# Mini E-commerce

## Instructions

- Use react-router for routing.
- Use redux and thunk for dispatching asynchronous actions.
- Keep the code clean, commented and documented. Maintain feature based coding. (separate action and reducers - feature wise folders ) example : all the action and reducer related to authentication in one folder etc.
- You are free to use any css solutions as long as it looks good. Remember this also has score

## Problem Statement:

- Your task is to build a mini e-commerce app where a user can check all available items and able to purchase those.
- Your app should have following pages with navbar to access all these pages.
  - Product page `(/)`
  - Product details page `(/product/:id)`
  - Cart page `(/cart)`
  - Orders page `(/orders)`

## Product page

- Populate this page using the following API.

- In Products page, use this API end point to fetch all products and populate on DOM

```
https://dbioz2ek0e.execute-api.ap-south-1.amazonaws.com/mockapi/get-products
```

- You can find documentation here
  - Refer to `GET` **Get all products with their price**

```
https://documenter.getpostman.com/view/9952810/VUjQmjcF
```

- Display these products in form of cards with following details
  - brand
  - title
  - image
  - category
  - price
- When clicking on any product, move user to product details page, more details about this page will be shown below.
- You should have only 4 cards per row (use Grid)
- Your app should have filter and sort functionality
  - Filter by category
    - Filter according to kids, women, homedecor etc
    - When clicking on any category, products related to that category should be displayed.
  - Sort by price
    - High to low
    - Low to high
      - Achieve this using `orderBy` param from API itself (check documentation)
- Your app should also have pagination, each page should have 12 items/cards (you can use `limit` param from API)

- Note that filter and sort functionality should work together.
  - User should be able to sort according to filtered data.
- Note that filter and pagination functionality should work together.
  - Pages should change dynamically according to filtered data.
- This page should be responsive as well , apply Media Queries
  - Large screen : Default
  - Medium screen : 2 cards per row
  - Small screen : 1 card per row
- Make sure you make good UI, it will have extra weightage.

## Product details page:

- When clicking on any product, move user to product details page, where you will display more info about this product along with some dummy description.
- User should be able to add to cart from this page.
- On clicking on `Add to cart`, move that particular product to cart page.

## Cart Page

- Display all items which was added to cart.
- User should be able to increment/decrement quantity of items and price should change accordingly.
- Also display total price along with Delivery charges (50/-)
- Also create a button `Place order`, on clicking user should be place order with success message alert.

## Orders Page

- Display all orders which user has successfully placed.

## Note:

- Maintain flow of app as mentioned.

- Error messages should be shown, make use of any CSS library of your choice (chakra and MUI preferred)
- Use loaders.
- Good designs will fetch bonus marks.
- Submitting local host links for mock server will lead to disqualification.

## Submission

- Please submit deployed link and Github link of code.
- **Push your code into `masai-repo`, don't submit personal repo links (submitting personal repo links will lead to disqualification)**
- Please double check if deployed version works or not (run deployed version on your laptop and then submit it).
- Any issues in deployed link, will be considered null and void.
- Please verify your submissions are correct.
- Make sure you follow all instructions carefully.
- Submit before deadline.

### Rubrics / Criteria to be judged upon

- HTML, CSS, React, **Redux**
- Filtering, sorting, pagination.
- Code cleanliness.
- Component structure and **Good Git Hygiene.**

### Time Limit - `4 Hours`