Lab 6 Artifacts - team2345

Members: Valeria Garibaldi, Krischin Layon, Jeevan Sandhu, Justin Wang

Features (user stories) to Implement in Next Sprint:

Listings data:

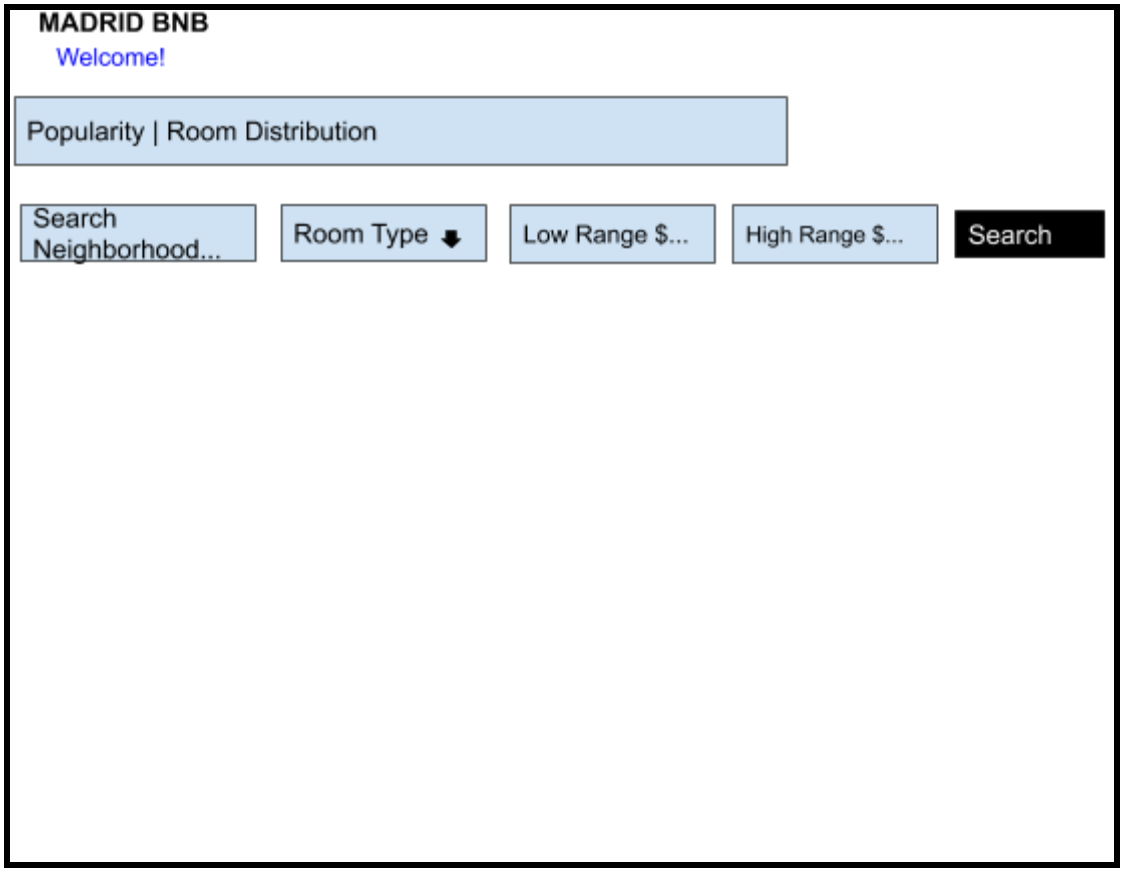
-**Feature 1:** As a user, I want less latency when analytics are calculated or recalculated.

-**Feature 2:** As a user, I want a more user friendly platform.

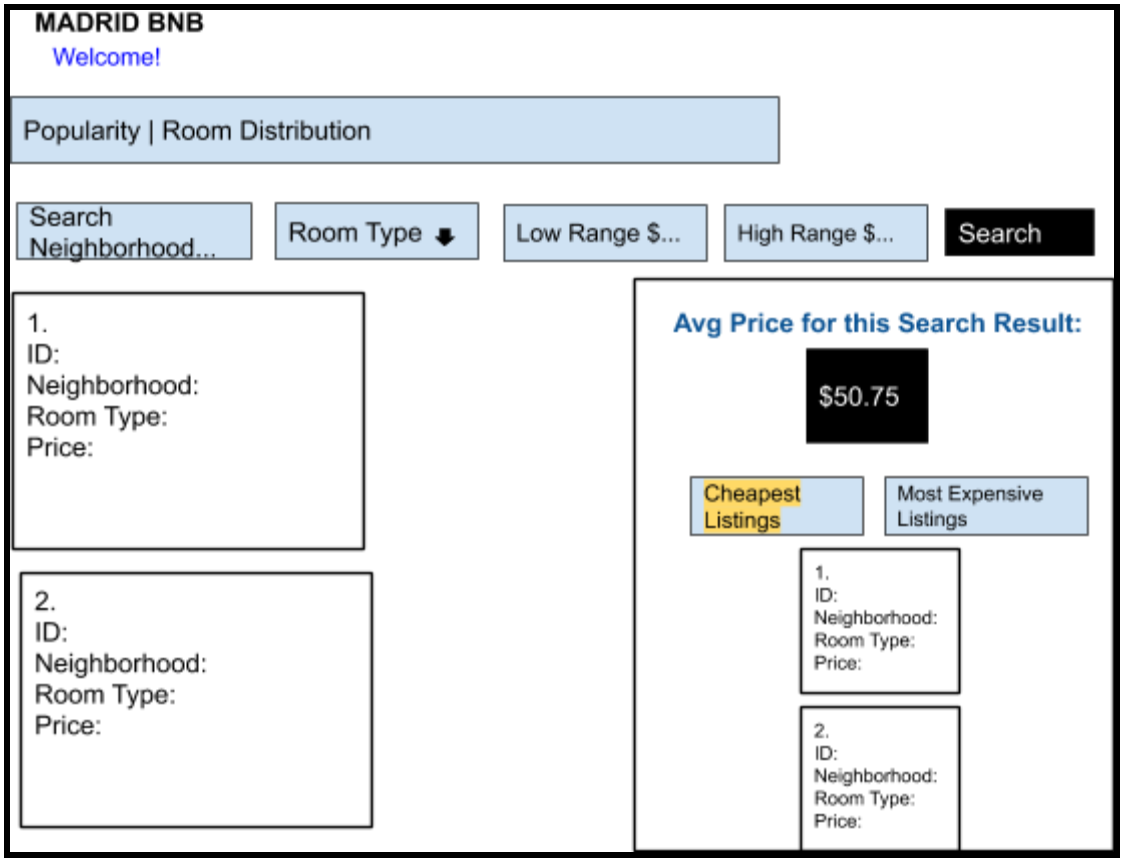
-Feature 3: As a user, I want to see the global average price of all listings without entering an all-encompassing search criteria every single time.

GUI:

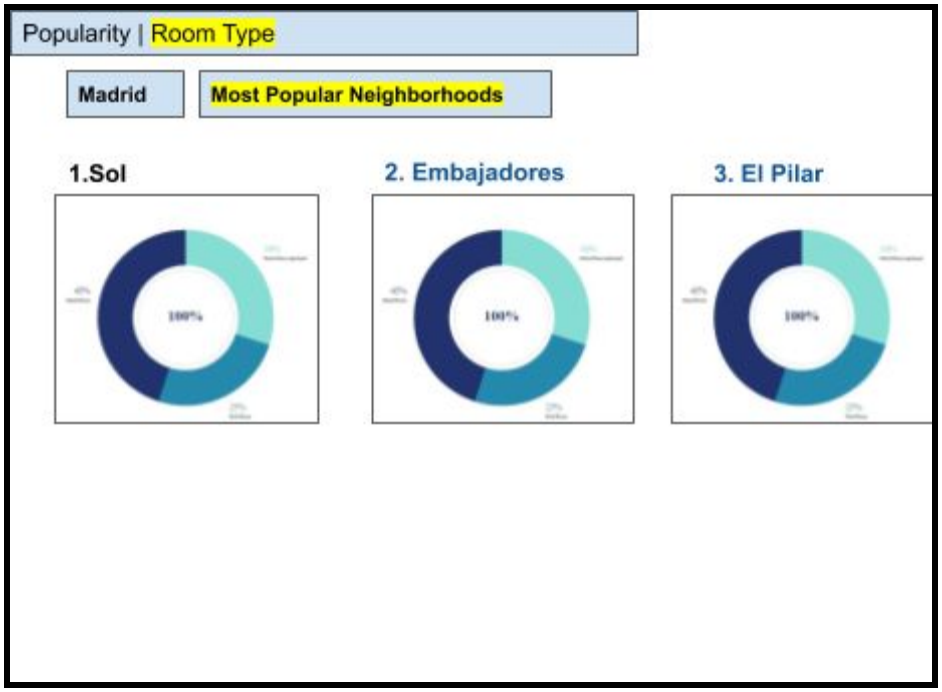
Home Page:



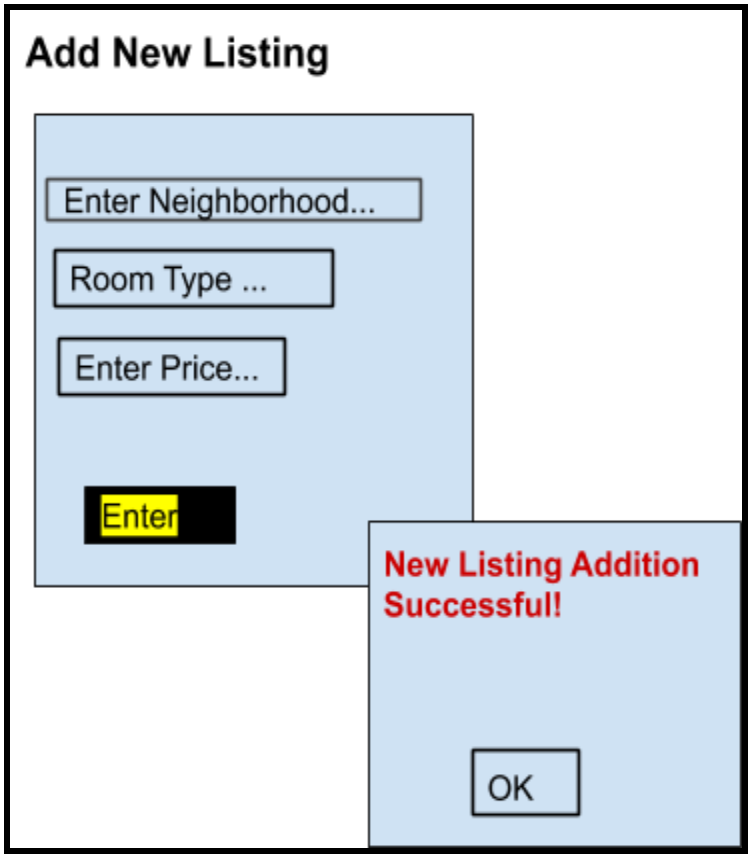
Home Page with Search Results:



Room Distribution Page:



Add Listing Page:



Test Cases:

- **Feature 1 Test Cases:** As a user, I want less latency when analytics are calculated or recalculated.
    - **Test case 1**: As a user, I click on the room distribution button or the average price button to calculate analytics when the app is first initialized.
      Correct Output: The front end displays all appropriate data on the room distribution or average price page. The backend outputs the elapsed time for calculating the analytics to the console.
    - **Test case 2:** As a user, I click on the back button from the room distribution page or from the average price page, and I click the room distribution button or average price button again to recalculate the analytics.
      Correct Output: The front end displays the same data as test case 1. Since there are no modifications, the backend should not recalculate the analytics calculated from test 1. As such, the backend outputs a faster elapsed time for calculating analytics in comparison to the time from test case 1.
    - **Test case 3:** As a user, I add a listing using the add listing button, and then I click the room distribution button or the average price button to recalculate analytics.
      Correct Output: The front end displays appropriately modified data on the room distribution or average price page. Since the local data was modified, the backend recalculates the analytics incrementally and outputs its elapsed time to the console. This time is faster than the time outputted in test case 1.
    - **Test case 4:** As a user, I edit a listing using the edit listing button, and then I click the room distribution button or the average price button to recalculate analytics.

Correct Output: The front end displays appropriately modified data on the room distribution or average price page. Since the local data was modified, the backend again recalculates the analytics incrementally and outputs its elapsed time to the console. This time is faster than the time outputted in test case 1.

- **Test case 5:** As a user, I delete a listing using the delete listing button, and then I click the room distribution button or the average price button to recalculate analytics.
  Correct Output: The front end displays appropriately modified data on the room distribution or average price page. Since the local data was modified, the backend again recalculates the analytics incrementally and outputs its elapsed time to the console. This time is faster than the time outputted in test case 1.

- **Feature 2 Test Cases:** As a user, I want a more user friendly platform.
  - **Test case 1**: As a user, I go to the home page.
    Correct Output: The front end only displays the search bar and menu bar. After I click on “Search”, the price analytics show up on the sidebar separate from the search results.
  - **Test case 2**: As a user, I click on the “Add Listing” button from the home page.
    Correct Output: The button redirects to a new page and asks for the new listing inputs. After clicking on the “Enter” button, I receive a confirmation message saying that the listing was successfully added.
  - **Test case 3**: As a user, I click on the “Room Distribution” tab from the home menu.
    Correct Output: The tab will redirect to a new page. There are two buttons to choose which category I want to see room type distribution for. I click on the “City of Madrid” and a pie chart displays the percentages each room type in Madrid.

- **Feature 3 Test Cases:** As a user, I want to see the global average price of all listings without entering an all-encompassing search criteria every single time.
    - **Test case 1:** As a user I click on the “Average Price” button from the home page.

      Correct Output: The front end displays the correct average price of all Madrid Listings.

To-Do List

Done list of last sprint:

- Modified HTML file for popularity analytics.

  [finished by Valeria and verified by Krishcin]
- Modified HTML file for room distribution analytics.

  [finished by Valeria and verified by Krishcin]
- Modified CSS file for popularity analytics. (Part 2 Analytics)

  [finished by Valeria and verified by Krishcin]
- Modified CSS file for room distribution analytics.

  [finished by Valeria and verified by Krishcin]
- Modified container functionality to Popularity HTML file to display contents.

  [finished by Krischin and verified by Everyone]
- Modified container functionality to Room Distribution HTML file to display contents.

  [finished by Krischin and verified by Everyone]
- Implemented Angular Components to handle modified API call (Part 2 Analytics)

  [finished by Krischin and verified by Everyone]
- Implemented Angular Services to handle modified API call (Part 2 Analytics)

  [finished by Krischin and verified by Everyone]
- Set up API functionality to send new JSON with analytic data upon receiving GET Request

  [finished by Justin and verified by Everyone]
- Integrate API with Data Layer popularity analytic functions. (Part 2 Analytics)

  [finished by Justin and verified by Jeevan]

- Integrate API with Data Layer room distribution analytic functions. (Part 2 Analytics)
  [finished by Justin and verified by Jeevan]
- Implement backend functionality to calculate most popular listings in Madrid.
  [finished by Jeevan and verified by Justin]
- Implement backend functionality to calculate the most popular listings in the most popular neighborhoods in Madrid.
  [finished by Jeevan and verified by Justin]
- Implement backend functionality to calculate room type distribution by percentage in the city of Madrid as well as in the most popular neighborhoods.
  [finished by Jeevan and verified by Justin]
- Conducted and confirmed Feature 1, 2, and 3 Test Cases from Sprint-5
  [finished by Jeevan and Justin and verified by everyone]
- Completed Sprint 5 Artifact
  [finished by everyone and verified by everyone]
- Record Sprint 5 Demo
  [finished by everyone and verified by everyone]

To-Do for <u>next sprint</u>:

- Implement a clean frontend framework
- Add the Pie Chart Framework
- Store most up-to-date analytics in local data store
  - "top_neighborhoods"
    - list of “popular_n” objects of top neighborhoods in Madrid
  - "top_listings"
    - list of “listing” objects of top listings in Madrid
  - "room_dist_data"
    - list of “room_distribution” objects (starting with Madrid) of room distributions of madrid and top 3 neighborhoods in Madrid
- Incrementally update analytics based on changes in stored data

- Incrementally update average price of listing when user adds, edits, or deletes a listing
- Incrementally update three most expensive listings analytic when user adds, edits, or deletes a listing
- Incrementally update three least expensive listings analytic when user adds, edits, or deletes a listing
- Incrementally update three most popular neighborhoods analytic when user adds, edits, or deletes a listing
- Incrementally update three least popular neighborhoods analytic when user adds, edits, or deletes a listing
- Incrementally update room type distribution analytic when user adds, edits, or deletes a listing

To-do if there's time:

- Implement fourth drop down for room type - Shared room
- Add confirmation message when deleting, adding, and editing listings
- Add separate tab "Pricings" to view total average and top 3 expensive and cheap listings in all of madrid
- Implement add review function to web app
    - Reviews page with review post request /review
        - Json
            - {
                "id": 0000 //id of listing to increment num reviews
              }
    - Backend to receive post request and increment num reviews in csv
    - Backup/import for reviews