PEERKART

CATEGORY

Web application, Android app

PURPOSE

The purpose of developing this application is to provide a convenient platform for the users to get supplies (groceries, medicines, and much more) home delivered by people who either live in their vicinity or are already present at the store, in exchange for some incentives.

SCOPE

This application will provide the users with an efficient delivery system that enables users to connect with people who are already out there shopping or are planning to go to the store. This will help those who cannot go out and shop themselves while also providing the people delivering the orders with some incentives. It is most likely that the person accepting the order lives near the one who has generated the order thus promoting intercommunity interactions.
Thus our app will reduce the carbon footprint as only one person will be commuting in place of two, helping the environment.

INTRODUCTION

Existing Systems

Currently, users can shop themselves or use other delivery services to get their supplies. The traditional way of shopping yourself, although it is very reliable, is not really convenient.
While there exist delivery services like Dunzo, they are not peer-to-peer like our app.
Some of the major drawbacks of the current systems:

- **High delivery charges:** Currently the users need to pay high delivery charges for the items they wish to get.
- **Greater delivery time**: In the present scenario most of the delivery systems are based around one or two days delivery options.
- **Waste generation:** The waste generated in the packaging of products is significantly large.

Proposed System

Our app will allow users to freelance and earn credits while costing them nothing extra. The credits can then be used while ordering something for themself or can be withdrawn in exchange for some coupons. Our model would be very similar to Uber where people delivering the orders would not be employed by us but would rather be working as “freelancers”, accepting orders on their free will. This model has been a huge success for Uber and thus can be replicated as an instant delivery system.

The application after careful analysis has been identified to be presented with the following features:

- **User registration:** Users can register and then log themselves in to use the application.
- **User authentication:** A robust user authentication system (Login system) will be present.
- **Create, modify and delete shopping order:** Users can put up a list of supplies they need and can even edit it later on if needed.
- **List filtering:** People can filter the orders according to some specifications.
- **Matching of orders within proximity:** Users will be shown orders based on the proximity, i.e. orders which have been generated at a closer proximity will be given higher priority in the list.
- **Payment integration:** Razorpay payments integration.
- **Order tracking:** Real-time order tracking available to the person placing the order.
- **Displaying route to carrier:** The carrier will be shown a route to the destination using google maps.
- **Order and points history:** A complete history of orders placed and accepted is shown. Along with this, a log of points history is available to the user as well.
- **Calculation of time saved:** People using this application will be shown how much time they have saved.

Advantages for person placing the order:

- **Time saved:** It saves time for the person placing the order as they no longer have to commute to the store rather they get the items home-delivered.
- **Convenience:** Home delivery increases the convenience factor for any user.
- **Social bonding:** It increases social bonding and builds a better community due to increased social interaction among people living in each other's vicinity.

Advantages for person delivering the order:

- **Incentives:** People completing the orders get credits which can later be converted to coupons or used while placing their orders.
- **Social bonding:** It increases social bonding and builds a better community due to increased social interaction among people living in each other's vicinity.

FUNCTIONAL REQUIREMENTS

- **Users must have a valid user id and password:** To use the application, the user needs to have a username and a strong password.
- **Users must provide a valid proof of identity:** A valid proof of identity is required to access all the features of the application.
- **A valid payment method must be provided for hassle free transactions:** All transactions are based around UPI payments so users must have and provide a valid UPI Id.
- **Only authenticated users can put up or accept orders:** Only the users who have been authenticated and have provided all the details necessary will be allowed to create orders.
- **Tracking of order by the user:** Users placing the orders can track the current status of their orders.
- **Showing the route of destination to the carrier:** The carrier must be shown a route to the destination address.
- **Access to user's location so they can see orders placed in the vicinity:** Users need to provide access to their location so that all the active orders present in their vicinity can be shown to them.
- **Availability of complete order history:** A complete history of all the orders placed or accepted by the users needs to be shown to it.
- **Log of users credit history:** A log of the user's credits earned or spent on each transaction will be provided.

NON-FUNCTIONAL REQUIREMENTS

- **Secure access of confidential data :** The database access needs to be secure.
- **Only authorised users can edit or modify the list:** Token-based authorization system in place so that only the authorised person can modify/delete the orders they have created.
- **Maximum time availability:** The application must be online for most of the time so that the users can access it at any time.

- **Flexible service based architecture for future extension:** The architecture of the application would be flexible so that any future extensions can be incorporated and there is the smooth integration of further features.
- **Providing real-time updates to the user:** The users would be provided with real-time updates regarding the orders they put up.
- **Notification updates:** Notification updates to users based on the change of the order status.

SOFTWARE TOOLS

Database Server:	MongoDB
Client:	Any web browser, Android phone
Development Tools:	VSCode, Postman, Chrome
Programming Language:	Javascript
Frameworks:	NodeJS, ExpressJS, ReactJS, React Native

DEPLOYMENT

Frontend:	Will be hosted on firebase.
Backend:	Will be hosted on heroku.

HARDWARE SPECIFICATION

Website:	Any device having access to a browser and an internet connection
App:	Any phone with Android 5.0 or newer.