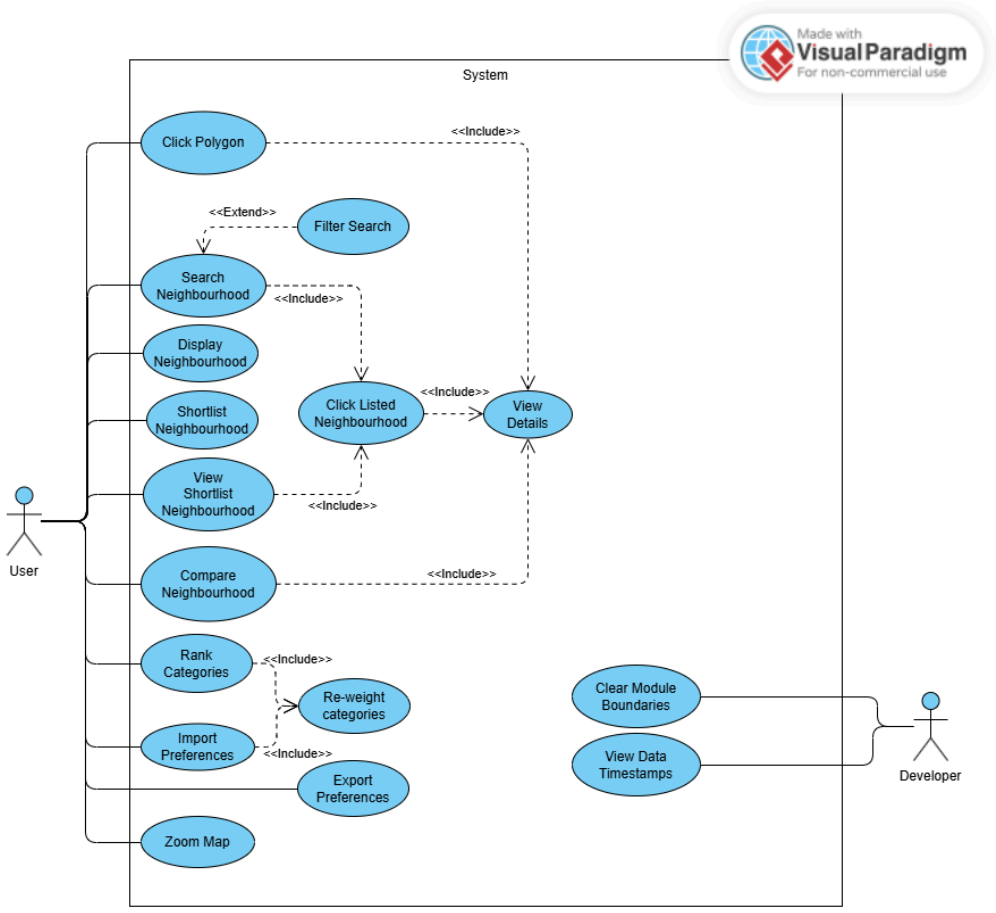
Use Case Diagram



Use Case Description

Use Case ID:	1		
Use Case Name:	Click Polygon		
Created By:	Loong Kiat	Last Updated By:	Loong Kiat
Date Created:	29/8/2025	Date Last Updated:	29/8/2025

Actor:	User
Description:	Users can click the polygons on the map to view more details of the selected neighbourhood.
Preconditions:	1. All calculations are done so that the polygons are coloured according to the score.
Postconditions:	1. The system is displaying a zoomed-in map of the selected neighbourhood. 2. The system is displaying buttons to see more details of the neighbourhood and to see the breakdown for rating score.
Flow of Events:	1. The user clicks on a polygon. 2. The system opens up a new page to show a zoomed-in version of the selected neighbourhood, along with buttons to see more details of the neighbourhood and the breakdown for rating score.
Alternative Flows:	Nil
Exceptions:	Nil

Includes:	View Details
Notes and Issues:	Nil

Use Case ID:	2		
Use Case Name:	Search Neighbourhood		
Created By:	Loong Kiat	Last Updated By:	Loong Kiat
Date Created:	29/8/2025	Date Last Updated:	29/8/2025

Actor:	User
Description:	Users can click on the search bar to search for a specific neighbourhood.
Preconditions:	Nil
Postconditions:	1. Searched results are listed out below the search bar
Flow of Events:	1. The user clicks on the search bar and enters a location name. 2. The system processes the text in the search bar. 3. The system lists out all matching results below the search bar.
Alternative Flows:	Nil
Exceptions:	EX1: Searched location does not exist 1. The system displays a message “No results found”
Includes:	Nil

Notes and Issues:	Nil

Use Case ID:	3		
Use Case Name:	Filter Search		
Created By:	Mun Kuan	Last Updated By:	Mun Kuan
Date Created:	29/8/2025	Date Last Updated:	29/8/2025

Actor:	User
Description:	The user would be able to filter the search result according to their budget and facilities available.
Preconditions:	1. The user should have clicked on the search function button.
Postconditions:	1. The system displays the listed neighbourhoods according to the filters and neighbourhood ratings.
Flow of Events:	1. The user clicks on the filter button. 2. The system will open a page with checkboxes to select facilities as well as a slider to determine the budget.
Alternative Flows:	1. The users can choose not to include any filters. 2. The system will display the neighbourhoods according to the neighbourhood rating only.
Exceptions:	Nil
Includes:	Nil

Notes and Issues:	Nil

Use Case ID:	4		
Use Case Name:	Display Neighbourhood		
Created By:	Mun Kuan	Last Updated By:	Mun Kuan
Date Created:	29/08/2025	Date Last Updated:	29/08/2025

Actor:	User
Description:	The users can view a list of all available neighbourhoods.
Preconditions:	1. The user has searched for a neighbourhood.
Postconditions:	1. The system opens the list of ranked neighbourhoods according to the search filters and neighbourhood ratings.
Flow of Events:	1. The user clicks enter to confirm the search. 2. The system displays the list of ranked neighbourhoods according to the search filters and neighbourhood ratings. 3. The user clicks the desired neighbourhood to view more details.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	Nil
Notes and Issues:	Nil

Use Case ID:	5		
Use Case Name:	Shortlist Neighbourhood		
Created By:	Xi Quan	Last Updated By:	Xi Quan
Date Created:	27/08/2025	Date Last Updated:	27/08/2025

Actor:	User
Description:	Users can save a shortlist of neighbourhood polygons.
Preconditions:	Have a neighbourhood polygon selected
Postconditions:	Neighbourhood polygon is saved into a shortlist
Flow of Events:	1. User taps on the bookmark button 2. Neighbourhood polygon is added to shortlist 3. Bookmark icon changes
Alternative Flows:	AF-S2: If polygon is already inside shortlist 1. Neighbourhood polygon is removed from shortlist 2. System proceeds to step 3
Exceptions:	Nil
Includes:	Nil
Notes and Issues:	Nil

Use Case ID:	6		
Use Case Name:	View Shortlist Neighbourhood		
Created By:	Samrath	Last Updated By:	Mun Kuan
Date Created:	29/08/2025	Date Last Updated:	01/09/2025

Actor:	User
Description:	Users can view shortlisted neighbourhoods.
Preconditions:	1. Have neighbourhoods added to the shortlist.
Postconditions:	2. System opens neighbourhood shortlist.
Flow of Events:	1. The user clicks the shortlist button in the bottom navbar. 2. The system displays a shortlist of saved neighbourhoods. 3. The user clicks the desired neighbourhood to view more details.
Alternative Flows:	AF-S2-If User has no neighbourhoods added to shortlist 1. The user clicks the shortlist button in the bottom navbar. 2. System displays screen with “Empty shortlist message”
Exceptions:	Nil
Includes:	Nil
Notes and Issues:	Nil

Use Case ID:	7		
Use Case Name:	Click Listed Neighbourhood		
Created By:	Calvin Kuan	Last Updated By:	Calvin Kuan
Date Created:	29/08/2025	Date Last Updated:	29/08/2025

Actor:	User
Description:	Users can view a list of all available neighbourhoods.
Preconditions:	1. The system requires the user to have searched a specific neighbourhood.
Postconditions:	1. The system displays a list of all neighbourhoods.
Flow of Events:	1. The user clicks on the search bar 2. The system displays a list of all available neighbourhoods
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	View Details
Notes and Issues:	Nil

Use Case ID:	8		
Use Case Name:	Compare Neighbourhood		
Created By:	Xi Quan	Last Updated By:	Xi Quan
Date Created:	27/08/2025	Date Last Updated:	29/08/2025

Actor:	User
Description:	Users can get a comparison overview of 2 neighbourhood polygons side by side.
Preconditions:	1. There are at least 2 neighbourhoods available
Postconditions:	1. System displays an overview comparing the 2 selected neighbourhood
Flow of Events:	1. User taps on the “Compare Neighbourhood” button 2. System enters the compare neighbourhood page 3. User selects 2 neighbourhoods 4. The system generates neighbourhood ratings for selected neighbourhoods. 5. The system displays ratings on a radar chart.
Alternative Flows:	Nil
Exceptions:	Nil
Includes:	View Details

Notes and Issues:	Nil

Use Case ID:	9		
Use Case Name:	View Details		
Created By:	Mun Kuan	Last Updated By:	Mun Kuan
Date Created:	01/09/2025	Date Last Updated:	01/09/2025

Actor:	User
Description:	Users can view details of neighbourhood rating, facilities available, categories and price trend.
Preconditions:	1. The user had chosen the location(s) to view details on. 2. Calculations for all the categories have finished. 3. Successfully pulled the relevant datasets or api for the facilities. 4. Successfully pulled the dataset for the price trend.
Postconditions:	1. Zoomed-in map view of the selected neighbourhood, with the facilities highlighted on the map. 2. The system is displaying a filled radar chart with five axes 3. Line graph that shows the price trend of the selected neighbourhood
Flow of Events:	1. The system opens a popup which contains two buttons, namely, Rating and Details. 2. Users can click on the Rating button to view the neighbourhood rating as well as the categories breakdown, displayed on a 5 axis radar chart for each category. 3. Users can click on the Details button to view the price history on a price trend line and facilities available within that neighbourhood. 4. For viewing facilities, there would be a dropdown button for filtering facilities.

	5. The user clicks the filter to select the facilities to be highlighted on the map. 6. The system highlights the position of the selected facilities.
Alternative Flows:	AF3: 2 Neighbourhoods are selected 1. The user determines 2 neighbourhoods to compare between. 2. The user confirms their choices by clicking a button named “compare”. 3. The system will open a new page displaying the categories score of the desired neighbourhoods in an overlaid radar chart and the price trend line AF4: The user did not select any facilities 1. The system does not highlight any position on the map and displays an empty map.
Exceptions:	EX1: Facilities do not exist
Includes:	Nil
Notes and Issues:	Nil

Use Case ID:	10		
Use Case Name:	Rank Categories		
Created By:	Loong Kiat	Last Updated By:	Loong Kiat
Date Created:	25/8/2025	Date Last Updated:	25/8/2025

Actor:	User
Description:	Users are prompted to select and rank preferences like budget, schools, gyms etc and these will affect the initial colouring of the polygon that they see on the map.
Preconditions:	1. First time using this mobile app
Postconditions:	1. Preferences get saved and subsequent calculations will reflect it. 2. Initial polygon colour will be initialised based on these preferences.
Flow of Events:	1. The user launches the app. 2. The system prompts the user to rank and select preferences that are important to them. 3. The user drags whatever is the most important to the top and checks the checkboxes for amenities that are important. 4. The system saves the preferences and calculates initial values for the polygons to be displayed on the map. 5. The system displays polygons varying from green to red on the map.
Alternative Flows:	AF-3: If the user attempts to skip

	1. The system displays the message “You are not allowed to skip this” 2. The system returns to step 2
Exceptions:	Nil
Includes:	Nil
Notes and Issues:	Nil

Use Case ID:	11		
Use Case Name:	Re-weight Categories		
Created By:	Loong Kiat	Last Updated By:	Loong Kiat
Date Created:	27/08/2025	Date Last Updated:	27/08/2025

Actor:	User
Description:	Users can choose to re-weight the categories and the system will recalibrate the scores. The newly calibrated scores will then be reflected by the colour of the polygons on the map.
Preconditions:	Nil
Postconditions:	1. The color of the polygons changed. 2. Preferences saved in the app are changed.
Flow of Events:	1. The user clicks on the preferences tab from the bottom navigation bar. 2. The system displays the current preferences for the different categories. 3. The user changes the weight values for the categories. 4. The user clicks the save button. 5. The system saves the new weight values for the different categories. 6. The system calculates new values and displays new colours for the polygons.
Alternative Flows:	AF-3: The user clicks on other tab on the bottom navigation bar 1. The system discards all changes made. 2. The system displays the respective page for the clicked tab.

Exceptions:	Nil
Includes:	Nil
Notes and Issues:	Nil

Use Case ID:	12		
Use Case Name:	Import Preferences		
Created By:	Loong Kiat	Last Updated By:	Xi Quan
Date Created:	27/08/2025	Date Last Updated:	29/08/2025

Actor:	User
Description:	Users can choose to import preferences from their phone.
Preconditions:	1. A valid preference file is available
Postconditions:	1. Preferences in the file has been applied to the app
Flow of Events:	1. The user clicks on the import button. 2. The system prompts the user to select a file. 3. The user selects the desired file. 4. The system checks if the selected file is of the correct file type. 5. The system overwrites the current ranking of the categories with the rankings from the file. 6. The system overwrites the shortlist of neighbourhood with the shortlist from the file 7. The system displays a successful message.
Alternative Flows:	AF-2: The user cancel selection: 1. The system displays the message “You did not select any file” 2. The system exits the use case and returns to the settings page.

	AF-4: The file selected is of the wrong file type: 1. The system displays the message “You can only select .csv and .pdf files”. 2. The system returns to step 2.
Exceptions:	Nil
Includes:	Nil
Notes and Issues:	Nil

Use Case ID:	13		
Use Case Name:	Export Preferences		
Created By:	Loong Kiat	Last Updated By:	Loong Kiat
Date Created:	27/08/2025	Date Last Updated:	30/08/2025

Actor:	User
Description:	Users can choose to export all their preferences, such as categories ranking and shortlisted neighbourhoods, as a csv or pdf file.
Preconditions:	1. User has already ranked their preferences 2. User's phone has sufficient storage
Postconditions:	1. A csv or pdf file will be saved onto the users' phone
Flow of Events:	1. The user clicks on the export button. 2. The system creates a csv or pdf file with the user's preferences. 3. The system checks if the phone has sufficient storage. 4. The system saves the newly created csv or pdf file onto the phone. 5. The system displays a successful message.
Alternative Flows:	AF-3: If the phone does not have sufficient storage 1. The system displays the message "Phone has insufficient storage, please try again" 2. The system exits the use case and returns to the settings page.

Exceptions:	Nil
Includes:	Nil
Notes and Issues:	Nil

Use Case ID:	14		
Use Case Name:	Zoom Map		
Created By:	Xi Quan	Last Updated By:	Xi Quan
Date Created:	29/08/2025	Date Last Updated:	29/08/2025

Actor:	User
Description:	Users can zoom in and out of the Map View to choose neighbourhoods to explore.
Preconditions:	1. The map is displayed
Postconditions:	1. The map is displayed at the new zoom level.
Flow of Events:	1. User pinches the map 2. System adjusts the map with new zoom level 3. System displays updated map
Alternative Flows:	AF-S2: If zoom level is at maximum 1. The system prevents zooming in further. AF-S2: If zoom level is at minimum 1. The system prevents zooming out further.
Exceptions:	Nil
Includes:	Nil

Notes and Issues:	Nil

Blue: Boundary classes
Green: Control classes
Grey: Entity classes

FrontEnd Boundary Classes

BackEnd Boundary Classes

CompareView
+render(): void
+selectAreas(areaA: string, areaB: string): void
+fetchComparison(areaA: string, areaB: string, weightsId: string): void

DetailsView
+render(): void
+showCategoryBreakdown(areaId: string): void
+showFacilities(areaId: string): void
+showPriceTrend(areaId: string, months: int): void
+operation()

MapView
+render():void
+onPolygonClick(areaId: string):void
+fetchChoropleth(weightsId: string): void

WeightsView
+render(): void
+adjustWeights(profile: WeightsProfile): void
+fetchActiveWeights(): void
+saveWeights(profile: WeightsProfile): void

ShortlistView
+render(): void
+addToShortlist(areaId: string): void
+removeFromShortlist(areaId: string): void
+fetchShortlist(shortlistId: string): void

ExportView
+render(): void
+exportCSV(shortlistId: string): void
+exportPDF(shortlistId: string): void
+importCSV(file: bytes): void

SearchView
+render(): void
+onSearchSubmit(filters: SearchFilters): void
+fetchSearchResults(filters: SearchFilters, weightsId: string): void

PolygonDetailsView
+areaID: string
+activeTab:string
-breakdown: Map<Category, float>
-facilities : FacilitiesSummary
-priceTrend:List<PriceRecord>
-loading:boolean
+display(areaID:string):void
+onTabChange(tab:string):void
+onBack():void
+loadCategoryBreakdown(areaID:string) void
+loadFacilities(areaID:string):void
+loadPriceTrend(areaID:string,months:int):void
-setLoading(v:boolean):void
-renderChart():void
-renderFacillities():void
-renderTrend():void

CompareController
+compare(areaA: string, areaB: string, weightsId: string): Comparison

DetailsController
+getCategoryBreakdown(areaId: string, asOf: Date): Map<Category, float>
+getFacilities(areaId: string): FacilitiesSummary
+getPriceTrend(areaId: string, months: int): List<PriceRecord>

MapController
+getChoropleth(weightsId: string): List<NeighbourhoodScore>
+getAreaSummary(areaId: string, weightsId: string): Map<Category, float>, NeighbourhoodScore

WeightsController
+getActive(): WeightsProfile
+update(profile: WeightsProfile): WeightsProfile
+list(): List<WeightsProfile>

ShortlistController
+get(shortlist: string): Shortlist
+add(shortlist: string, areaId: string): Shortlist
+remove(shortlist: string, areaId: string): Shortlist

ExportController
+exportCSV(shortlistId: string): bytes
+exportPDF(shortlistId: string): bytes
+importCSV(data: bytes): Shortlist

SearchController
+search(filters: SearchFilters, weightsId: string): List<NeighbourhoodScore>

TrendService
+series(areaId: string, months: int): List<PriceRecord>

AuditService
+trace(areaId: string, category: Category): List<ScoreTrace>
+record(trace: ScoreTrace): void

RatingEngine
+computeCategoryScores(areaId: string, asOf: Date): Map<Category, float>
+aggregate(scores: Map<Category, float>, weights: WeightsProfile): float
+getFeatureVector(areaId: string, asOf: Date): FeatureVector

WeightsService
+current(): WeightsProfile
+update(profile: WeightsProfile): WeightsProfile
+list(): List<WeightsProfile>
+normalize(profile: WeightsProfile): WeightsProfile

ShortlistService
+get(id: string): Shortlist
+save(s: String): void

ExportService
+toCSV(areas: List<PlanningArea>, weights: WeightProfile): bytes
+toPDF(areas: List<PlanningArea>, weights: WeightProfile): bytes
+fromCSV(data: bytes): Shortlist

SearchService
+rankAreas(filters: SearchFilters, weights: WeightProfile): List<NeighbourhoodScore>

PrecomputeService
+refreshAll(asOf: Date): int
+refreshArea(areaId: string, asOf: Date): FeatureVector

Normalizer
+minMax(x: float, minV: float, maxV: float): float
+zScore(x: float, mean: float, std: float): float

AccessibilityCalculator
+score(areaId: string, asOf: Date): float

AmenitiesCalculator
+score(areaId: string, asOf: Date): float

EnvironmentCalculator
+score(areaId: string, asOf: Date): float

CommunityCalculator
+score(areaId: string, asOf: Date): float

AffordabilityCalculator
+score(areaId: string, asOf: Date): float

<<Interface>> IPriceRepo
+series(areaId: string, months: int): List<PriceRecord>
+latestMedian(areaId: string): int

<<Interface>> ITraceRepo
+save(t: ScoreTrace): void
+byAreaCategory(areaId: string, category: Category): List<ScoreTrace>

<<Interface>>I WeightsRepo
+getActive(): WeightsProfile
+save(profile: WeightsProfile): void
+list(): List<WeightsProfile>

<<Interface>> IFeatureRepo
+get(areaId: string): FeatureVector
+save(f: FeatureVector): void

<<Interface>>I ShortlistRepo
+get(id: string): Shortlist
+save(s: Shortlist): void

<<Interface>>IPlann ingAreaRepo
+get(id: string): PlanningArea
+list(): List<PlanningArea>

<<Interface>>I AmenityRepo
+byArea(areaId: string): List<Amenity>

<<Interface>> ITransitRepo
+mrtByArea(areaId: string): List<MRTStation>
+busByArea(areaId: string): List<BusStop>

<<Interface>> ICarparkRepo
+byArea(areaId: string): List<Carpark>

<<Interface>> IGreenRepo
+byArea(areaId: string): List<GreenSpace>

<<Interface>>ICo mmunityRepo
+byArea(areaId: string): List<CommunityCentre>

<<Interface>> IScoreRepo
+save(score: NeighbourhoodScore): void
+latest(areaId: string, weightsId: String): NeighbourhoodScore

TransitStop
+id: string
+kind: TransitKind
+location: GeoPoint

BusStop

MRTStation

FacilitiesSummary
+schools: int
+sports: int
+hawkers: int
+healthcare: int
+greenSpaces: int
+carparks: int

FeatureVector
+areaId: string
+affordability: float
+accessibility: float
+amenities: float
+environment: float
+community: float
+asOf: Date

CategoryScore
+areaId: string
+category: Category
+value: float
+methodVersion: string
+computedAt: Date

ScoreTrace
+areaId: string
+category: Category
+inputsJSON: string
+sourceTimestamps: Map<string, Date>
+calcVersion: string
+computedAt: Date

CommunityCentre
+id: string
+location: GeoPoint

WeightsProfile
+id: string
+label: string
+wAff: float
+wAcc: float
+wAmen: float
+wEnv: float
+wCom: float
+version: int
+isValid(): bool
+normalized(): WeightsProfile

NeighbourhoodScore
+areaId: string
+total: float
+weightsprofileId: string
+computedAt: Date

Shortlist
+id: string
+userId: string
+areaIds: List<string>
+savedWeightsProfileId: string
+createdAt: Date
+add(areaId: string): void
+remove(areaId: string): void

PlanningArea
+id: string
+name: string
+geometryId: string
+bbox: BoundingBox
+contains(p: GeoPoint): bool

GreenSpace
+id: string
+type: GreenType
+areaSqm: float
+location: GeoPoint

Carpark
+id: string
+location: GeoPoint
+lots: int

Amenity
+id: string
+type: AmenityType
+location: GeoPoint

School

Healthcare

HawkerCentre

SportsFacility

PriceRecord
+areaId: string
+month: Date
+medianResale: int
+p25: int
+p75: int
+volume: int

Comparison
+areaA: string
+areaB: string
+scoresA: Map<Category, float>
+scoresB: Map<Category, float>
+trendA: List<PriceRecord>
+trendB: List<PriceRecord>

SearchFilters
+maxBudget: int
+mustHaveFacilities: Set<AmenityType>
+minAccessibility: float
+page: int
+pageSize: int

BoundingBox
+min: GeoPoint
+max: GeoPoint

GeoPoint
+lat: float
+lon: float

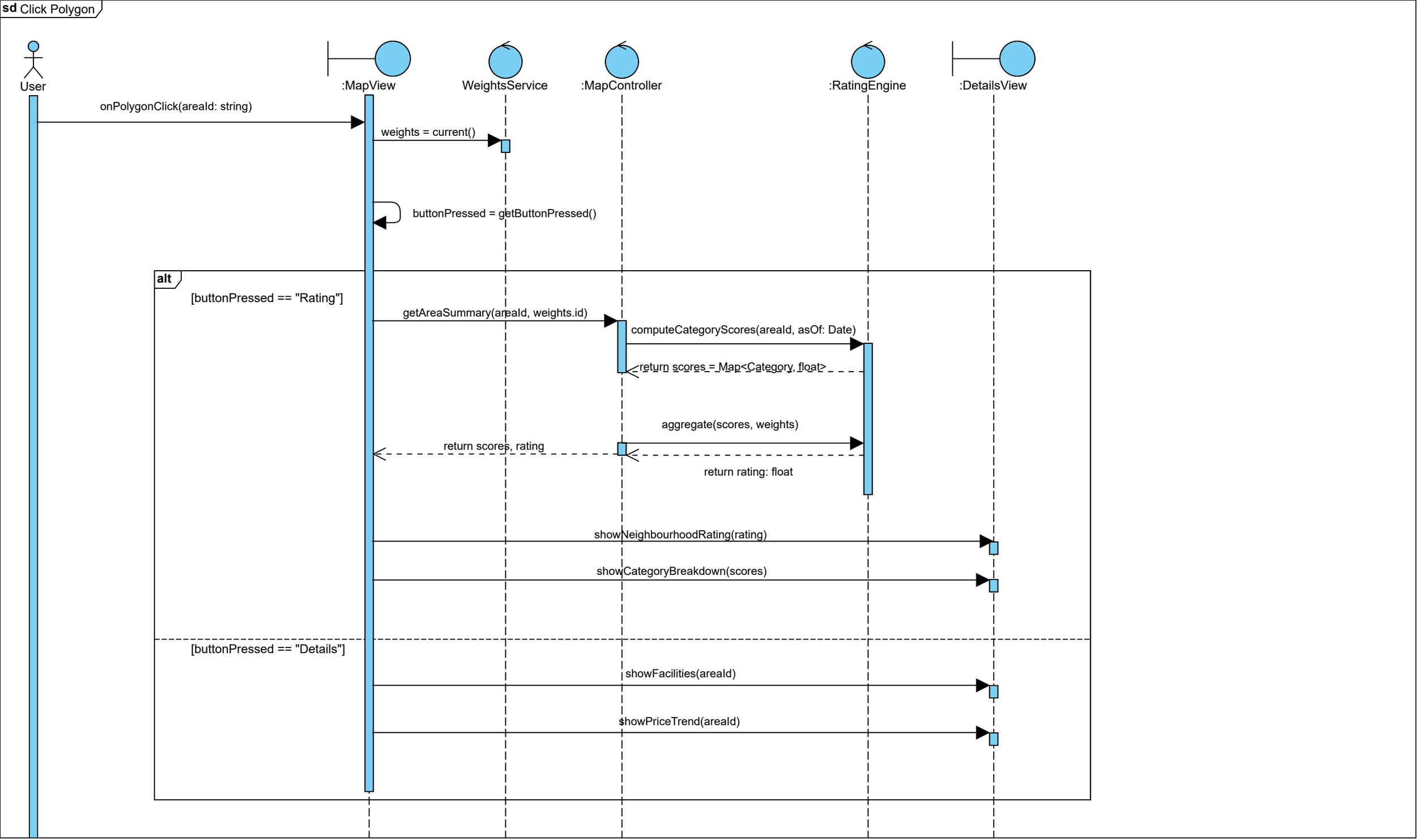
<<enumeration>> TransitKind
MRT
Bus

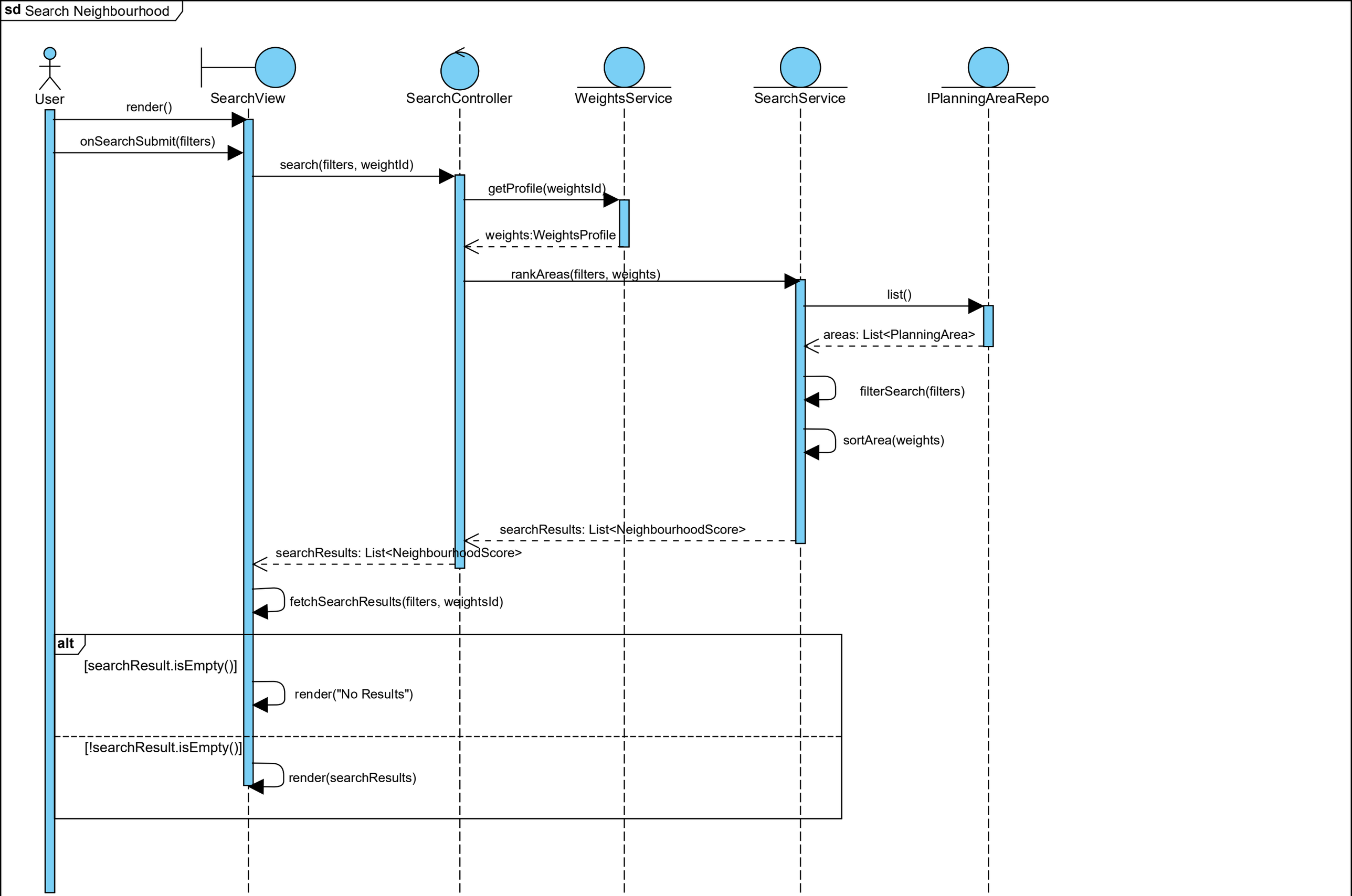
<<enumeration>> AmenityType
School
SportsSG
Hawker
Healthcare
Supermarket

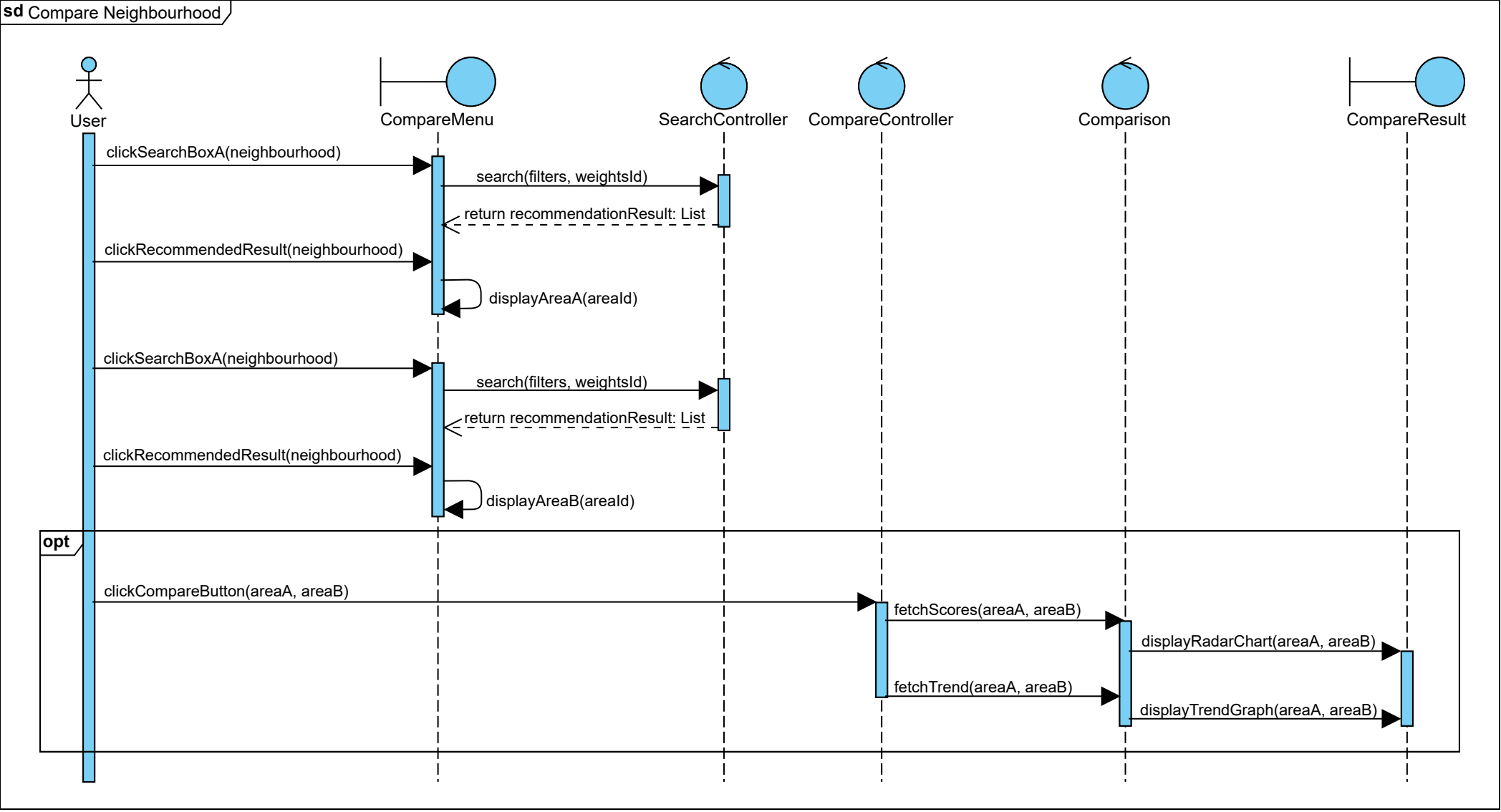
<<enumeration>> Category
Affordability
Accessibility
Amenities
Environment
Community

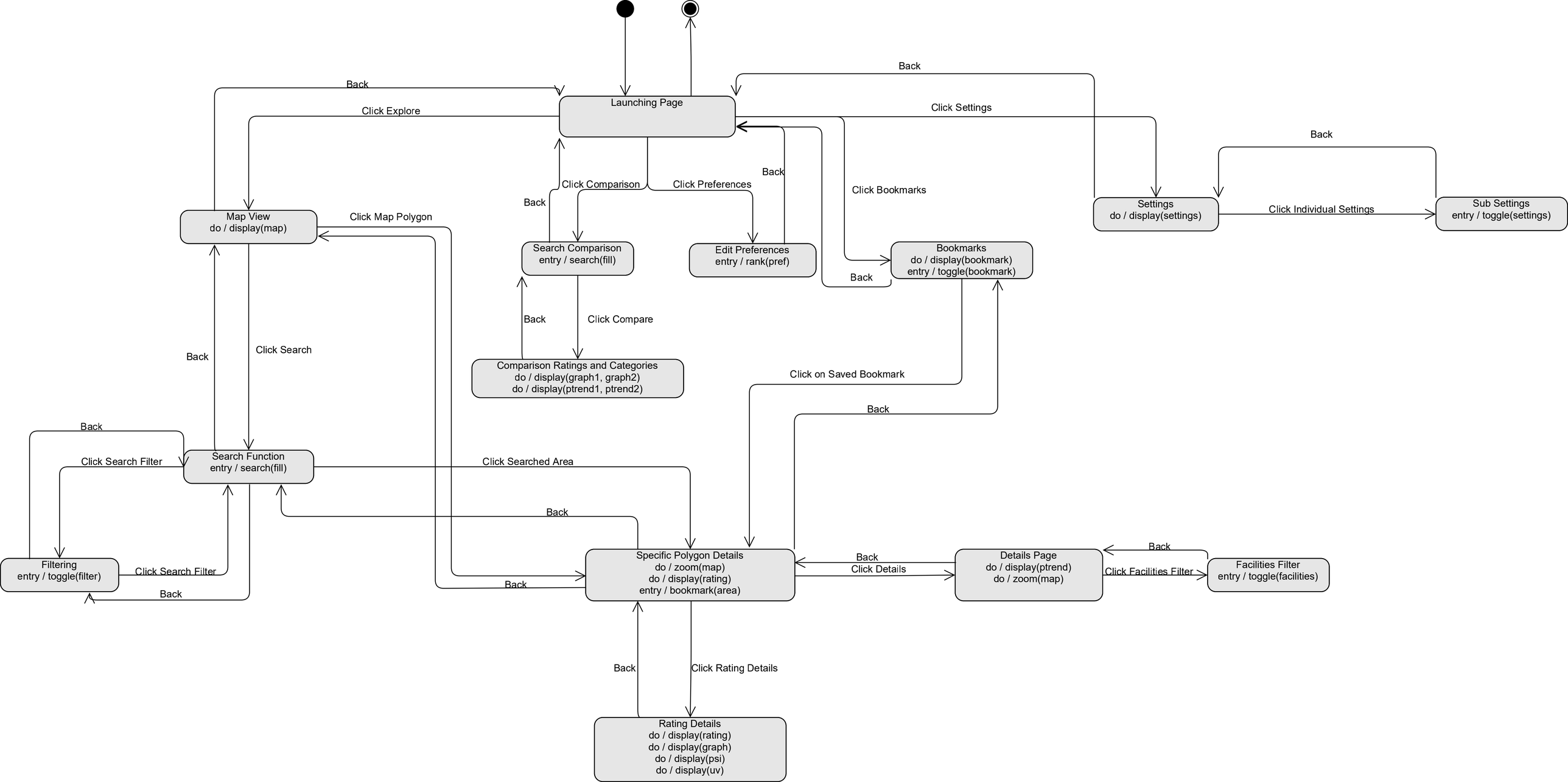
<<enumeration>> GreenType
Park
ParkConnector
NatureReserve

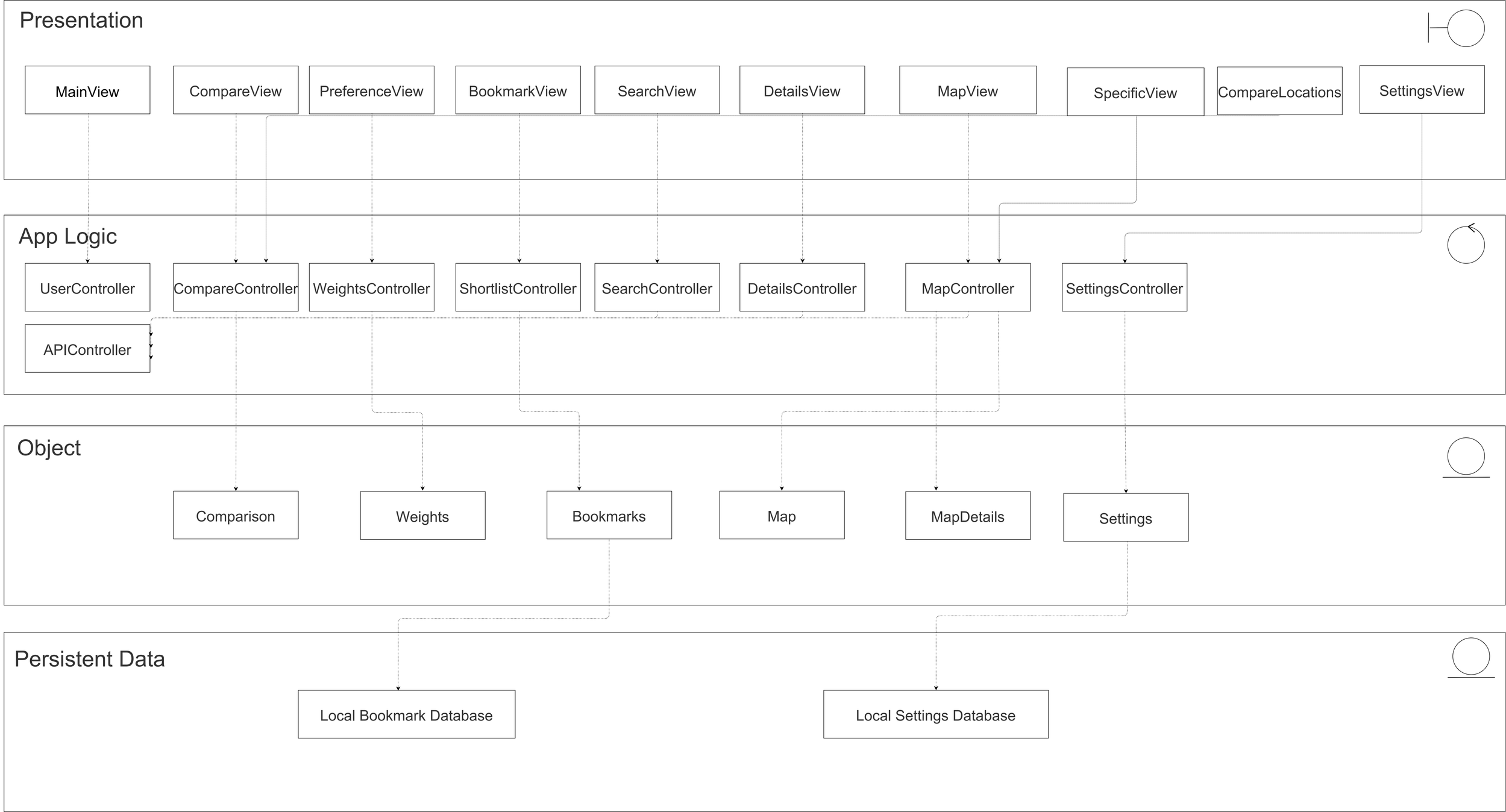
has
latest
derives
incorporates
produces
contains


sd Click Polygon
User
:MapView
WeightsService
:MapController
:RatingEngine
:DetailsView
onPolygonClick(areaId: string)
weights = current()
buttonPressed = getButtonPressed()
alt
[buttonPressed == "Rating"]
getAreaSummary(areaId, weights.id)
computeCategoryScores(areaId, asOf: Date)
return scores = Map<Category, float>
aggregate(scores, weights)
return scores, rating
return rating: float
showNeighbourhoodRating(rating)
showCategoryBreakdown(scores)
[buttonPressed == "Details"]
showFacilities(areaId)
showPriceTrend(areaId)


sd Search Neighbourhood
User
SearchView
SearchController
WeightsService
SearchService
IPlanningAreaRepo
render()
onSearchSubmit(filters)
search(filters, weightId)
getProfile(weightsId)
weights:WeightsProfile
rankAreas(filters, weights)
list()
areas: List<PlanningArea>
filterSearch(filters)
sortArea(weights)
searchResults: List<NeighbourhoodScore>
searchResults: List<NeighbourhoodScore>
fetchSearchResults(filters, weightsId)
alt
[searchResult.isEmpty()]
render("No Results")
[!searchResult.isEmpty()]
render(searchResults)


sd Compare Neighbourhood
User
CompareMenu
SearchController
CompareController
Comparison
CompareResult
clickSearchBoxA(neighbourhood)
search(filters, weightsId)
return recommendationResult: List
clickRecommendedResult(neighbourhood)
displayAreaA(areaId)
clickSearchBoxA(neighbourhood)
search(filters, weightsId)
return recommendationResult: List
clickRecommendedResult(neighbourhood)
displayAreaB(areaId)
opt
clickCompareButton(areaA, areaB)
fetchScores(areaA, areaB)
displayRadarChart(areaA, areaB)
fetchTrend(areaA, areaB)
displayTrendGraph(areaA, areaB)


Launching Page
Back
Click Explore
Back
Click Settings
Back
Map View
do / display(map)
Click Map Polygon
Click Comparison
Click Preferences
Back
Back
Click Bookmarks
Settings
do / display(settings)
Click Individual Settings
Sub Settings
entry / toggle(settings)
Search Comparison
entry / search(fill)
Edit Preferences
entry / rank(pref)
Bookmarks
do / display(bookmark)
entry / toggle(bookmark)
Back
Back
Click Compare
Back
Click Search
Comparison Ratings and Categories
do / display(graph1, graph2)
do / display(ptrend1, ptrend2)
Click on Saved Bookmark
Back
Back
Search Function
entry / search(fill)
Click Search Filter
Click Searched Area
Back
Filtering
entry / toggle(filter)
Click Search Filter
Back
Back
Specific Polygon Details
do / zoom(map)
do / display(rating)
entry / bookmark(area)
Back
Click Details
Details Page
do / display(ptrend)
do / zoom(map)
Back
Click Facilities Filter
Facilities Filter
entry / toggle(facilities)
Back
Click Rating Details
Rating Details
do / display(rating)
do / display(graph)
do / display(psi)
do / display(uv)


Presentation
MainView
CompareView
PreferenceView
BookmarkView
SearchView
DetailsView
MapView
SpecificView
CompareLocations
SettingsView
App Logic
UserController
CompareController
WeightsController
ShortlistController
SearchController
DetailsController
MapController
SettingsController
APIController
Object
Comparison
Weights
Bookmarks
Map
MapDetails
Settings
Persistent Data
Local Bookmark Database
Local Settings Database