Mumbai Housing Detector

Spatial Informatics

Ruthvik Kodati (2019101035)

Finding Housing Regions in Mumbai -

Problem Definition:

When choosing a house to buy, there are many factors to consider such as its proximity to schools, hospitals, restaurants, malls, banks, marketplaces, etc. On top of that, it is essential to know the house price differences between different localities. Homebuyers typically expend a lot of time on the house selection process and I believe that if they incorporate filters such as searching for localities containing the amenities they need as well as selecting a suitable budget, it would hasten the process.

Objective:

My project aims to provide the user with an interactive interface that allows them to enter their monthly budget and proximity preferences. After the selection process, it delineates the housing regions in Mumbai based on the preferences selected by the user. The user can see the highlighted regions that fit their personal selections and can finalise the region they want to live based on their opinion.

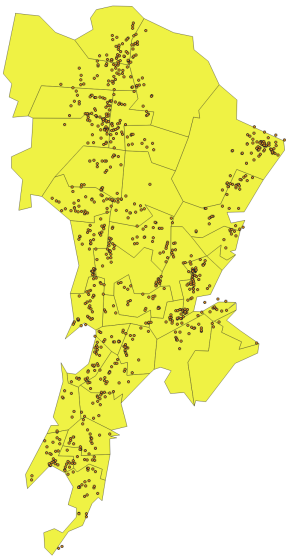
Assumptions:

- The user is only interested in finding a house in Mumbai (the project is constrained to the suburbs of Mumbai).
- The region is generated by creating a buffer with a radius entered by the user (it is not road distance).
- The rent data is considered as average monthly rent for a particular suburb.

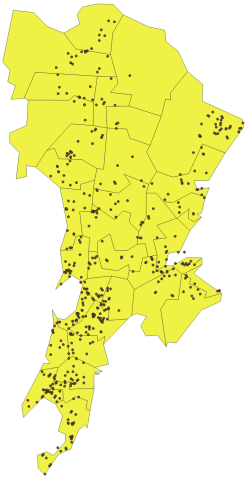
- Malls and department stores are considered the same as the data for malls was quite low (both layers were merged).
- Marketplaces and supermarkets are considered the same as the data for marketplaces was low (both layers were merged).

Data Collected :

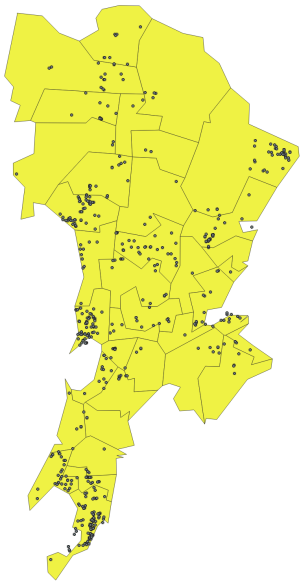
1) Used the QuickOSM plugin in QGIS to collect the point data for the amenities. The plugin takes a “key” such as amenity and a “value” for that key such as restaurant (amenity = restaurant).



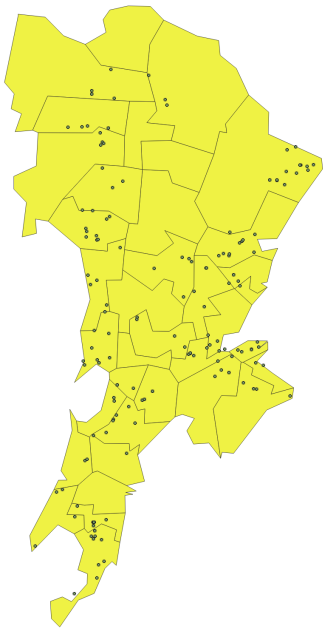
Total Hospitals



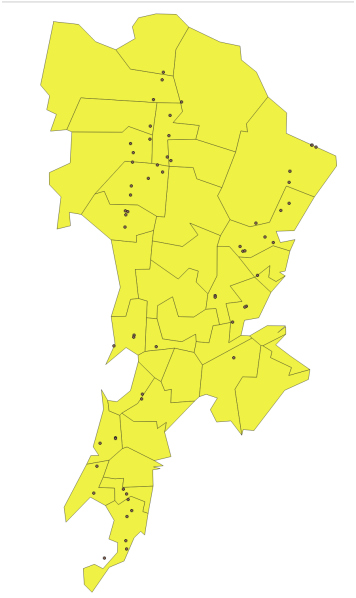
Total Schools



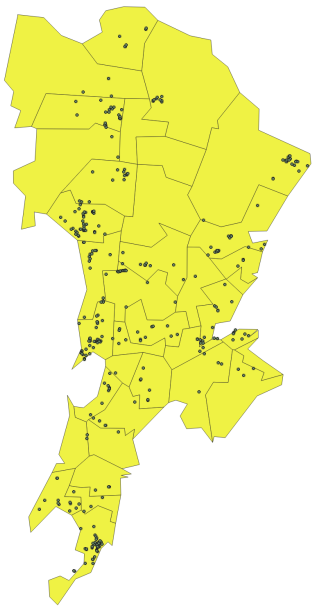
Total Restaurants



Total Marketplaces

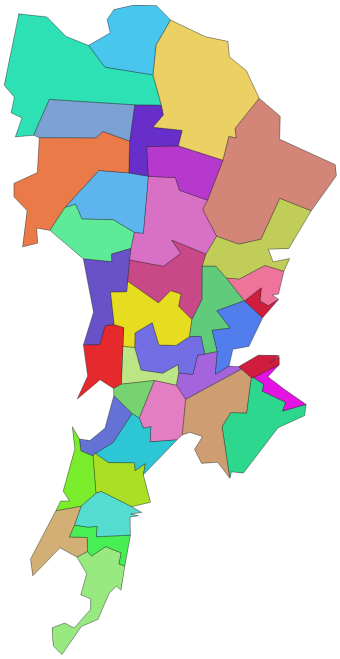


Total Malls



Total Banks

2) The following link has data for the Mumbai suburbs:
https://github.com/datameet/maps/tree/master/assembly-constituencies



3) The following link has the data for monthly rent in the various Mumbai suburbs:

https://www.makaan.com/price-trends/property-rates-for-rent-in-mumbai

4) Google Streets data : http://{s}.google.com/vt?lyrs=m&x={x}&y={y}&z={z}

GIS Methods Used:

1) **Centroids** - This QGIS method is found in `Vector > Geometry Tools` and converts the polygon data into point data.

2) **Merge** - This method is found in `MMQGIS > Combine` and merges multiple layers into a single layer.

3) **Dissolve** - This tool is found in `Vector > Geoprocessing Tools` and unifies adjacent boundaries based on common attribute values. Only if neighbour polygons have the same dissolving attribute, then it melts the boundary into one.

4) **Buffer** - It is found in `Vector > Geoprocessing Tools` and the algorithm computes a buffer area for all the features in an input layer, using a fixed or dynamic distance.

5) **Clip** - This tool is found in `Vector > Geoprocessing Tools` and it clips a vector layer using the features of an additional polygon layer. Only the parts of the features in the Input layer that fall within the polygons of the Overlay layer will be added to the resulting layer.

6) **Intersection** - It is found in `Vector > Geoprocessing Tools` and it extracts the overlapping portions of features in the Input and Overlay layers.

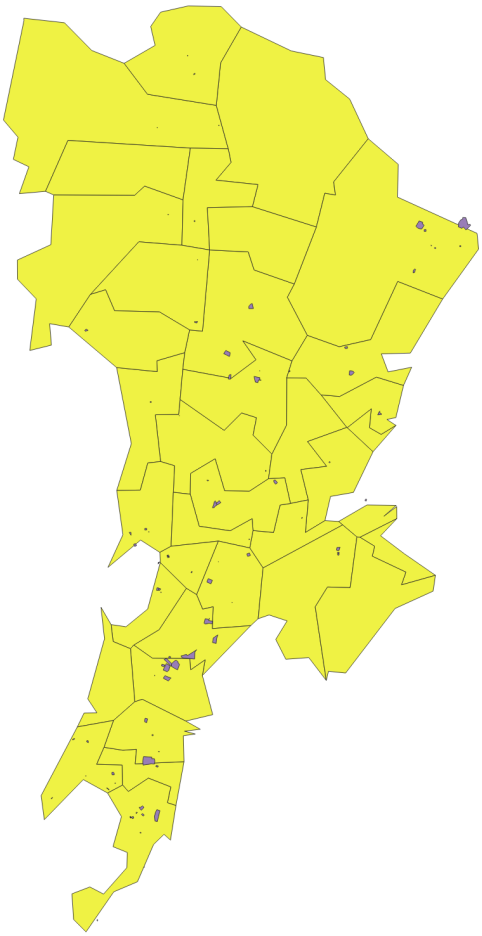
Tech Stack Used:

I used React for the frontend and Leaflet for plotting and interacting with the maps. The backend was written in Python using Flask to build the API's.

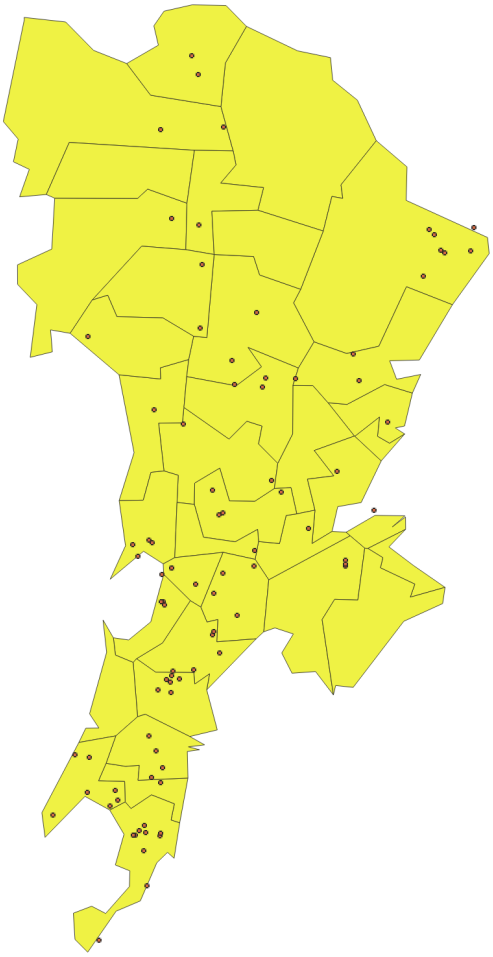
Procedure:

1) Collect the data for the various amenities. Some data may be polygon data and must be converted to point data using the Centroid function in QGIS.

2) Use Centroid to convert polygon data into point data. The example below shows some hospital data being converted from polygon to point:



Hospital Polygon Data



Hospital Point Data (After conversion)

3) Store the monthly rent data as a JSON object:

```
const rent_data = {
  "Vikhroli": [[5, 9], [35, 65], [70, 110]],
  "Vile Parle": [[25, 50], [50, 70], [80, 115]],
  ...
  "Byculla": [[27, 43], [55, 85], [90, 130]],
  "Malabar Hill": [[55, 85], [95, 145], [160, 300]],
}
```

The numbers are in thousands (5 indicates 5000 rupees).

4) The user enters their house type and monthly budget. After that, they select their proximity preferences for the amenities. Then, they make the query.

5) The backend creates a layer with the suburbs which satisfy the house type and monthly budget of the user. Let's call this Layer 1.

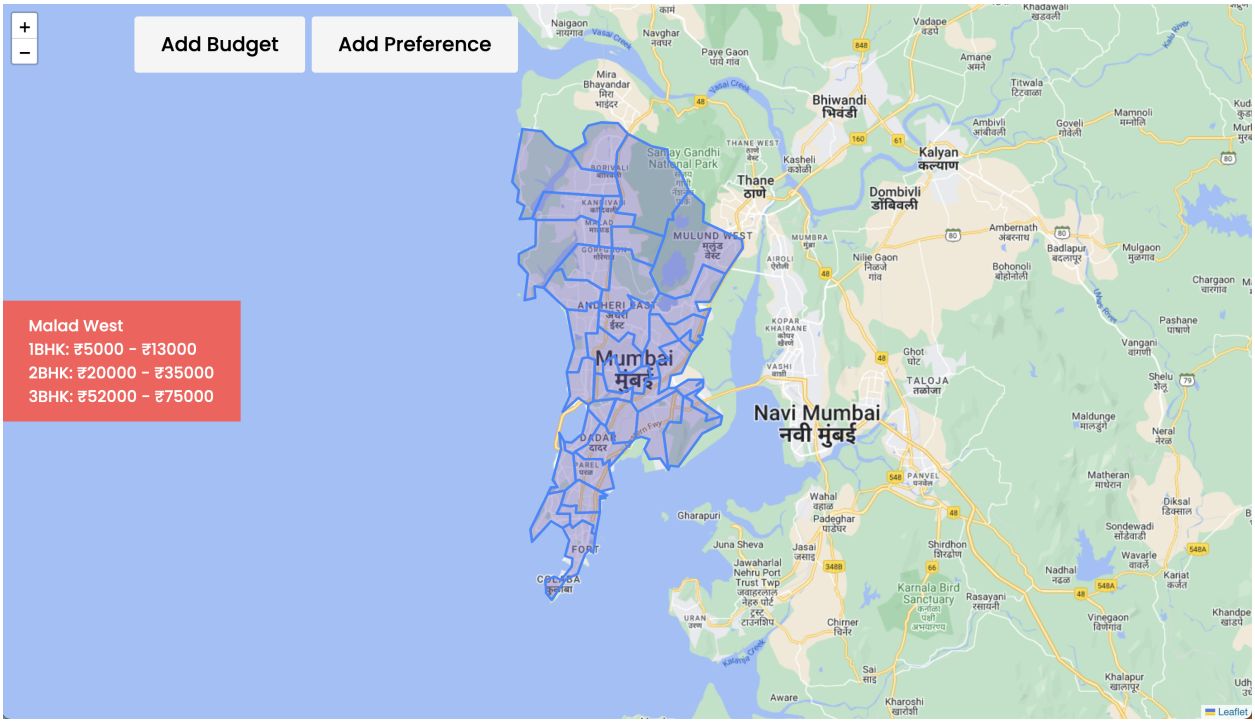
6) The individual buffers are made for each amenity using the distance given by the user and each layer is dissolved into a single feature layer.

7) A new layer is created and is the intersection of all the amenity buffer layers. This new layer is dissolved into a single feature layer. Let's call this Layer 2.

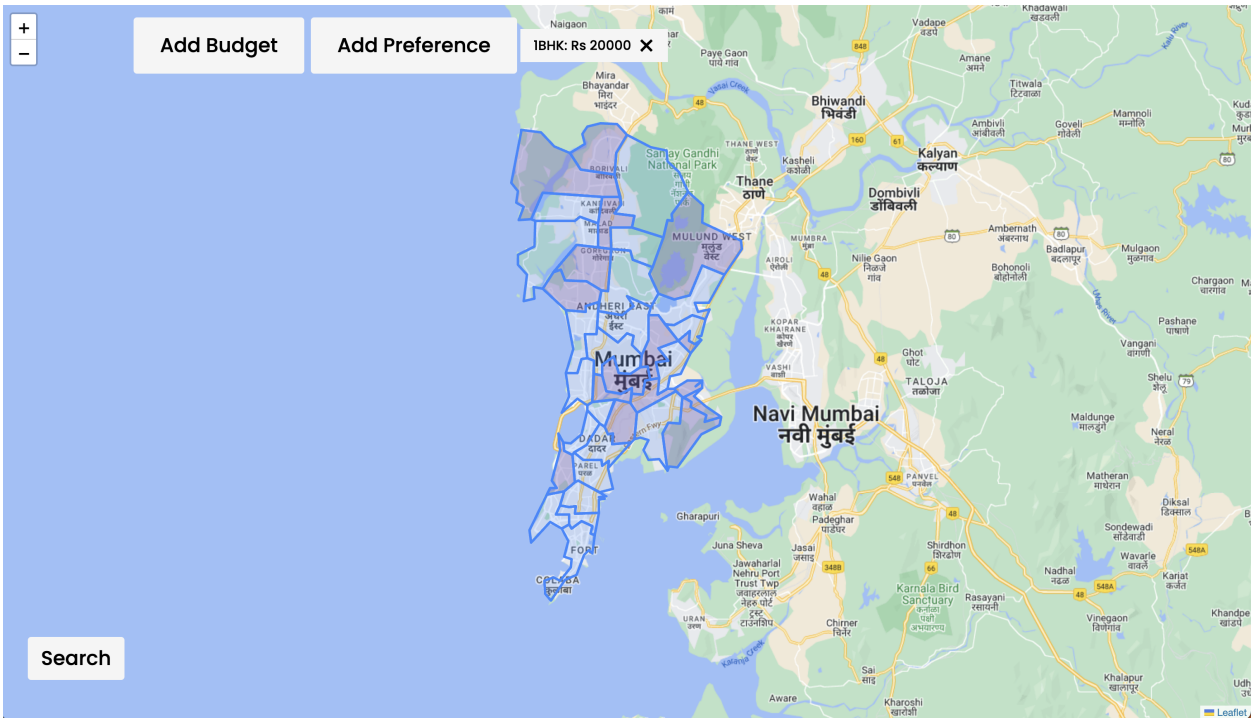
8) Then, Layer 2 (used as "OVERLAY" in clip) is clipped onto Layer 1 (used as "INPUT" in clip) to get the targeted regions for the user. The final layer which contains the preferred housing regions is sent back to the frontend as a GeoJSON layer.

9) The frontend displays the final layer received from the backend and the user can hover over the suburbs (hovering mentions the suburb name and its monthly rent data) which contain the regions that satisfy their preferences.

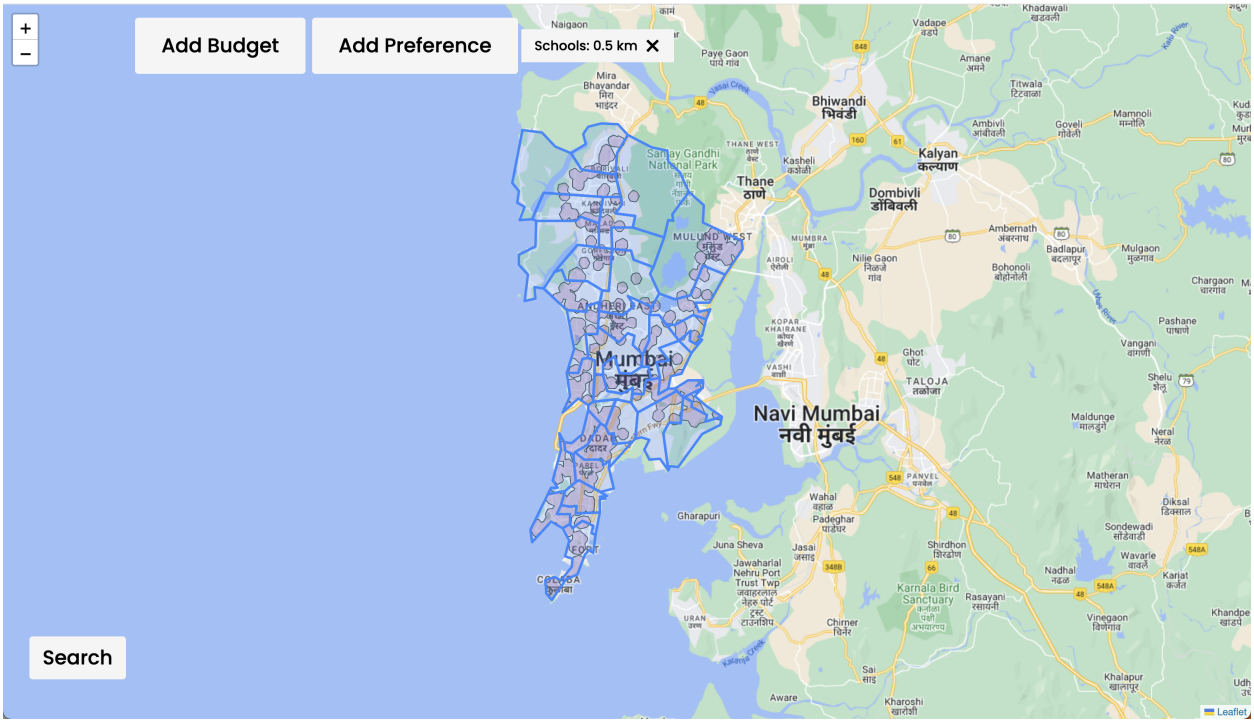
Web Interface Outputs:



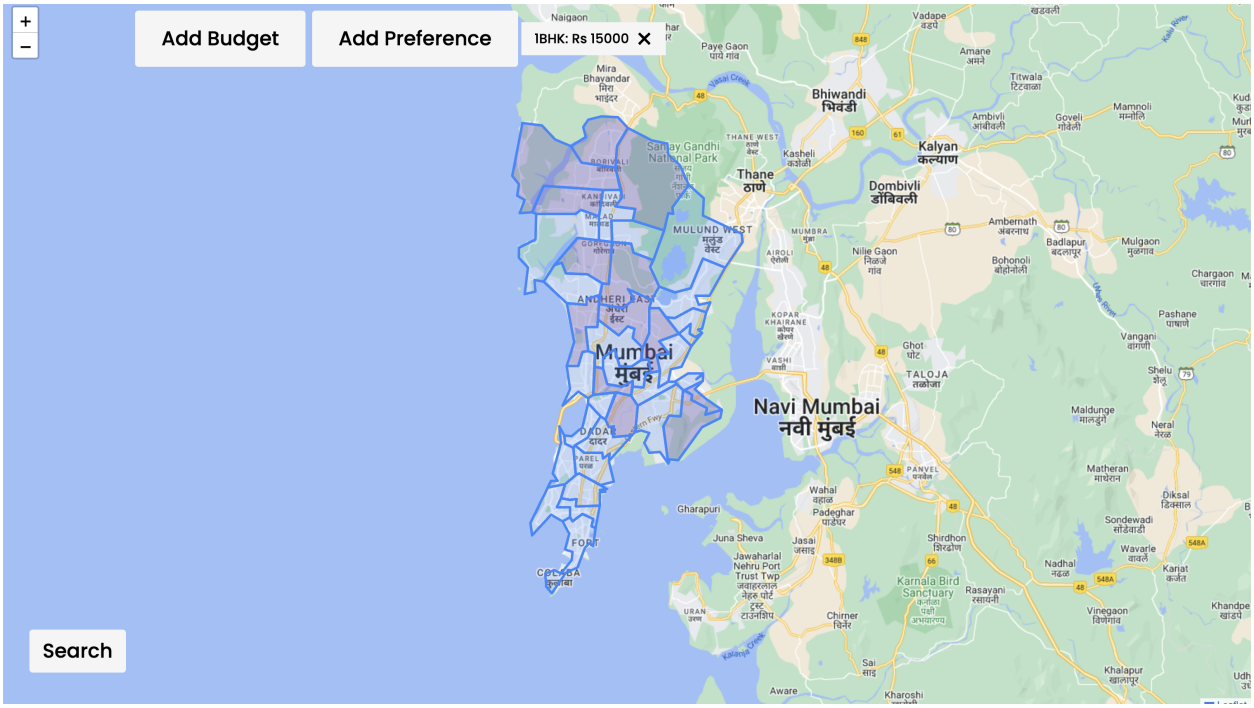
The React application displays the suburb by placing the cursor over it (the cursor was over Malad West in this example).



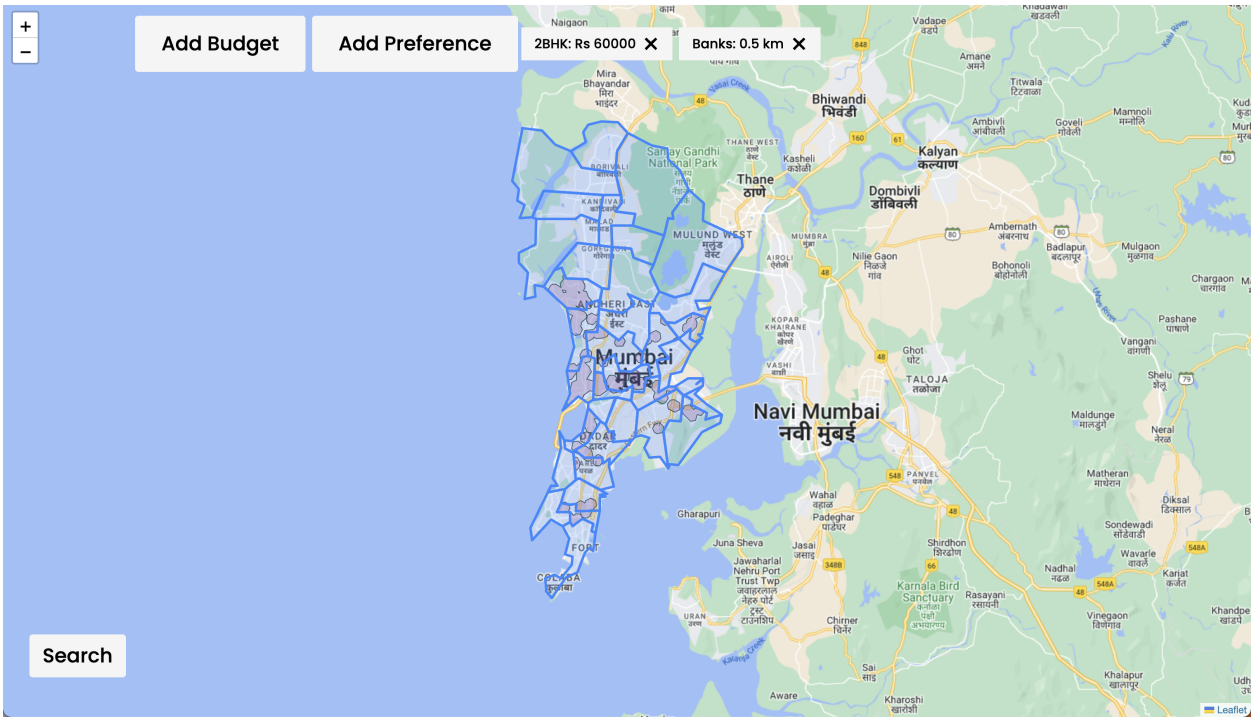
The highlighted areas show the regions that have 1bhk houses priced at 20000 rupees.



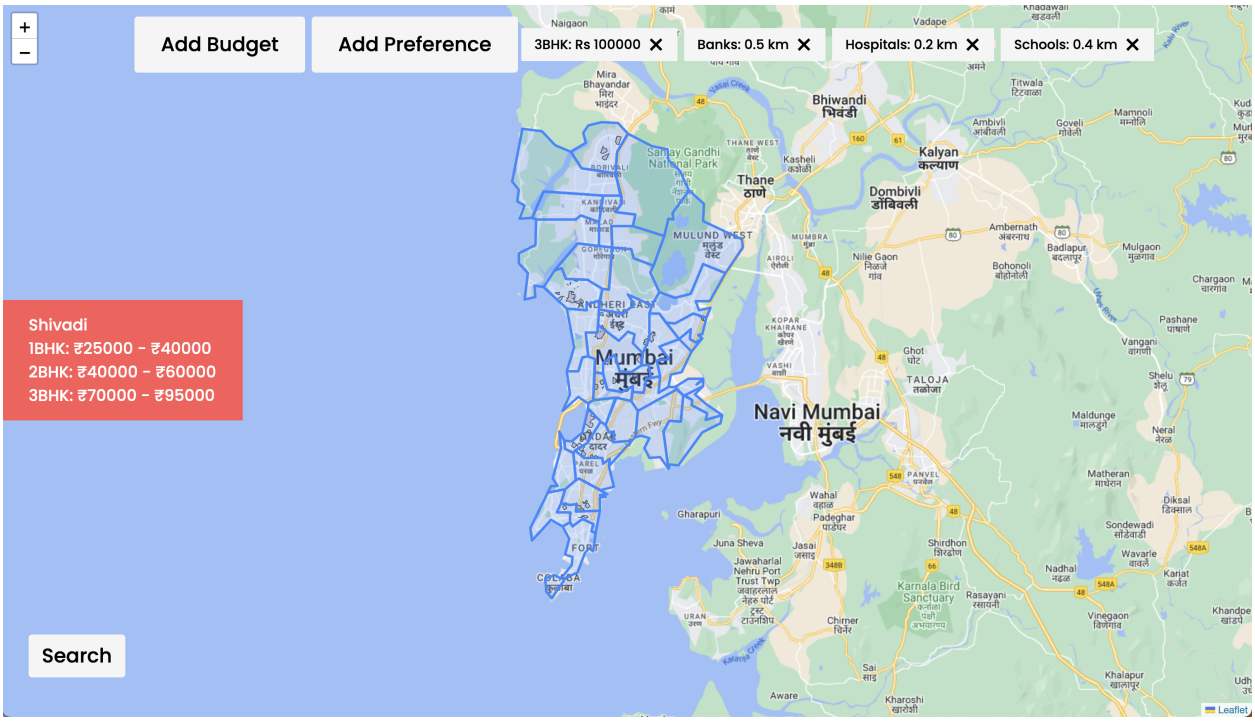
The highlighted regions delineate 0.5 km buffers for schools within Mumbai.



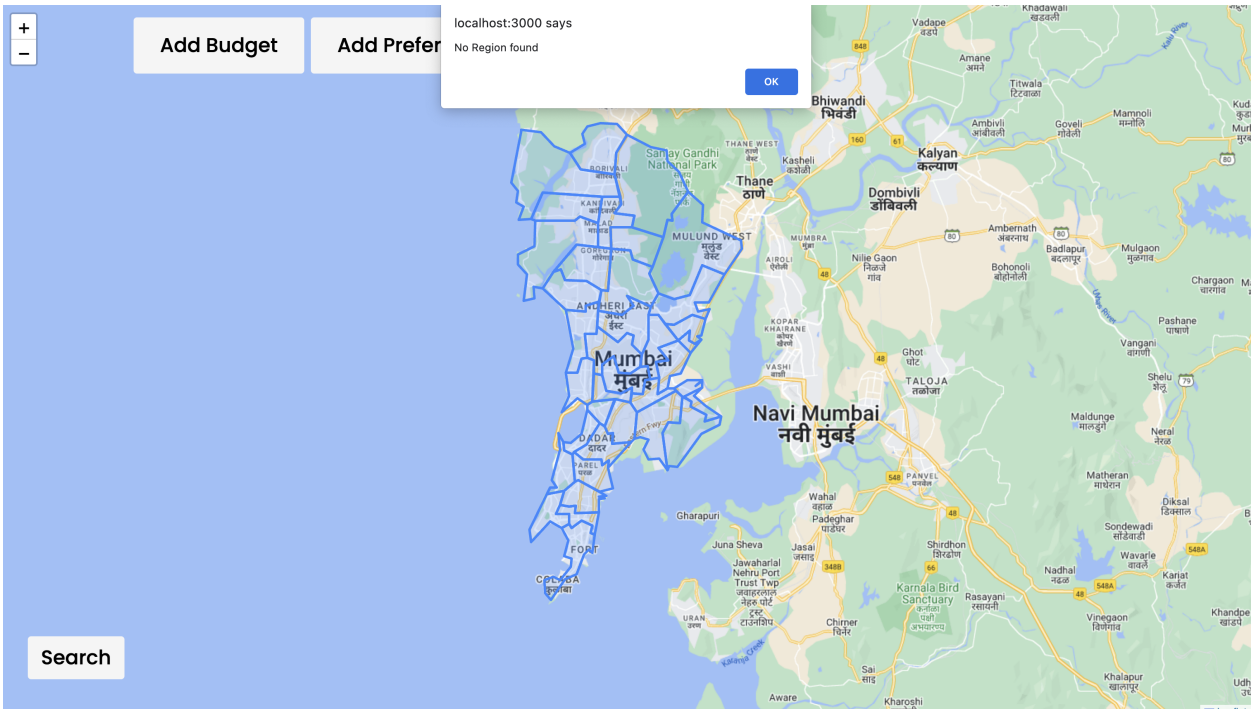
The highlighted regions show the suburbs that have 1bhk properties priced at 15000 rupees per month.



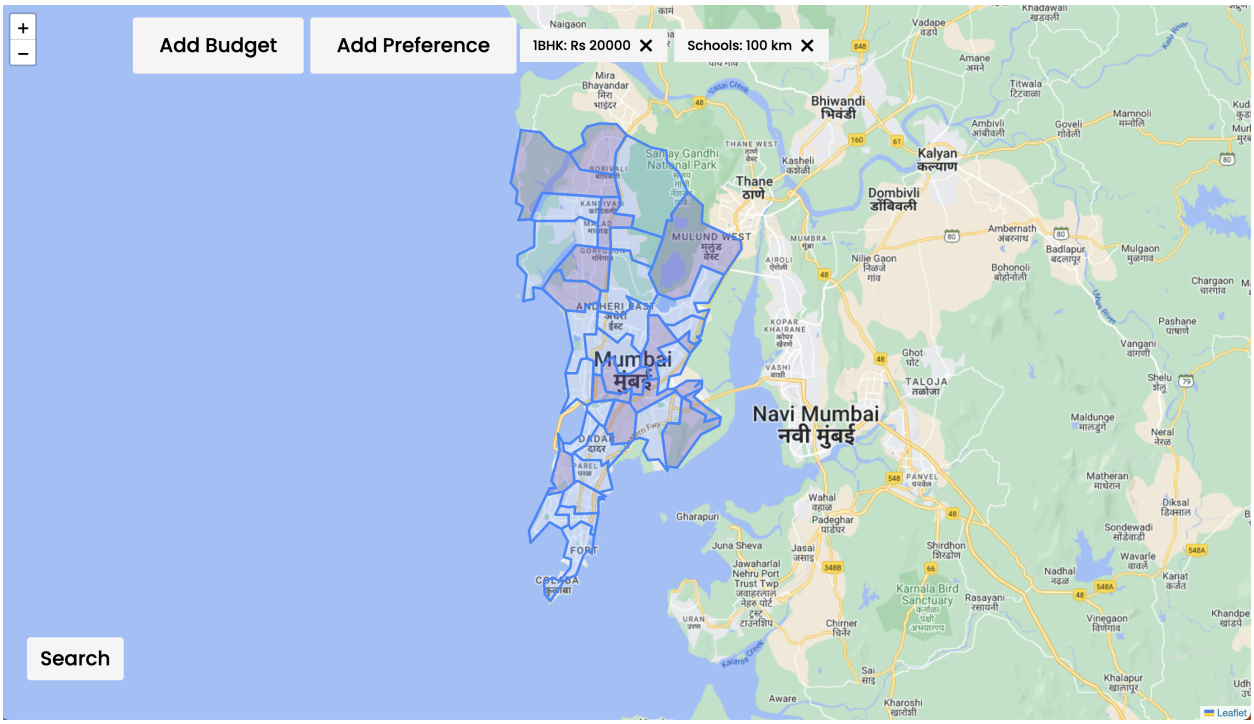
The marked areas show the regions that have 2bhk properties priced at 60000 rupees per month and are within 0.5 km (radial distance) to the nearest bank.



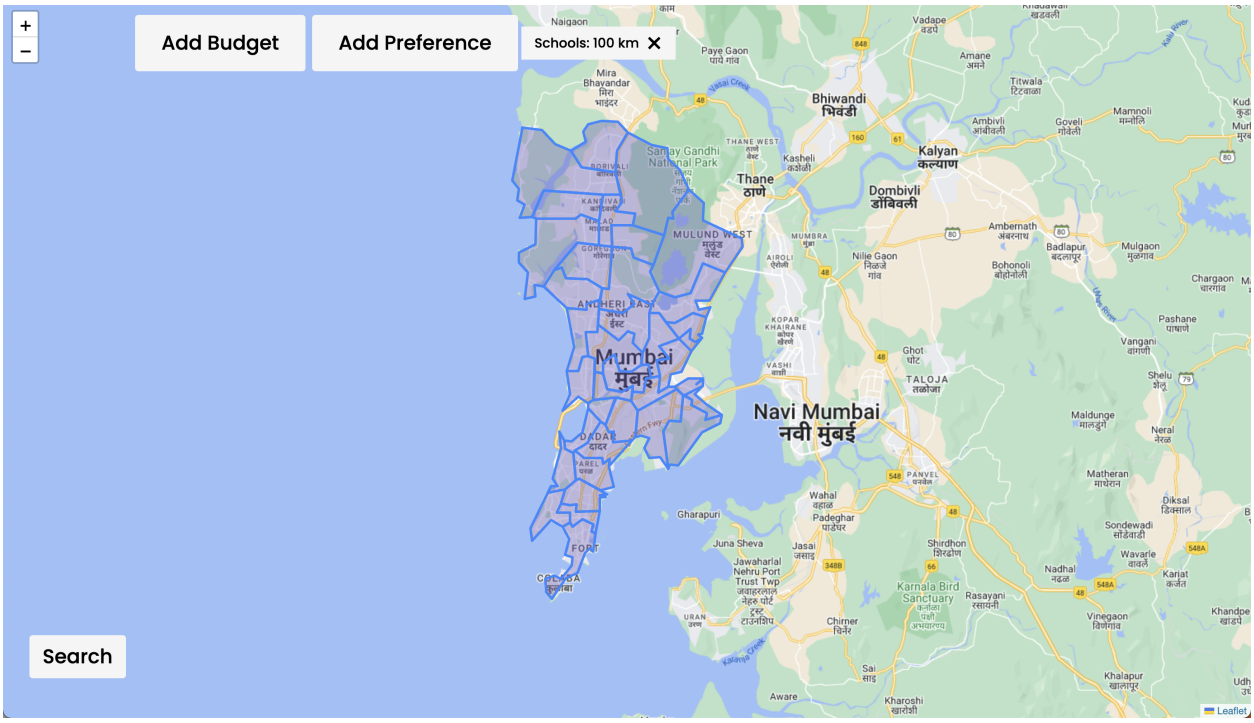
The marked areas show regions that have 3bhk properites priced at 100000 rupees per month and are within 0.5 km to the nearest bank, 0.2 km to the nearest hospital, and 0.4 km to the nearest school.



The application displays *“No Region found”* if the maximum possible monthly rent for a particular bhk is exceeded (in this example, 1bhk and 100000 monthly rent was selected but the maximum monthly rent for a 1bhk is far lower).



The highlighted regions are the areas which have 1 bhk house prices at 20000 rupees per month and are within a 100 km radial distance to the nearest school.



The marked areas show the regions with nearest school within a 100 km radial distance (it covers the entirety of Mumbai because the region considered for the project is roughly $440 km^2$).

Learnings:

1) I learnt about the QuickOSM plugin to retrieve the amenity data.

2) To process queries from the web app, I had to learn how to run QGIS python scripts outside the QGIS application.

3) I needed a thorough understanding of the various functions in QGIS such as centroids, merge, dissolve, etc.

4) I learnt how to make maps using Leaflet in a React app.

Limitations:

1) The project utilises radial distance and not road distance for the proximity preferences.

2) The application does not pinpoint individual houses but only displays regions which contain such houses.

3) The rent data may vary according to the condition of the house, but it was generalised for this project.

4) The user cannot select their interested amenities which are not present in the app.

5) The user may not want to select a bhk type and may only want to enter their monthly budget.

Challenges:

1) Collecting the required data was difficult and I had changed my region of interest for the project several times. Finally, I ended up choosing Mumbai as I was able to find sufficient amenity and suburb data.

2) Choosing the right tech stack and required libraries was difficult because I wanted to incorporate all of the functionalities in the simplest manner.

3) From all of the amenities available, I believe the six I selected are the most significant regarding house selection.

4) Learning how to take user input required for this project. Utilising python and QGIS as well as Leaflet were new techniques for me.