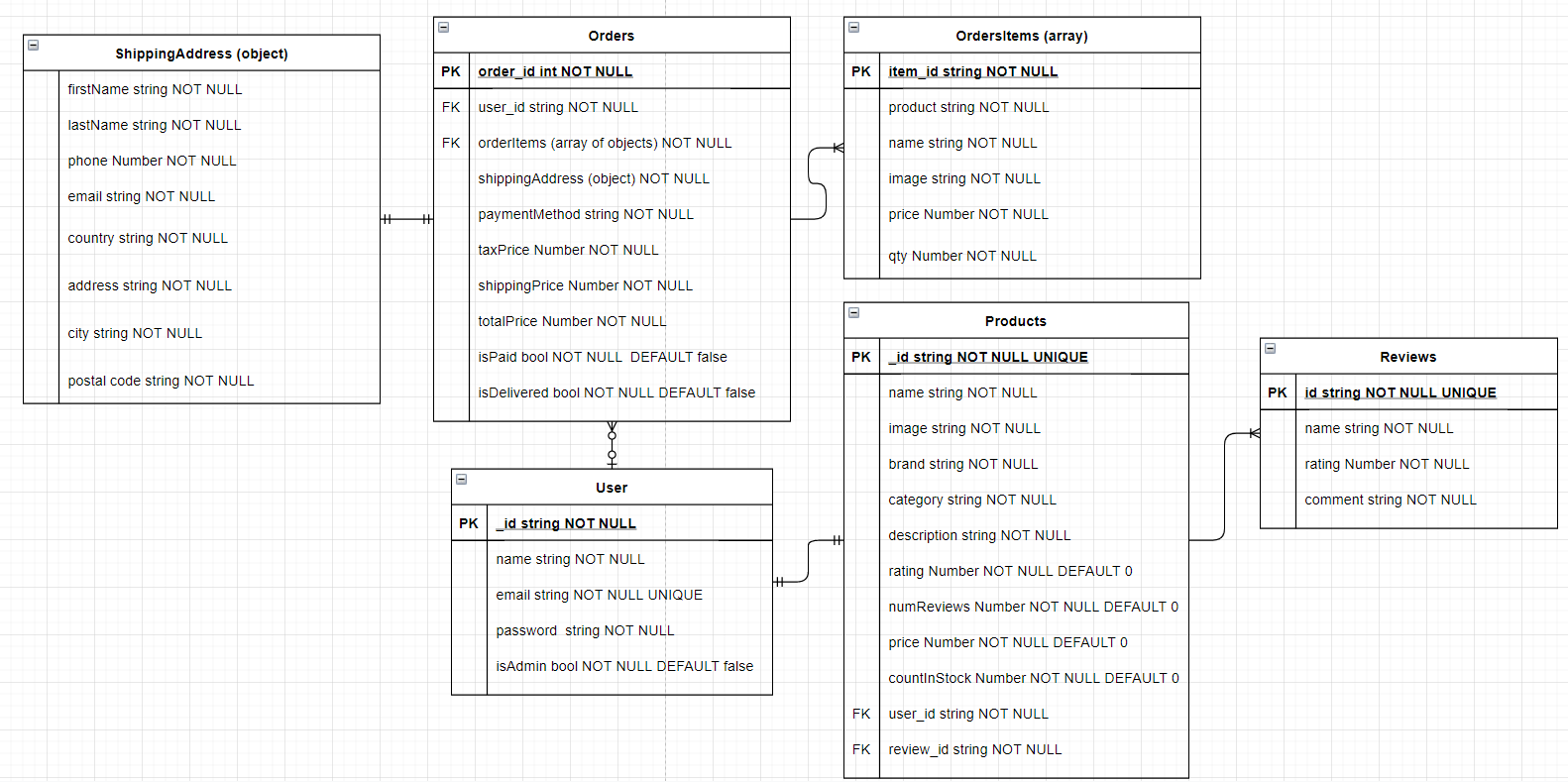
Diagram

Description automatically generated

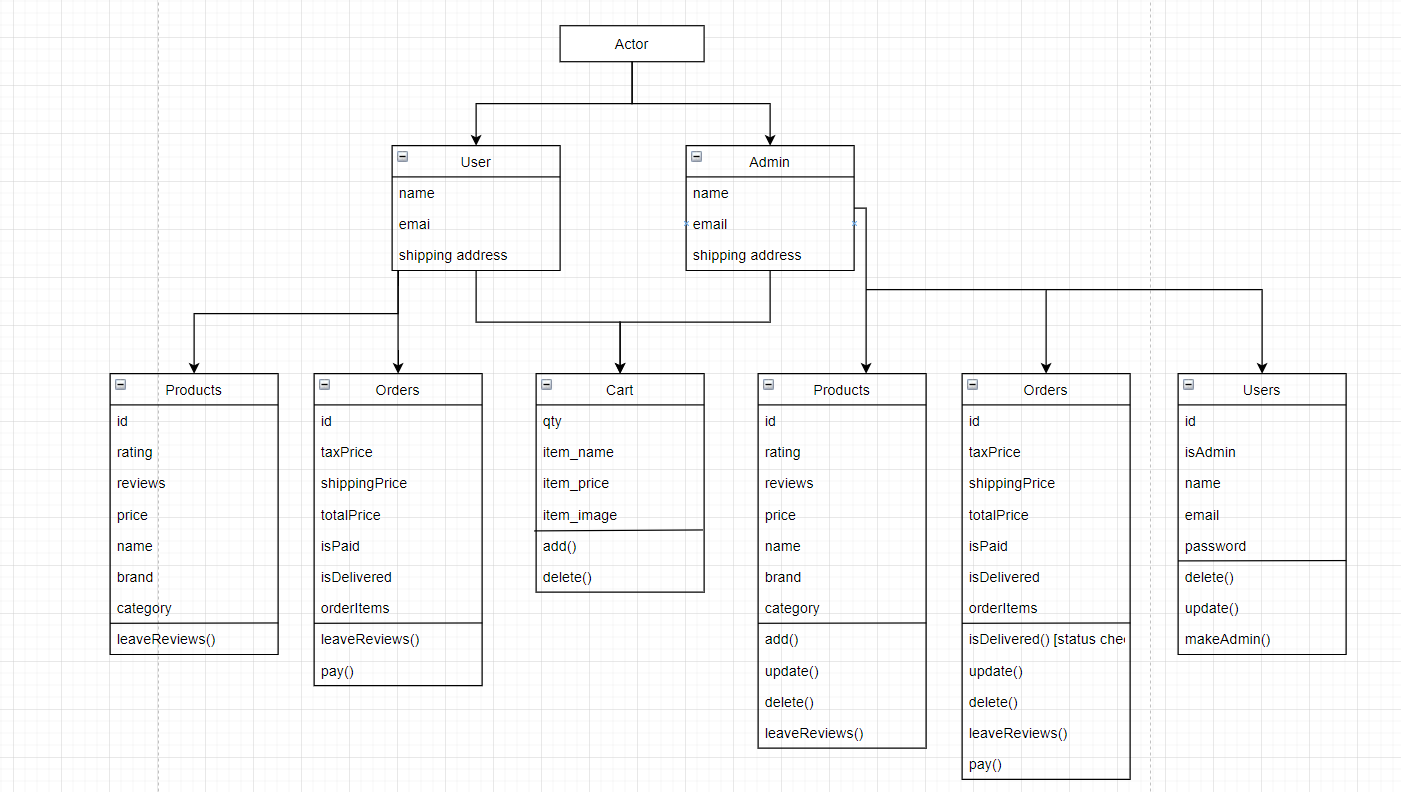


## Entity-relationship diagram

This is a mongoDB NoSQL database.



## Structure diagram



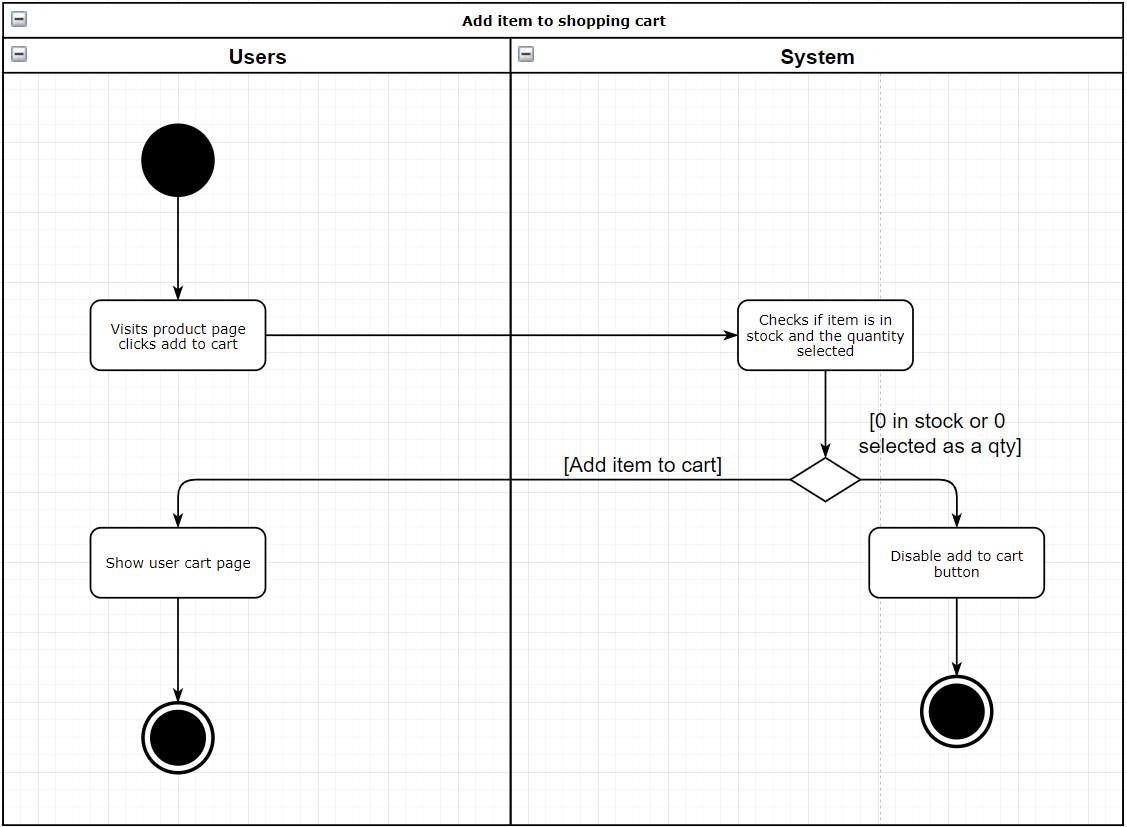
## System behavior diagrams

**activity diagram for each use-case and state diagram for the whole system**

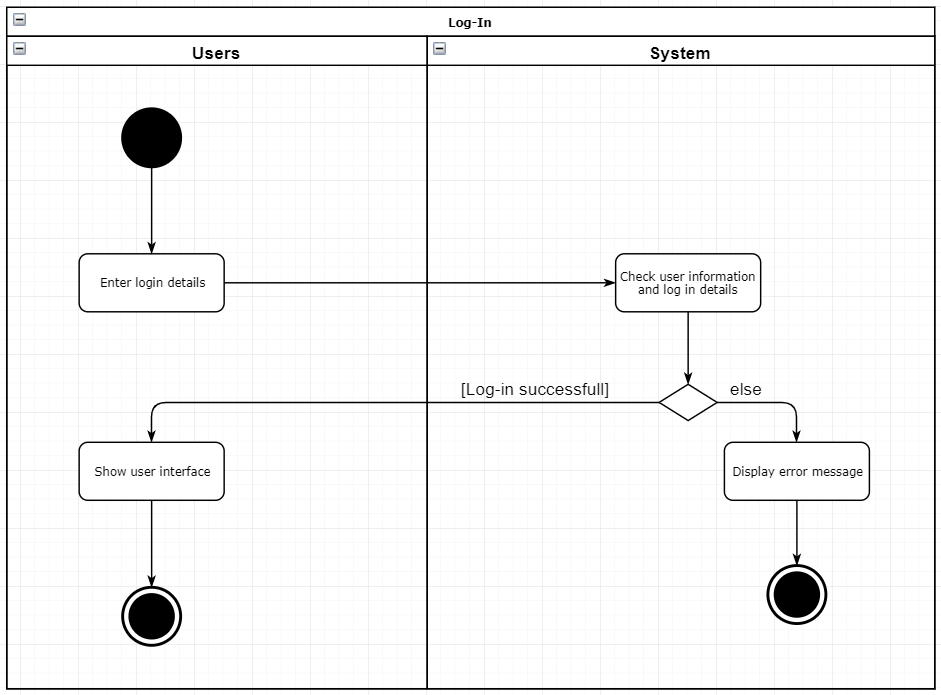
Diagram

Description automatically generated

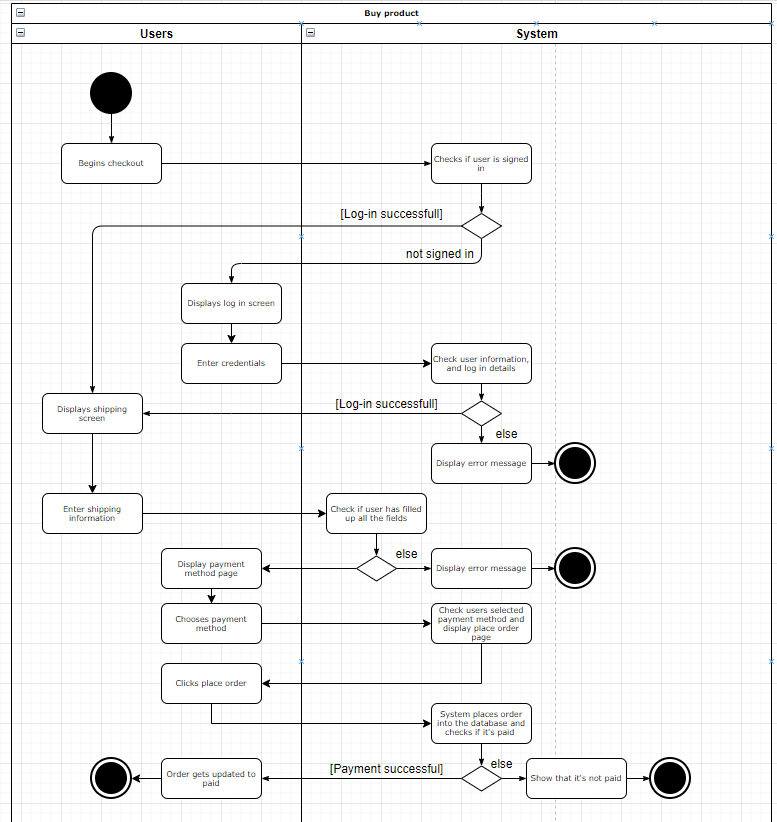
11UC-2 Add item to the shopping cart



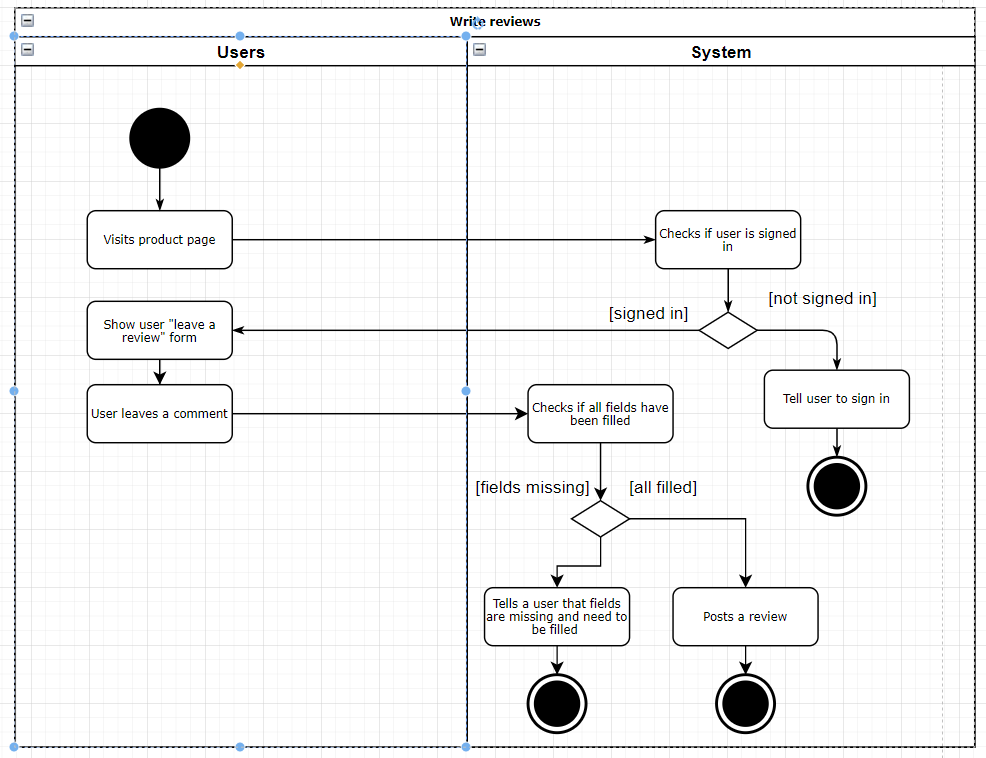
12. UC-3 Login



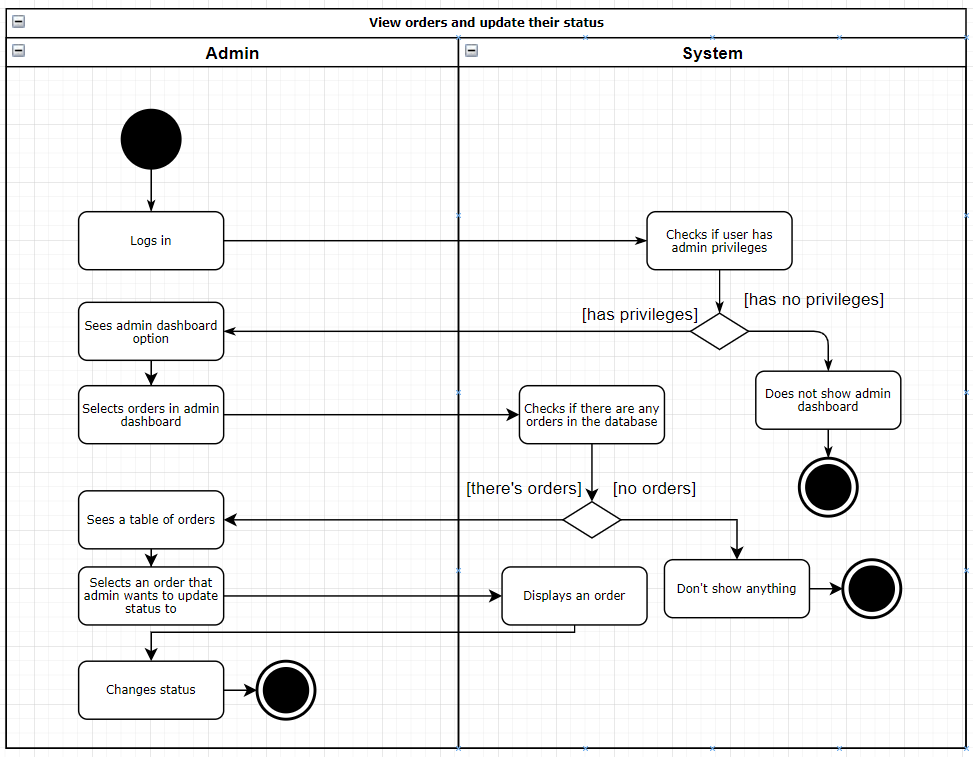
13. UC4 - Buy product



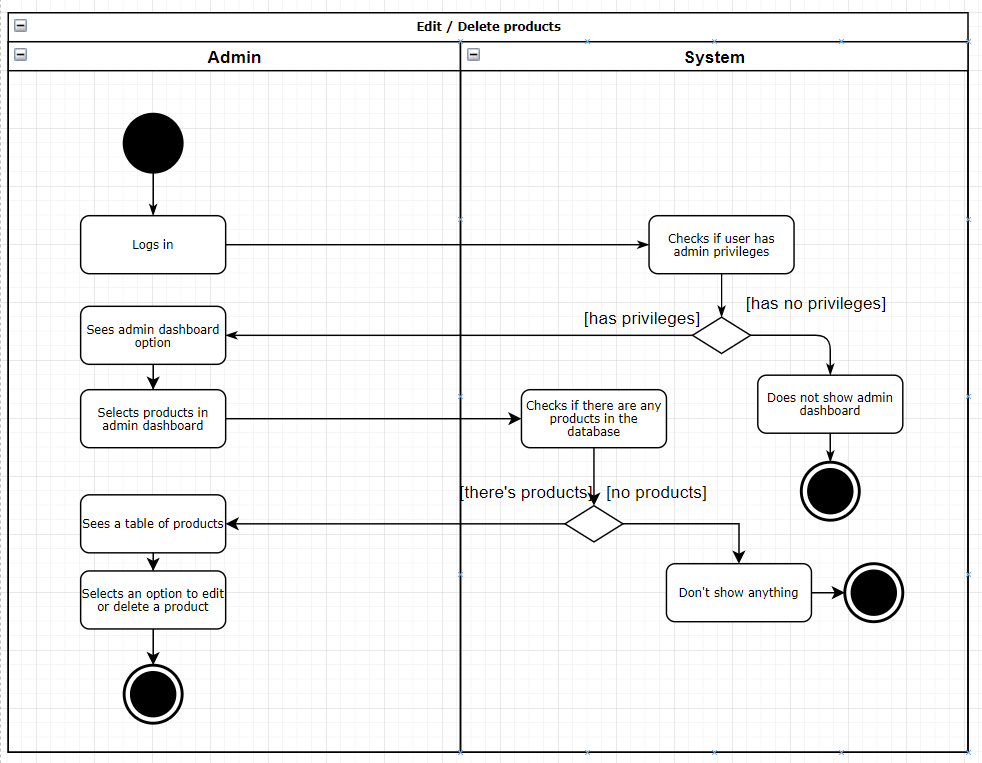
14. UC6 - Write reviews



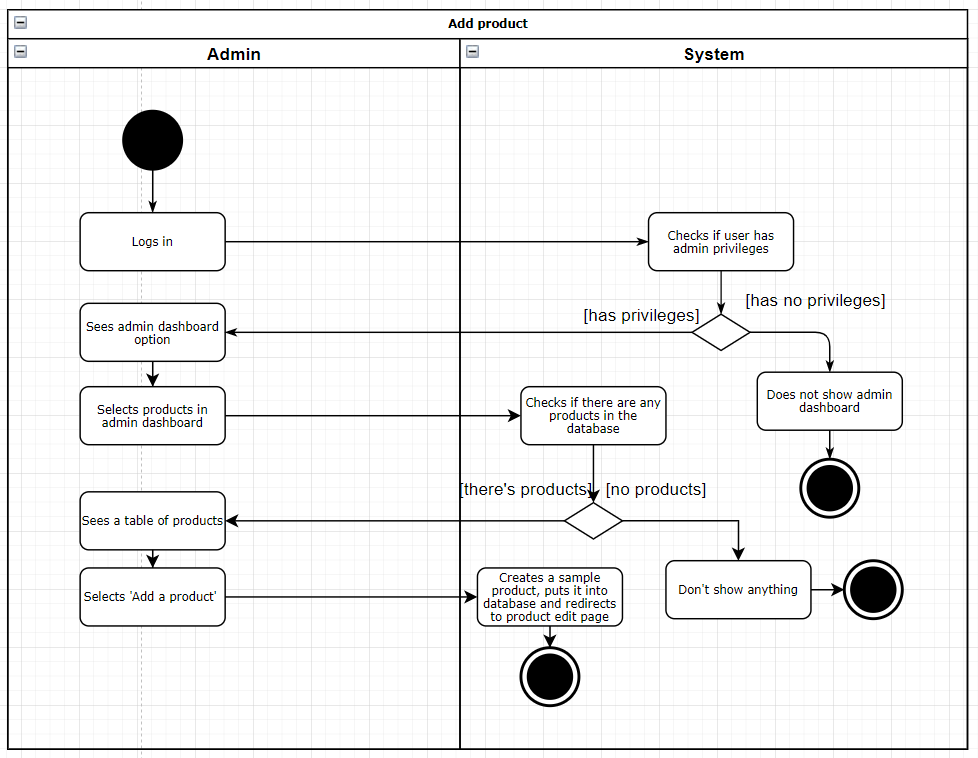
15. UC7& UC8 - view orders & update their status



16. UC9 - Edit / Delete products

**

17. UC10 - Add a product



# SOFTWARE IMPLEMENTATION TOOLS

Tools that have been used to implement the software:

**React.js + redux** – client side / frontend

**Node.js / Express.js** - server-side / backend.

**MongoDB** – database.

**Jest + Enzyme** – testing.

**VS Code** code editor for writing code and tests.

**Git + GitHub** – for version control

**Trello + teamgantt –** for tracking software development progress.

# USER DOCUMENTATION

User documentation can be found [here](https://dziugaspeciulevicius.github.io/eds-store-docs/).

Repository for user documentation can be found [here](https://github.com/dziugaspeciulevicius/eds-store-docs).

# CONCLUSIONS

In this project we built fully working ecommerce project. In this project we learnt how to fulfill roles as a project manager, business analyst, programmer and tester. We finished everything on time, all requirements were fulfilled. This app will be monitored and improved since there’s still lots of improvements to be made from better styling to new, more complex functionality.