TerraLink

**Project Description**: A kind of real estate management where the website consists of buyers and sellers(Sections. And each section has its own sign-in and login requirements) . The buyers can view the site once after signed-in, if they are interested in buying any property or rented places they can contact the seller via email which is provided there as a separate section with other essential information. As for the seller he/she has to sign-in, login, verify them by their aadhar number, and can start providing information about the property. Information are as follows 1) Location 2) Prize range(negotiable or not) 3)Area 4)Pictures of that property. 5)Important documents proving that you have the ownership of that property(This info will not be displayed in frontend ie: buyers section).

**Application Flow**

This is how the website normally functions.

1)Starts with a login page for either buyer or seller(by default its buyer, and a link to redirect it to seller). The user must have to sign in if he/she dosent have a account. (Flask Authentication)

**BUYERS PORTAL**

2)(After logged in buyer portal) Homepage appears, where the information about the website and decorations are present(kind of landing page or dashboard).

3)This homepage should have a navbar which has the following links(routes) **->homepage ->Explore ->MoveToCart ->Cart/Confirmation ->Email ->OrderHistory ->Logout.**

**->homepage:** same as the landing page.

**->Explore:** The page where the details and prices of all the lands or rented places are found. Here the user can select a particular property if he/she wishes to rent or buy according to the property details.

A get request is made to the backend.

**->MoveToCart:** After finding the desired property to buy/rent the user can select that property upon which there will be some extra information along with "Add to cart Button" which will move that product to cart page.

**->Cart/Confirmation:** After selecting a particular property(can be more than one also) he/she can come into this section for final confirmation, final bill after taxes and other convenience fee(if the price is mentioned below the property or the estimated range will be present). At last we can see a confirm button which redirects to a page where he/she can send e-mail to that respective seller for more clarification and negotation.

**->Email:** Where he/she can send mail to sellers about their interest in buying/renting the product.

**->OrderHistory:** Its contains the details about previous history orders.(Get request from the database).

**->Logout:** redirects it to login page.

**SELLERS PORTAL**

4)(After logged in seller portal) Homepage appears, where the information about the website and decorations are present(kind of landing page or dashboard).

5)This homepage should have a navbar which has the following links(routes) **->homepage ->upload ->uploadHistory(acts a upload\_view as well) ->upload\_Updations(Like removal or price change) ->logout.**

**->homepage:** same as the landing page.

**->upload**(the most important one): The seller is prompted with the upload forms where he/she has the following parameters to fill in. 1)Name of the property, 2)Owner's official name, 3)location, 4)Images of that location(at least 1 image required)(.png format), 4)Price/range of price, 5)can/cannot negotiate(the prices), 6)Description of that property(in words). If all the parameters are filled in the seller can press submit/upload button to store the information of that property in sqlite database which is then fetched in BUYERS PORTAL. A Post request is made here.

**->UploadHistory**( kind of upload view): Here the user can view all the previous listing(only from that particular user) made in this SELLERS PORTAL. A simple get request is sent to the backend to fetch all the past uploads from that user.

**->Upload\_Updations**(only applies to current property that is available in buyers portal). The seller can change and modify all the information about the property which will reflect in the buyers portal. The user can also delete the property. A post request is made to update the existing info in the database.

**->Logout:** redirects it to login page.