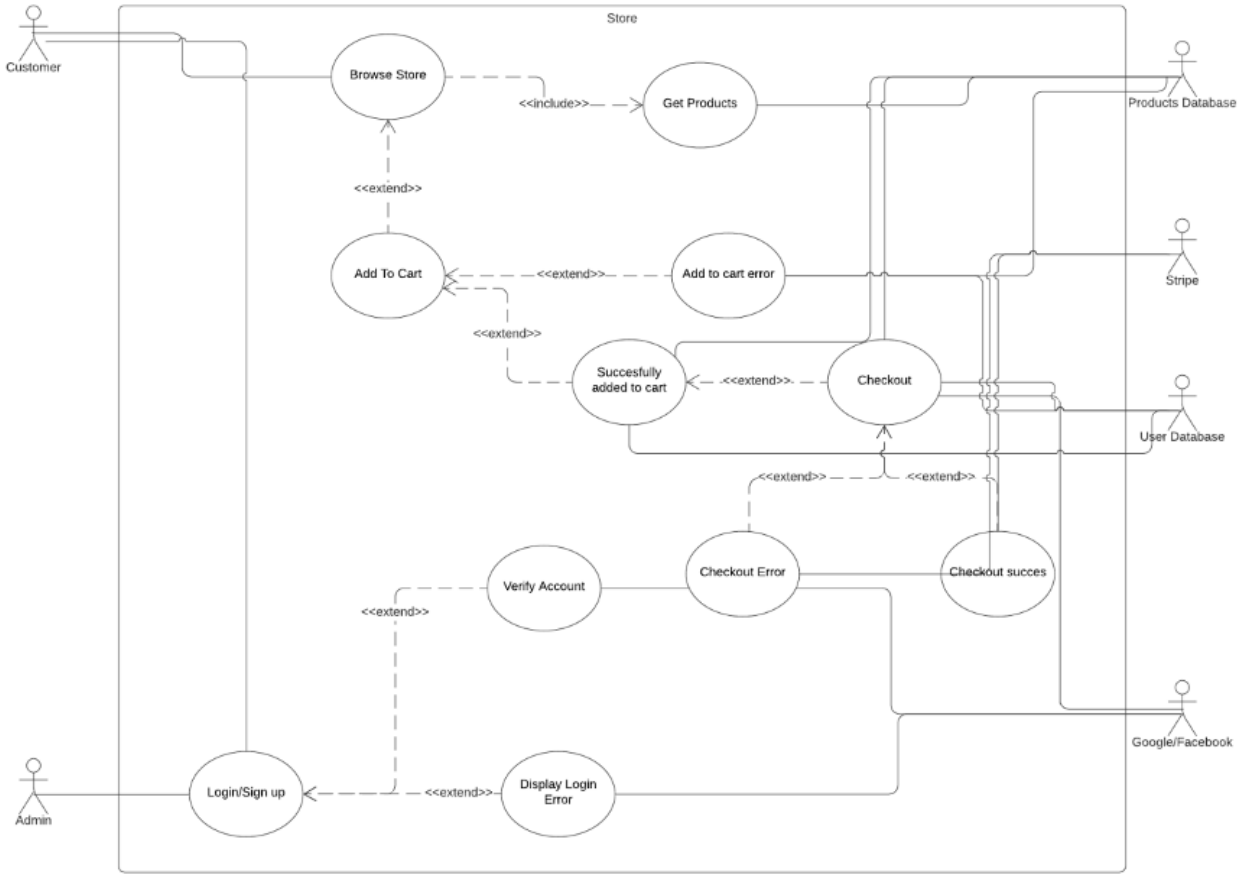
1.1 Business Case

Emarketplace is an online platform where users can buy and sell products. The products on display are populated by the users. All payment transactions and the storing of product information is handled by us. The principal use case for our application is to connect buyers and sellers on a beautiful and functional platform. Our aim is to build a platform where sellers can showcase their products in an elegant manner and where sellers can browse those products and purchase them right in our store.

1.2 System Requirements

1.2.1 Use Case Model



Use Case	Description
Browse Store UC-1	This is the bread and butter of our store. This is where the user can browse our selection of products available for purchase. The products and their associated information is stored on our database and subsequently fetched and displayed from our frontend.
Add to cart UC-2	The user can select a product to add to their cart along with a quantity. The cart will update the cost accordingly if the products are available in their requested quantity.
Add item UC-3	Users can add items to the store under their accounts.
Checkout UC-4	This is where the user can begin the checkout process. Final checks of the product's availability as well confirmation that the user is logged.
Login/Sign up UC-5	Here users and admins can login and sign up using our login system.
Add to cart error UC-6	If the items are no longer available the user will get an error.
Checkout error UC-7	If there are any problems during the checkout process - like payment errors or the item is no longer available - the user will get an error.

1.2.2 Quality Attribute Scenarios

ID	Quality Attribute	Scenario	Associated Use Case
QA-1	Security	When a user makes a payment, their payment information and transaction must be processed in a secure and lawful manner. All regulations must be followed during this process.	UC-6, UC-7, UC-8, UC-9, UC-10, UC-11
QA-2	Performance	The user must have a seamless experience when accessing the website. All interactions must be executed elegantly with accurate and performant functionality through a beautiful UI.	All
QA-3	Availability	The Databases and third-Party services must be always available 24/7.	All
QA-4	Modifiability	All components of the system must be modularized and separated as much as possible. For example, the payment processor that we chose - stripe - should be swappable with another service like Square with relative ease.	All

QA-5	Testability	There must be a system developed to constantly test features on the application to make sure that it is always working. The main areas of concern are that all the apis are always functioning as expected.	All
QA-6	Usability	The application should be intuitive and should have a small to no learning curve to the users.	All

1.2.3 Constraints

ID	Constraint
CON-1	The application must run as intended on Chrome, Firefox, Safari and Edge.
CON-2	All the users purchase history must be stored
CON-3	Before processing a payment, the system must be certain that the item/items being purchased are in stock
CON-4	A relational database must be used
CON-5	User should be tracked on the website to create a model of their interests for marketing purposes
CON-6	The user must be informed about this tracking and storing of information and must be presented with an option to opt out

1.2.4 Architectural Concerns

ID	Concern
CRN-1	Designing and implementing a distributed system that can easily scale with a growing user base.
CRN-2	Leveraging the team's knowledge in multiple domains of web development. Frontend, backend and devops.
CRN-3	Appropriately dividing up tasks within the development team.
CRN-4	Develop an overarching architecture for the application.