

MY BOSTON HOOD

By Atharva Hankare | Keya Goswami | Priyank Bagad

BACKGROUND

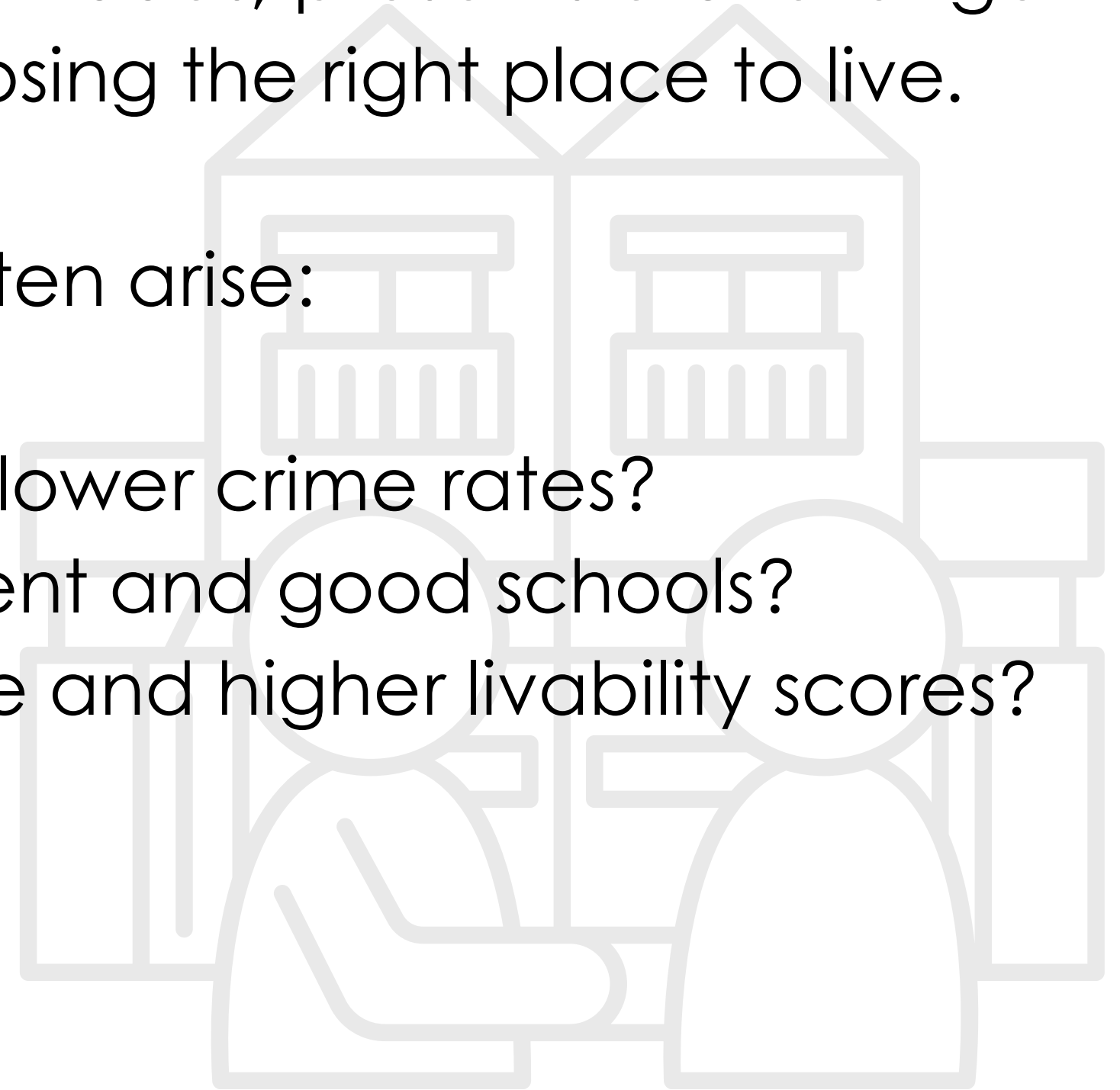
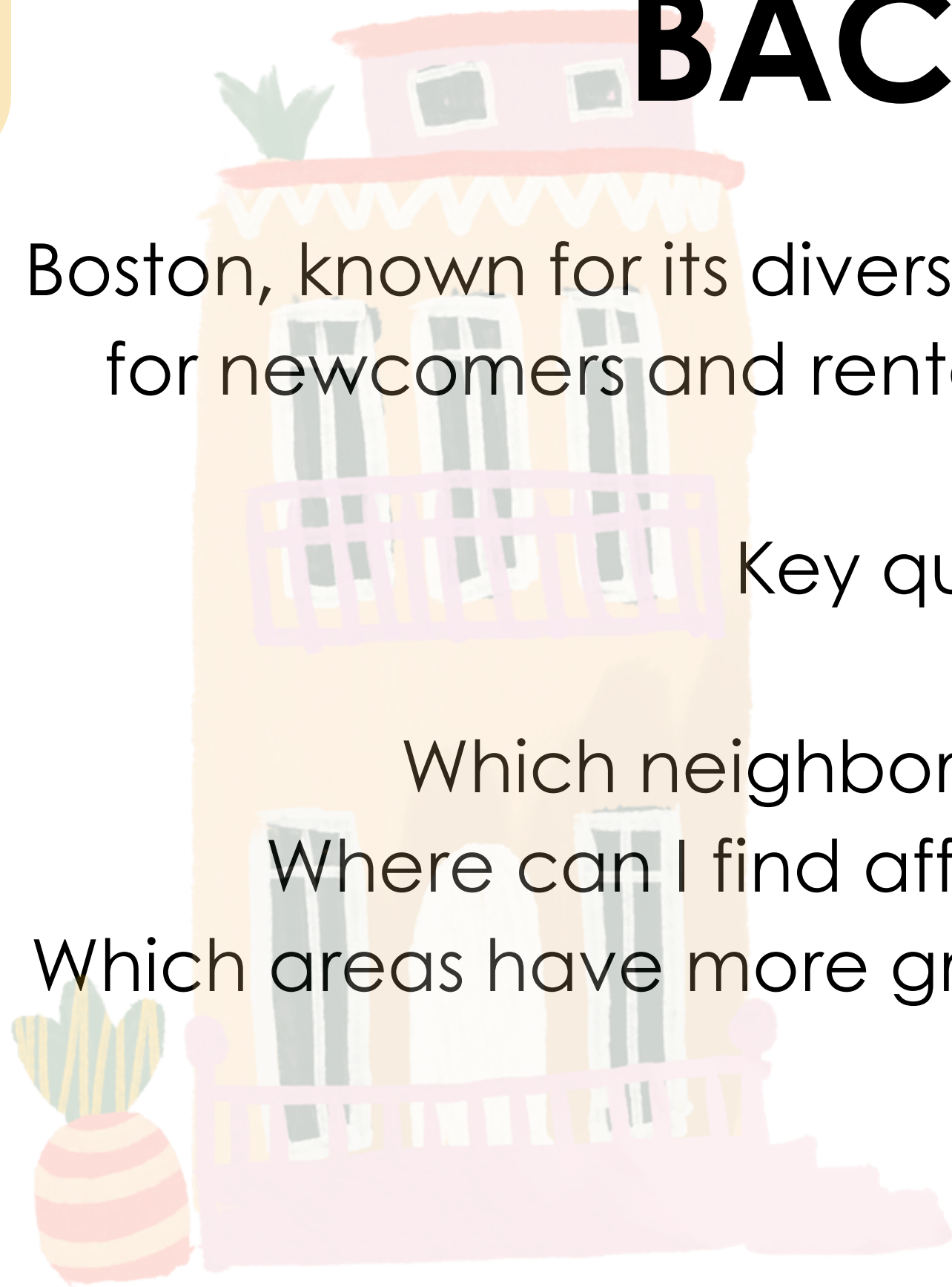
Boston, known for its diverse neighborhoods, presents a challenge for newcomers and renters in choosing the right place to live.

Key questions often arise:

Which neighborhood has lower crime rates?

Where can I find affordable rent and good schools?

Which areas have more green space and higher livability scores?

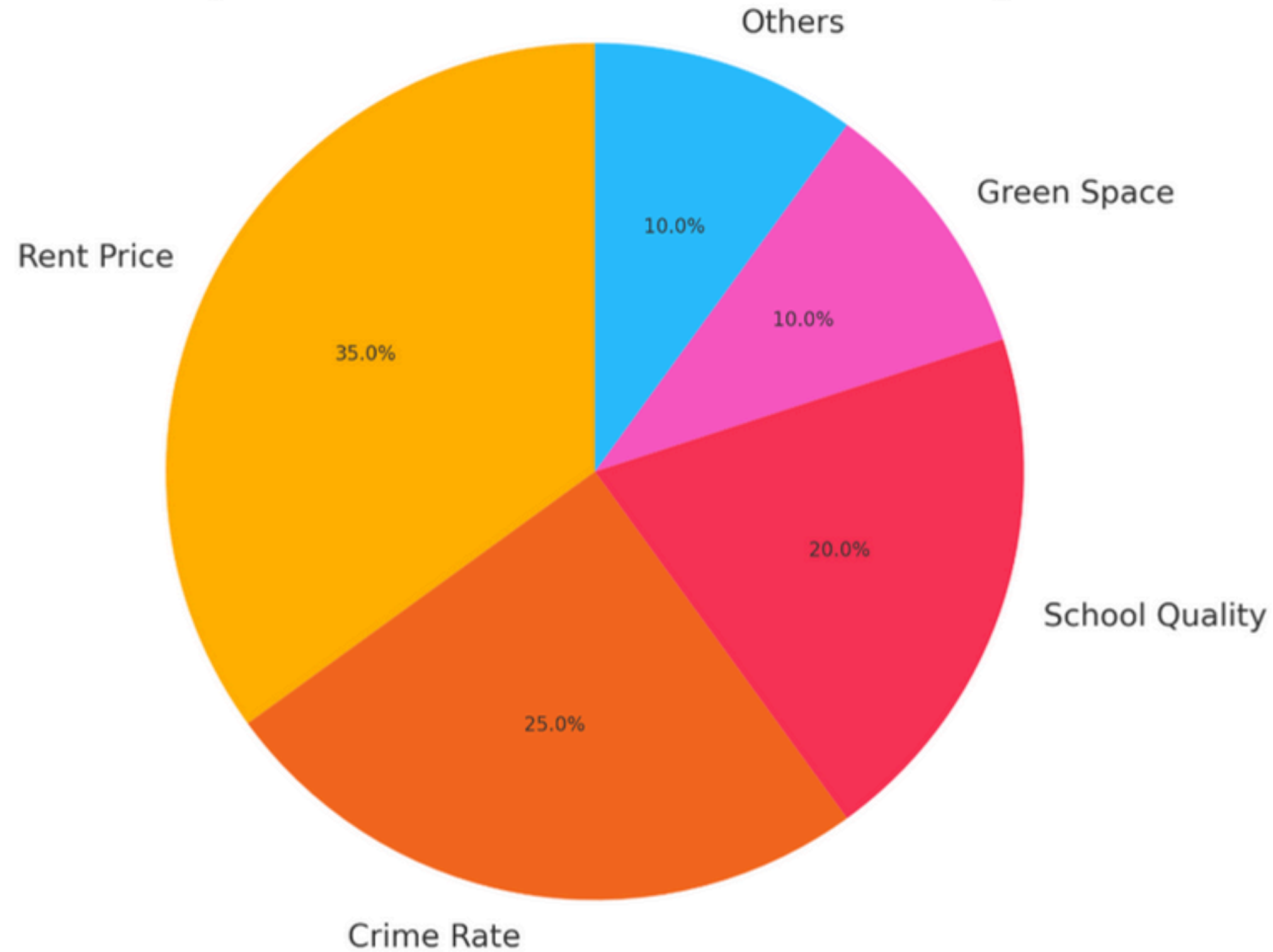


PROBLEM DESCRIPTION

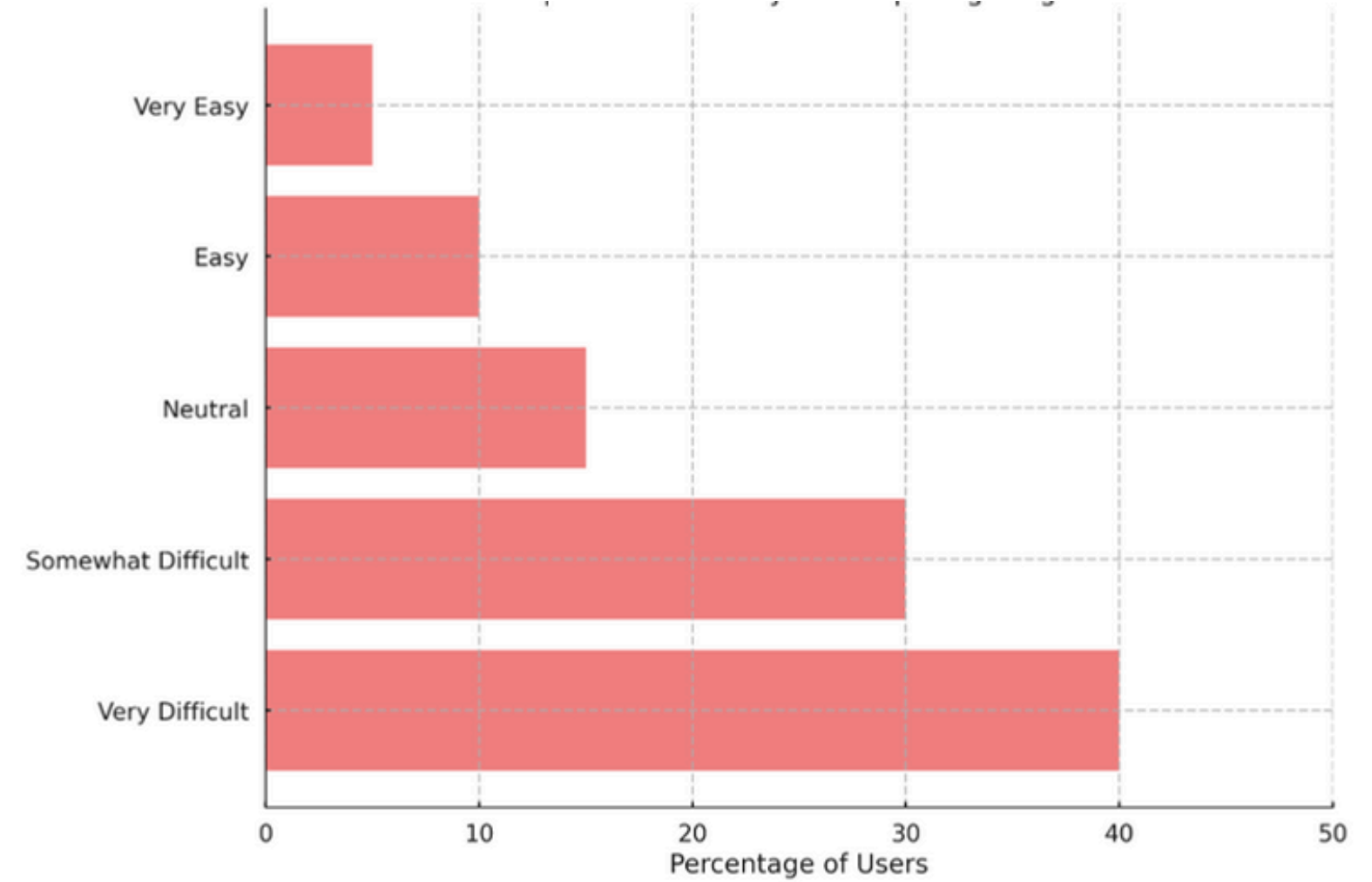
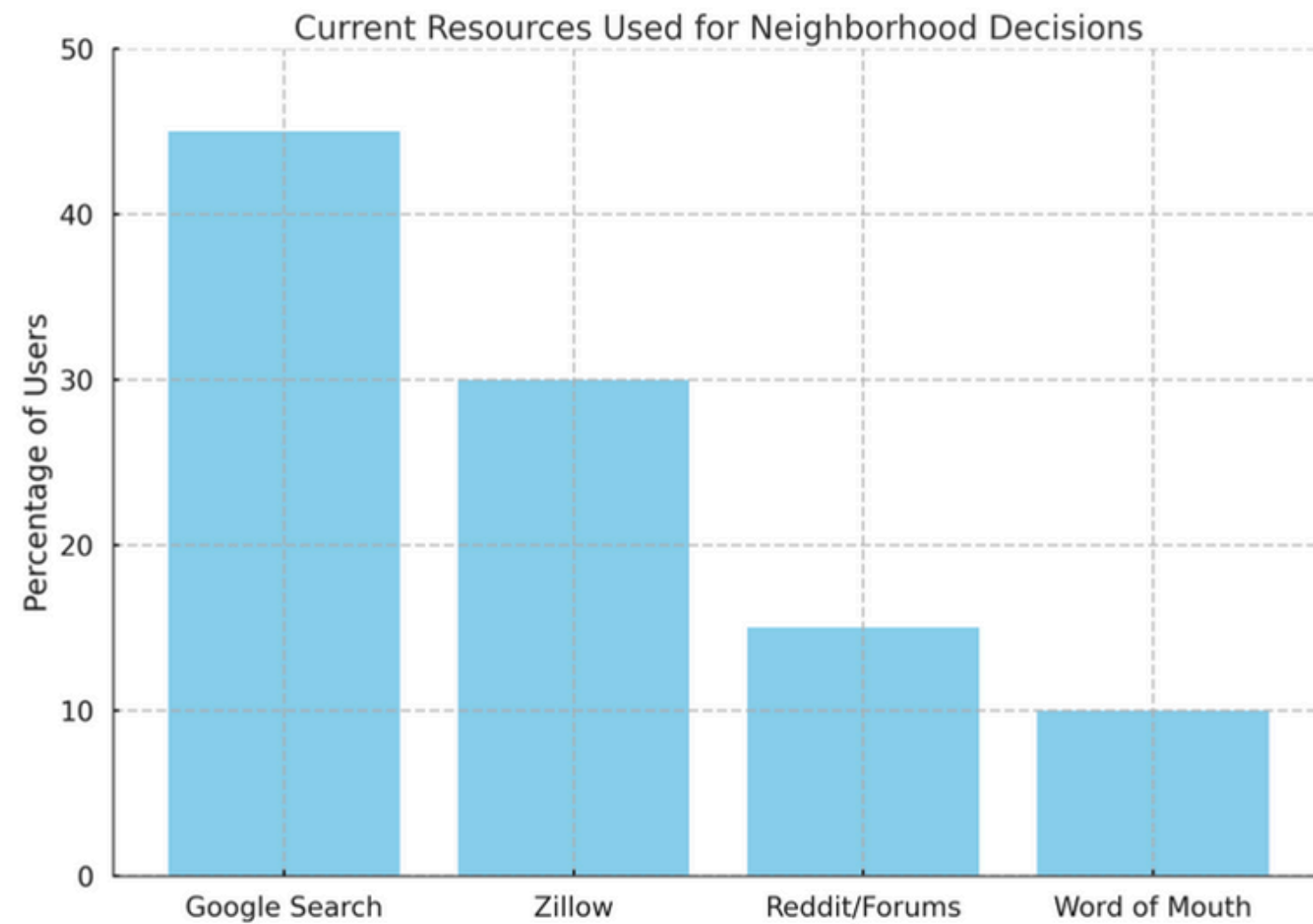
Choosing the right neighborhood in Boston can be difficult due to varying factors like rent, crime, schools, and green space. There's no simple tool that presents this data in one place. MyBostonHood solves this by offering a user-friendly desktop app where users can view, explore, and compare neighborhoods visually and statistically. It uses custom-built data structures like a HashMap for storage, Binary Search Tree for sorting, and Stack for navigation. Admins can add or update neighborhoods, while users can compare based on livability scores. The platform empowers better, data-driven decisions when choosing where to live in Boston.

USER RESEARCH

Top Decision Factors Identified in User Survey

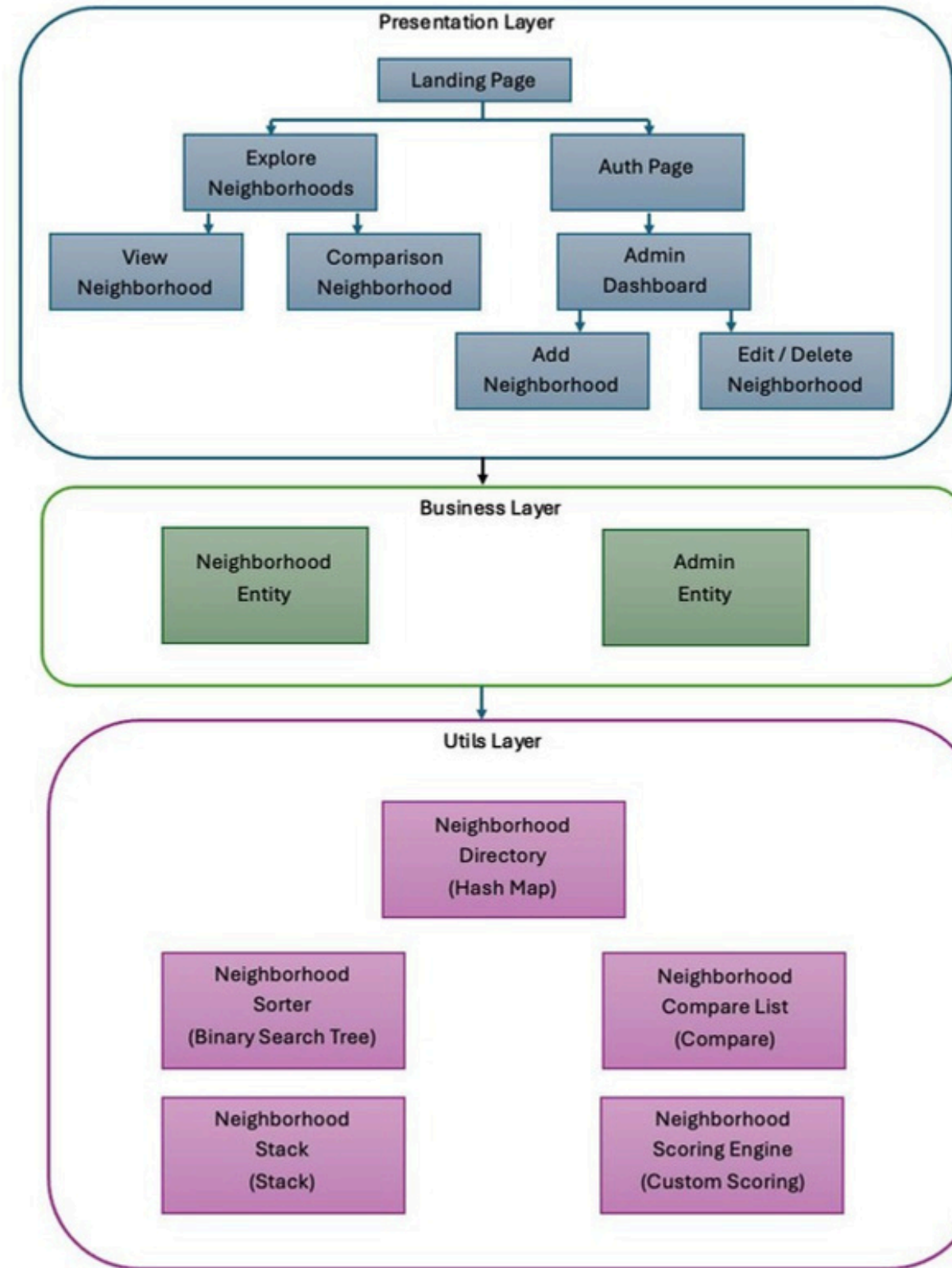


USER RESEARCH



ANALYSIS

System Design



ANALYSIS

Challenges Encountered

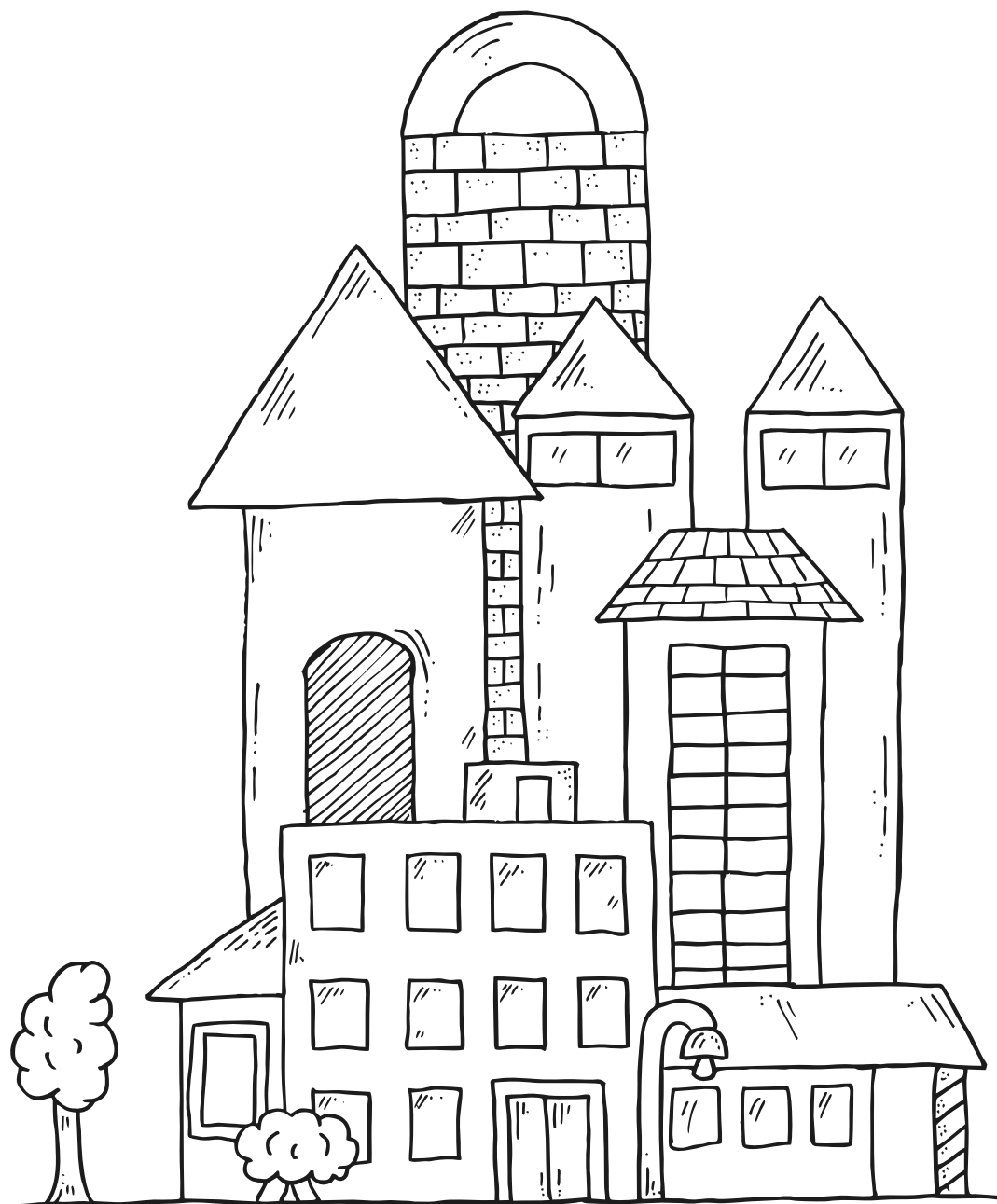
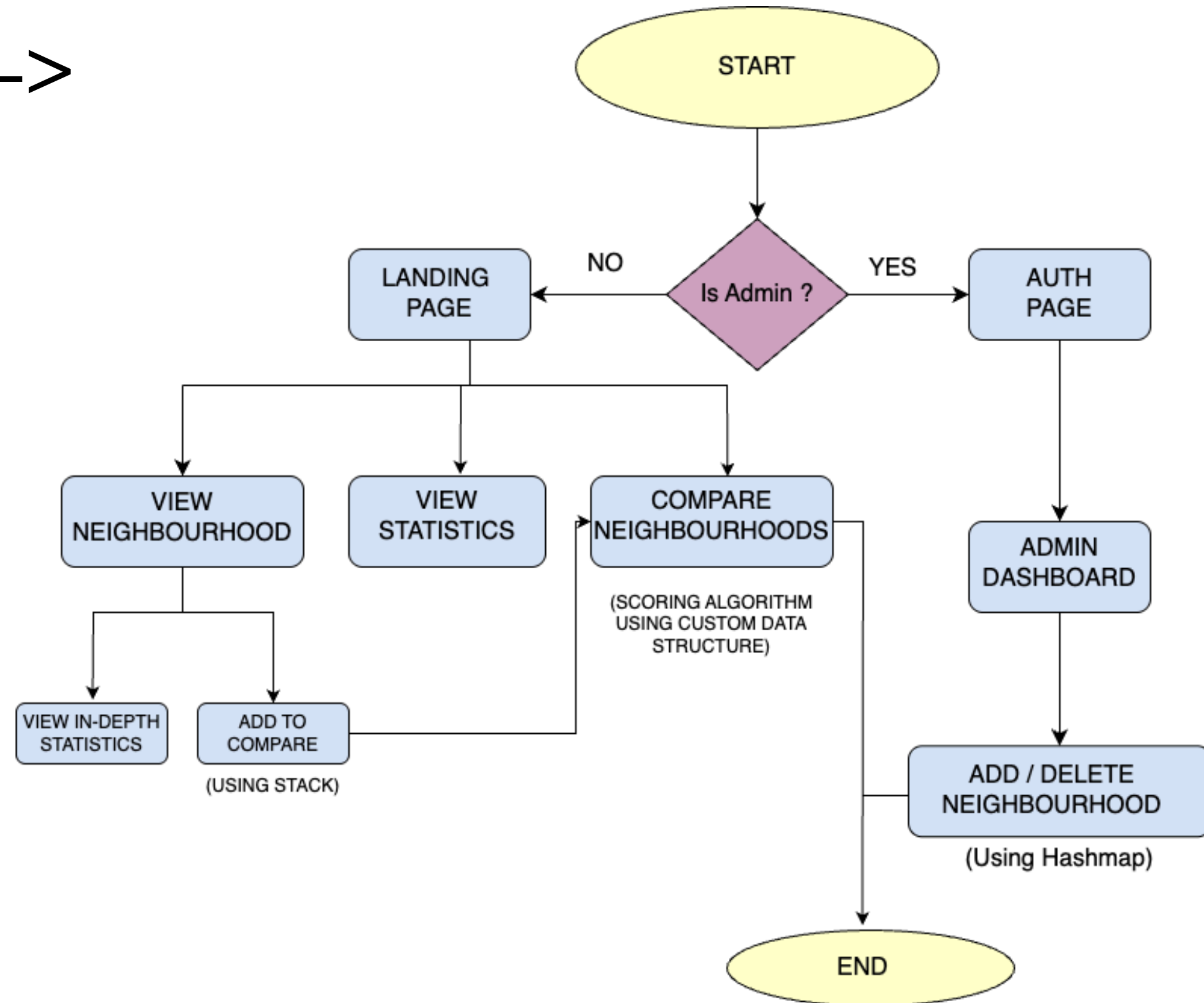
1. Maintaining data consistency between stored neighborhoods and comparator views
2. Avoiding duplicate entries in neighborhood
3. User Navigation tracking required a simple yet effective mechanism like a stack to avoid unnecessary UI complexity
4. Scoring engine tuning to balance real-world significance

Potential Improvements

1. Switching from Binary Search Tree to AVL Tree or other self-balancing trees for optimal sorted insertions even with large datasets
2. Adding Persistent storage using databases
3. Extending the scoring engine with adding other factors
4. Pagination & search optimizations for fast suggestion retrievals

ANALYSIS

Flow Chart -->



IMPLEMENTATION

HashMap :

- a. Acts as the primary in memory store for neighborhoods
- b. Enables constant-time lookup, insertion and updates
- c. Perfect for managing and verifying neighborhood entries

BST :

- a. Supports in-order traversal for sorted views based on any comparable metric
- b. Allows dynamic insertion while maintaining order, which avoids needing to sort lists repeatedly

Stack :

- a. Mimics the browser-style navigation to allow users to move back through previous views or comparison steps
- b. Simple LIFO (Last in First Out) mechanism helps track the most recent actions

Weighted Scoring Engine :

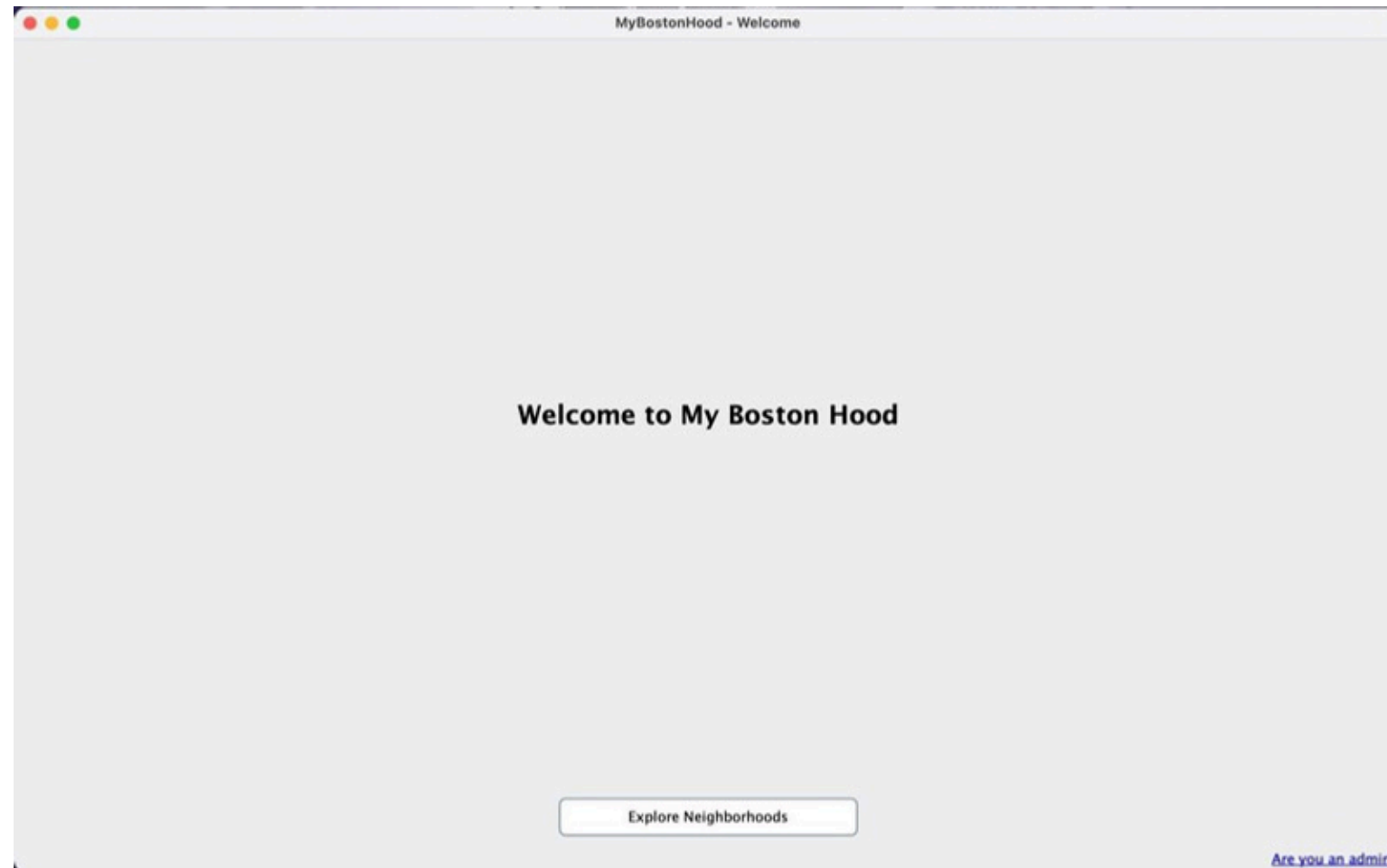
Implemented through comparators passed to the sorter for different metrics.



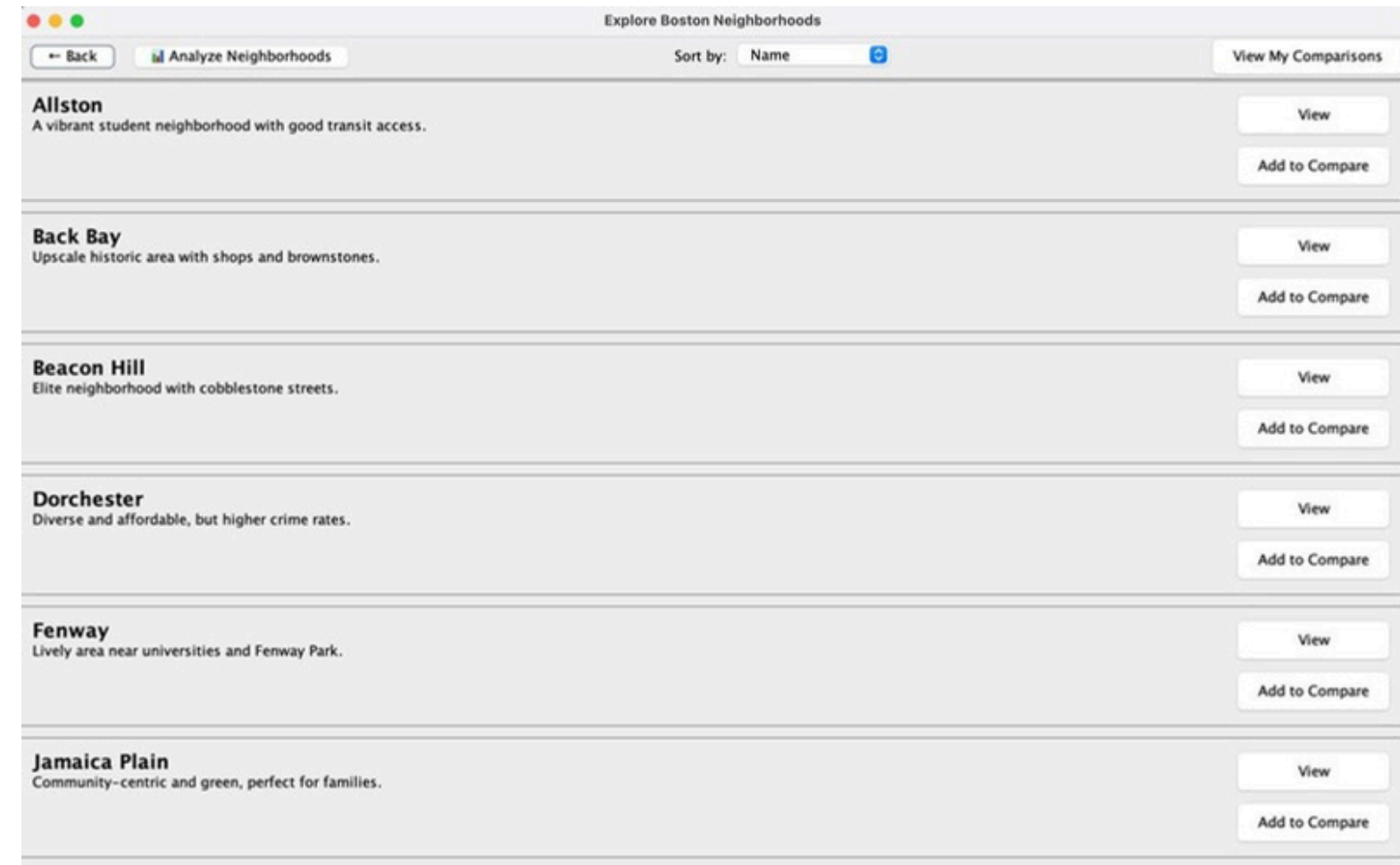
Comparator Based List :

- a. Implements flexible sorting techniques for multiple user-selected attributes (For ex. Sort by green space, rent etc.)

RESULTS

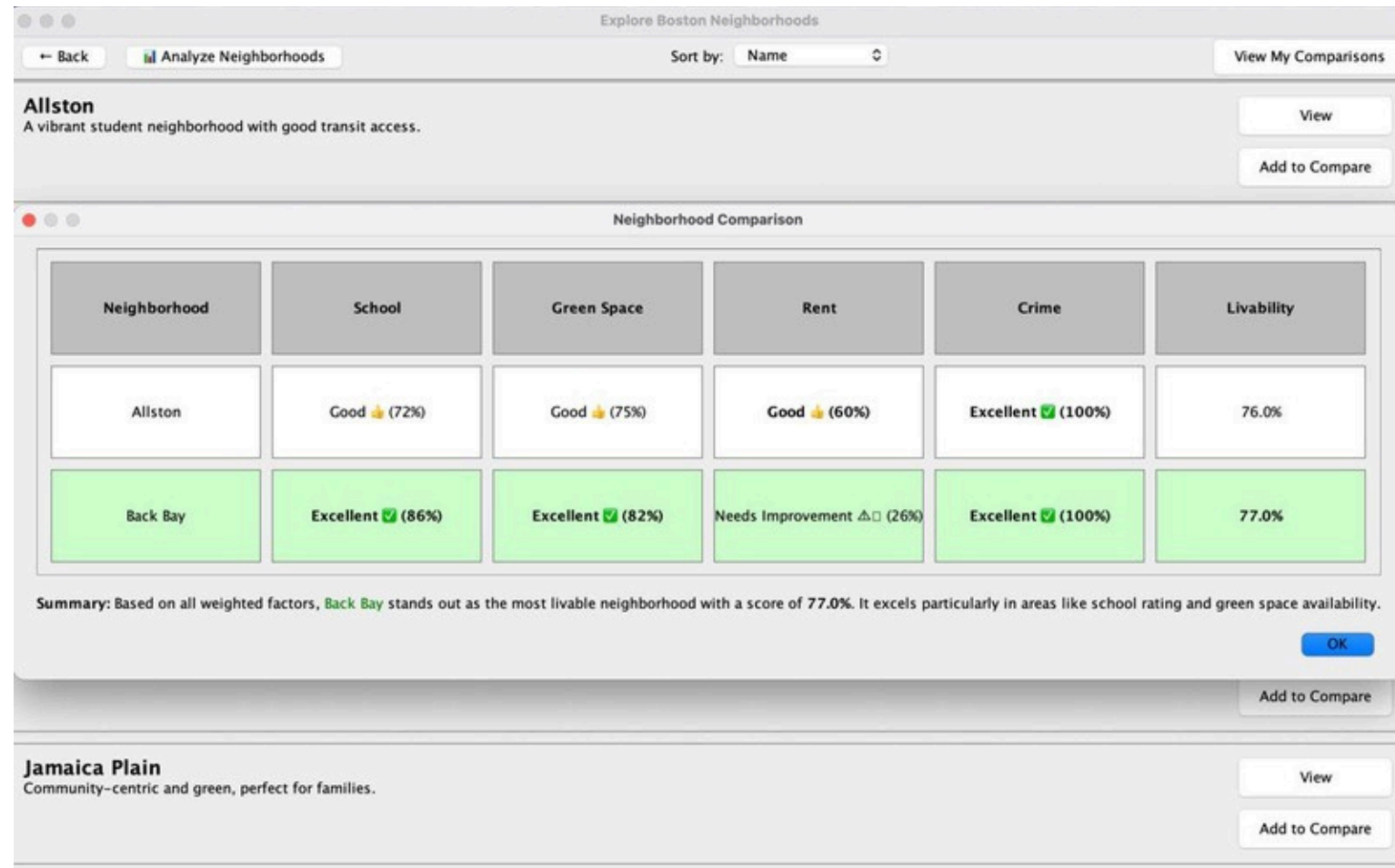


Landing Page



Explore Neighborhood Page

RESULTS

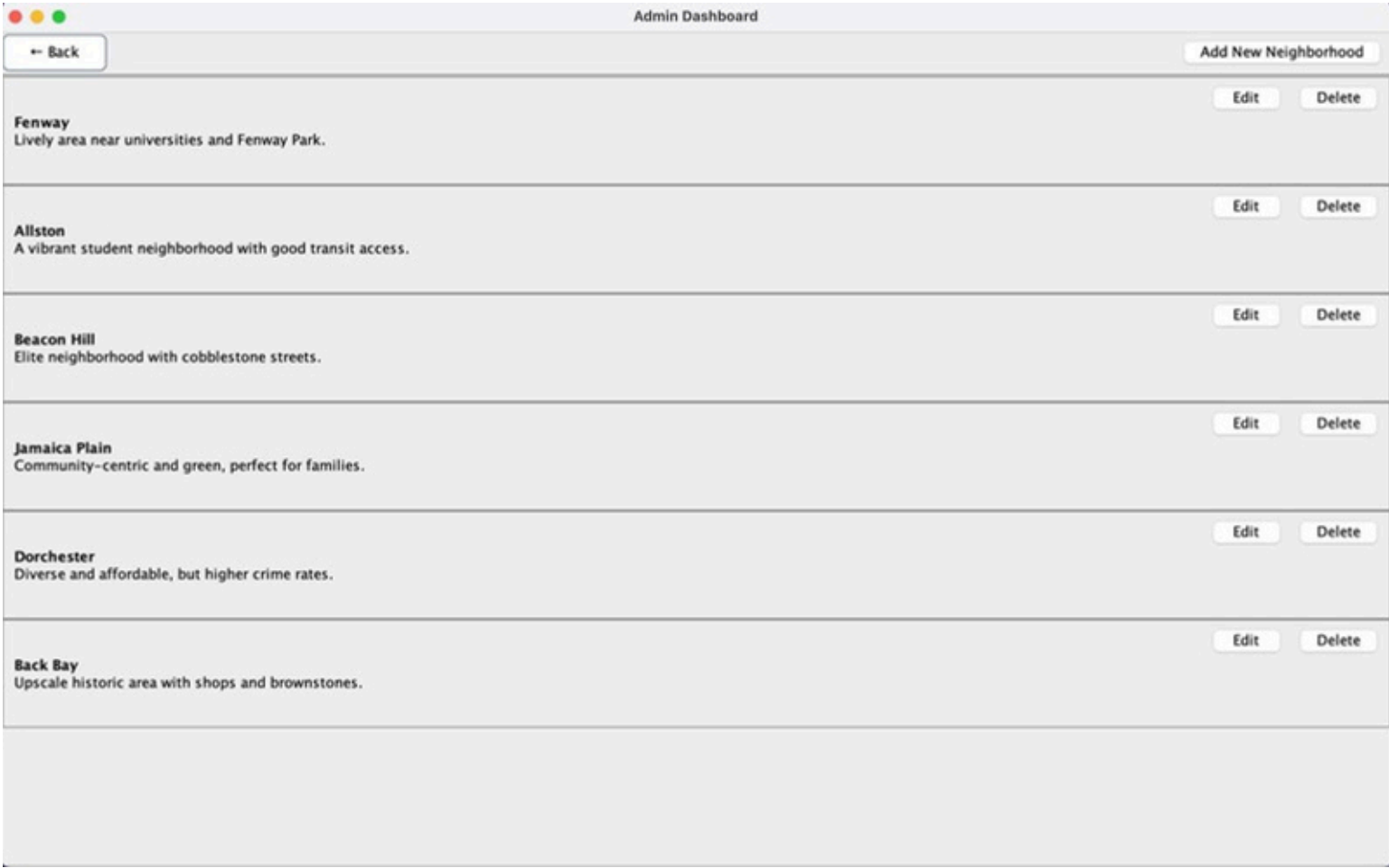


Comparison Metrics



Neighborhood Analysis

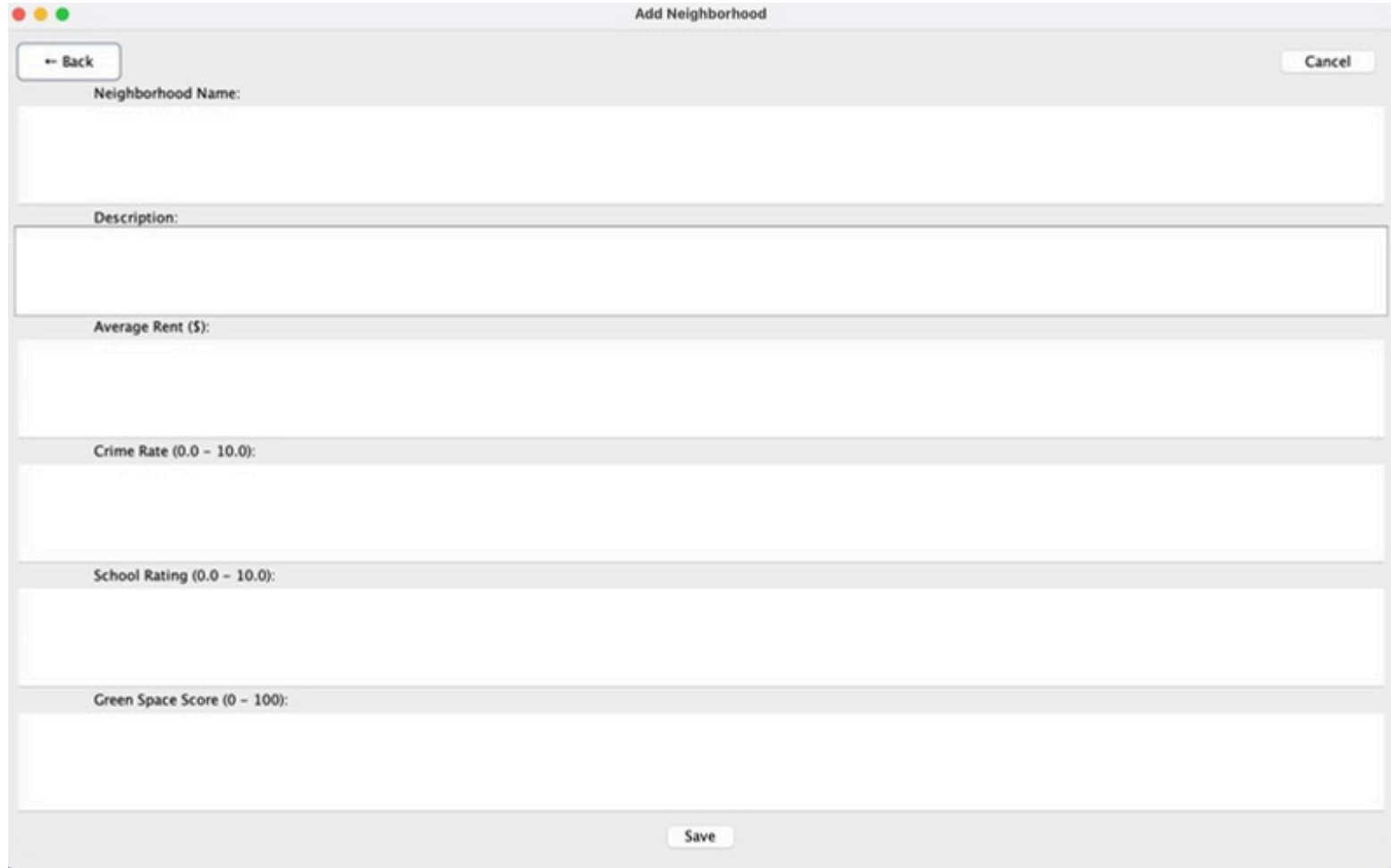
RESULTS



The Admin Dashboard UI mockup features a light gray header bar with a title "Admin Dashboard", a "Back" button, and an "Add New Neighborhood" button. Below the header is a table with five rows, each representing a neighborhood. Each row contains the neighborhood name, a brief description, and "Edit" and "Delete" buttons. The neighborhoods listed are Fenway, Allston, Beacon Hill, Jamaica Plain, and Back Bay.

Neighborhood	Description	Edit	Delete
Fenway	Lively area near universities and Fenway Park.	Edit	Delete
Allston	A vibrant student neighborhood with good transit access.	Edit	Delete
Beacon Hill	Elite neighborhood with cobblestone streets.	Edit	Delete
Jamaica Plain	Community-centric and green, perfect for families.	Edit	Delete
Dorchester	Diverse and affordable, but higher crime rates.	Edit	Delete
Back Bay	Upscale historic area with shops and brownstones.	Edit	Delete

Admin Dashboard



The Add / Edit Neighborhood Page UI mockup features a light gray header bar with a title "Add Neighborhood", a "Back" button, and a "Cancel" button. Below the header is a form with six input fields: "Neighborhood Name:", "Description:", "Average Rent (\$):", "Crime Rate (0.0 - 10.0):", "School Rating (0.0 - 10.0):", and "Green Space Score (0 - 100):". A "Save" button is located at the bottom right of the form.

Neighborhood Name:

Description:

Average Rent (\$):

Crime Rate (0.0 - 10.0):

School Rating (0.0 - 10.0):

Green Space Score (0 - 100):

Save

Add / Edit Neighborhood Page

DISCUSSION

Insights from Comparison :

The livability scoring algorithm helped surface neighborhoods like Back Bay and Beacon Hill as more livable due to high school ratings and green space scores, despite higher rent.

Tradeoffs in Scoring Model :

While the weighted scoring provides a fair baseline, some metrics like crime rate or rent could have subjective importance based on the user's context — future versions could allow user-customized weights.

Value of Visual Analytics :

Integrating JFreeChart to visualize neighborhood factors brought clarity to otherwise complex numeric data — helping users make informed, data-driven comparisons.

CONCLUSION & FUTURE SCOPE

1. Shows how to combine custom Data Structures for performance with simpler UI Design
2. Can be extended to web/mobile platforms
3. Can be used to design for city planner, location-based planner

MEMBER'S CONTRIBUTION

Meet The Team!



Atharva Hankare

1. Model Creation
2. Charts and analysis
3. Binary Search Tree
4. User Interface



Keya Goswami

1. HashMap
2. Admin Dashboard
3. Validations and Exceptions
4. User Interface



Priyank Bagad

1. Stack based Navigation
2. Weighted Scoring Algorithm
3. Comparator and List
4. User Interface



THANK YOU