SWAPIFY Project Planning

**1. Frontend - React.js**

**Design Layout**

* **Homepage**:
  + User registration and login section (for new and existing users).
  + Two main buttons: "Buy" and "Sell"
* **Buy Page**:
  + Display products as cards with images, names, categories, and short descriptions.
  + A filter option for categories (electronics, furniture, etc.).
  + Option to view more details for each product (click to see full details on another page).
* **Sell Page**:
  + Form for sellers to upload product details (fields like name, category, description, price, images, contact info,quantity,etc).
  + A limit of up to 10 images.
* **Wishlist Page**:
  + Users can wishlist items and view their wishlisted products.

**Key Functionalities**

1. **User Registration and Login**:
   * **Components**: Signup.js, Login.js
   * Use React forms for user signup/login (can use local state management or integrate with backend/auth services).
   * Fields: name, email, password, phone, etc.(email verification)
2. **Buying & Viewing Products**:
   * **Components**: ProductList.js, ProductDetail.js
   * On the **Buy Page**, display all products available. Each product card should have a “View More” button to display full details (product name, description, price, category, contact info, and more images).
   * Users can **wishlist** the product and directly contact the seller via phone or WhatsApp.
3. **Selling Products**:
   * **Components**: SellProduct.js
   * Form for sellers to input product details: product name, category (dropdown options), description, price, images (upload up to 10), quantity, and contact details (phone, WhatsApp number).
   * Once submitted, the product should be displayed on the Buy page with all the details.
   * Option for the seller to reduce the quantity and delete the product once sold.
4. **Product Wishlist**:
   * **Component**: Wishlist.js
   * Allow users to add products to a wishlist.
   * Display all wishlisted items in a separate wishlist page where users can remove items or keep track of them.
5. **Seller Contact & Order Confirmation**:
   * **Component**: ContactSeller.js
   * On the product detail page, display the seller’s phone number, email, and WhatsApp number.
   * Users can contact the seller directly to confirm the order outside the platform (through call or WhatsApp).
6. **Quantity Management for Sellers**:
   * Sellers can log in, and once an item is sold, they can reduce the quantity or delete the item from the product list.

**Detailed Breakdown**

**Components Structure**

* **App.js**: Main component handling routing to different pages (Buy, Sell, Product Detail, Wishlist, etc.).
* **Navbar.js**: A navigation bar that includes links to login/signup, buy, sell, and wishlist pages.
* **Signup.js**: Component for user signup.
* **Login.js**: Component for user login.
* **BuyPage.js**: Displays all the products available for purchase.
  + Each product will have a **ProductCard.js**.
  + Cards should display product name, price, a thumbnail image, and a "View More" button.
* **ProductDetail.js**: Displays full details of a specific product when a user clicks "View More."
  + Includes additional images, full description, contact details (phone, email, WhatsApp).
* **SellPage.js**: A form for sellers to upload their product details.
  + Form fields include name, category (dropdown), description, price, quantity, images (up to 10), phone number, and WhatsApp.
* **Wishlist.js**: Displays products that the user has wishlisted.
* **ContactSeller.js**: Displays the seller’s phone number, email, and WhatsApp on the product detail page.

**Backend (if applicable)**

Although you mentioned focusing on the frontend with React.js, a simple backend (Node.js/Express or Django) would be helpful to handle:

* **User authentication**: User registration, login.
* **Product management**: Store product data, manage uploaded images.
* **Wishlist feature**: Store users' wishlisted products.
* **Seller functionality**: Allow sellers to manage their products (update, delete, adjust quantity).

Alternatively, you could use Firebase as the backend to handle user authentication, data storage, and image uploads.

**Key Features to Implement**

* **Responsive Design**: Ensure the site works well on mobile, tablet, and desktop.
* **React State Management**: Use React hooks like useState and useEffect for state management, and consider using Redux if needed.
* **Routing**: Implement navigation with react-router for smooth transitions between pages (e.g., Buy, Sell, Product Details).
* **Form Validation**: Add validation for the sell form to ensure all mandatory fields (phone number, WhatsApp, etc.) are filled out.

**Tools & Libraries**

* **React Router**: For navigation between pages (Buy, Sell, Product Detail).
* **React Hook Form**: For handling forms and validation.
* **Axios**: For making API calls if you're integrating a backend.
* **Material UI/Bootstrap**: For styling the frontend and creating a clean, modern design.
* **Cloudinary/Firebase**: For handling image uploads (up to 10 images).

**Next Steps**

1. **Design UI**: Sketch out a simple UI wireframe to see how the pages will look.
2. **Create Components**: Start by building the reusable components (like the product cards, form inputs).
3. **Implement Routing**: Set up routes for navigating between the Buy, Sell, and Product Detail pages.
4. **Backend (optional)**: Decide if you want to use a backend (Node.js/Firebase) to handle product data and user authentication.