# Requirement Document

## Project Overview

- Objective: Build a modern real estate second hand marketplace with JWT authentication an and Redux Toolkit.
- Target Audience: Users interested in buying or selling used products.

## Functional Requirements

- Authentication:
  - Implement JWT authentication for secure user login and registration.
  - Include user profile management features (update, delete, sign-out).
- Listings:
  - Users can create new product listings with details such as images, description, price, etc.
  - Implement CRUD functionality for listings (Create, Read, Update, Delete).
  - Admin approves the listing of the product and its information.
- User Interaction:
  - Seller and Buyer can contact each other using their contact information.
  - Implement a messaging or notification system for communication between users.
- Frontend:
  - Use React.js for the frontend development.
  - Implement a responsive and user-friendly design with Tailwind CSS.

## Non-Functional Requirements

- Performance: Ensure the application is responsive and can handle a reasonable number of concurrent users.
- Security: Implement secure authentication and authorization to protect user data.

## Technology Stack

### -BackEnd

- Node.js as the runtime environment for the backend
- Express.js as the backend framework
- MongoDB for the database
- Mongoose as the ODM (Object Data Modeling) library for MongoDB

### -FrontEnd

- React.js for the frontend
- Tailwind CSS for styling
- Redux Toolkit for state management

### -Authentication

- JWT for authentication and authorization

## Deployment

- Deploy the application to a hosting service (e.g., Vercel).

# Design Document

## System Architecture

- Client-Server architecture with React.js as the frontend and Express.js/Node.js as the backend.
- Use Redux Toolkit for state management.

## Database Schema

- User table: id, username, email, password, etc.
- Listing table: id, title, description, images, price, contact details, etc.
- Bidding table: id, amount, message, contact details, etc.

## Authentication Flow

- User registration and login with JWT tokens.
- Secure routes and API endpoints requiring valid JWT tokens.

## User Interface Design

- Design responsive and intuitive UI using React.js and Tailwind CSS.
- Include pages for listings, user profile, search, filter, etc.

## API Endpoints

- Define API endpoints for user authentication, CRUD operations on listings, bidding, etc.

## Error Handling

- Implement a consistent error-handling mechanism for API requests and form submissions.

## Deployment Strategy

## Testing

- Define testing strategies for both frontend and backend components.

## Scalability

- Consider potential future scalability requirements and design the system to handle increased user loads.