

# CBR Project Report

---

COMPSCI 760 — ASSIGNMENT 1

Ruoyi (Zoe) Cai  
6007959 | RCAI861

## Table of Contents

<b>Introduction .....</b>	<b>3</b>
<b>What is CBR? .....</b>	<b>3</b>
CBR Logic .....	3
<b>CBR Logic.....</b>	<b>4</b>
<b>Holiday Type .....</b>	<b>5</b>
<b>Price .....</b>	<b>7</b>
<b>Number of Persons.....</b>	<b>8</b>
<b>Transportation .....</b>	<b>9</b>
<b>Duration.....</b>	<b>10</b>
<b>Season .....</b>	<b>11</b>
<b>Accommodation .....</b>	<b>12</b>
<b>Design &amp; Implementation.....</b>	<b>13</b>
<b>Class Structure .....</b>	<b>13</b>
<b>Graphical User Interface .....</b>	<b>15</b>
<b>User Manual.....</b>	<b>16</b>
<b>Works Cited.....</b>	<b>17</b>

## Introduction

This report describes case-based reasoning and the travel CBR application developed using the methodology. The travel CBR application was built using the travel case base data, and the k nearest neighbour algorithm as the CBR logic.

This report explains the attributes used in the travel CBR application and the similarity calculation methods for each attribute in section 'CBR Logic', and the design and implementation details of the application in the 'Design and Implementation' section.

### What is CBR?

Case-based reasoning (CBR) is a methodology [1] for solving problems based on existing sets of problems and solutions. A case is an experience of a solved problem, and a case base is a collection of cases. Case-based reasoning therefore means drawing conclusions based on the case base. However, a distinguishing characteristic of CBR is its approximate reasoning by adaptation, as well as its reuse of new experiences. [2]

A conceptual description of the CBR cycle comprises of four activities: [1]

1. Retrieve similar cases to the problem description
2. Reuse a solution suggested by a similar case
3. Revise or adapt the solution to fit the new problem, if necessary
4. Retain the new solution once confirmed or validated

### CBR Logic

The most widely used technology in CBR is the nearest neighbour algorithm. [3] In a nearest neighbour algorithm, the similarity of the problem case to a case in the case base for each case attribute is calculated, which may be weighted by a factor. The similarities of all attributes are then summed to provide a measure of the similarity of the existing case to the target case. This can be represented by the equation:

$$Similarity(T, S) = \sum_{i=1}^n f(T_i, S_i) \times w_i$$

where T is the target case; S the existing case; n the number of attributes in each case; i an individual attribute from 1 to n; f a similarity function for attribute i in cases T and S; and w the importance weighting of attribute i. [1]

## CBR Logic

From the sample data taken from the travel case-base Excel file, we can see 11 attributes for each case.

defcase	1	
	objects	
	case	Journey1
	JourneyCode:	1
	HolidayType:	Bathing,
	Price:	2498
	NumberOfPersons:	2
	Region:	Egypt,
	Transportation:	Plane,
	Duration:	14
	Season:	April,
	Accommodation:	TwoStars,
	Hotel:	Hotel White House, Egypt.

Of the 11 attributes, nine attributes could potentially be used for case retrieval, while two — case and JourneyCode — are useful for identifying cases.

Of the nine potentially useful attributes, both Region and Hotel require more information to be used for comparison. Region requires information such as geographic coordinates (latitude and longitude) as well as possibly type of region, such as mainland or island, or features of the region like rivers, or even languages spoken. Hotel requires information such as services offered, type (boutique, luxury), or ratings. As these information are not provided by the Excel file, and if retrieved from elsewhere it is not known if they will be useful to the retrieval part of CBR, both Region and Hotel are ignored in this application.

Therefore, only HolidayType, Price, NumberOfPersons, Transportation, Duration, Season, and Accommodation are used in this application.

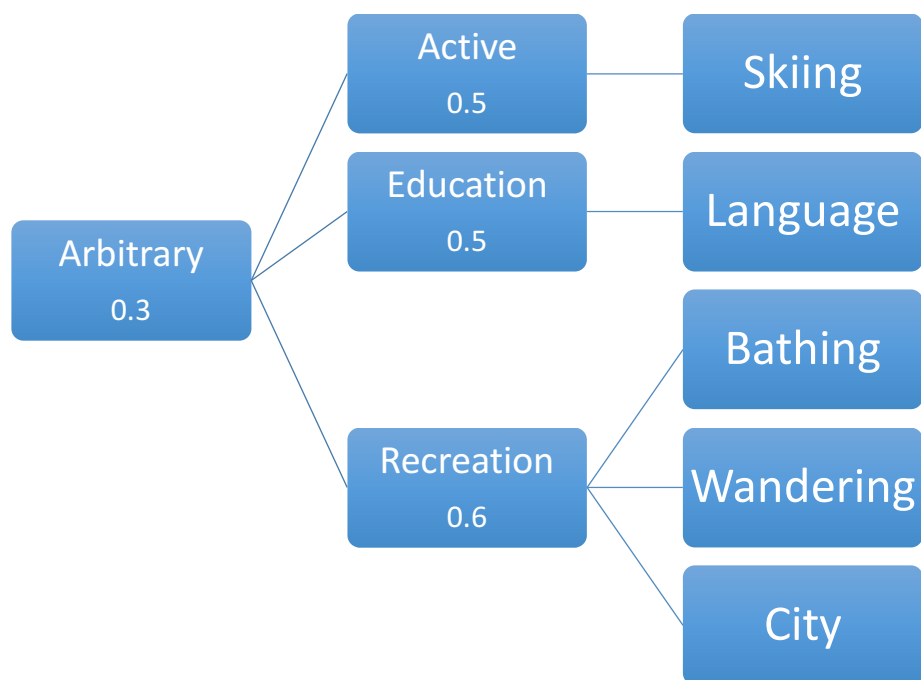
## Holiday Type

There are nine different holiday types in the case base.

- Active
- Bathing
- City
- Education
- Language
- Recreation
- Skiing
- Wandering

To represent the similarities between the different holiday types, a few different approaches could be used, including hierarchical tree or decision tables.

Hierarchical tree was considered, however, not all holiday types are mutually exclusive. For example, a City holiday, though it could be classified under Recreation, could be combined with a Language holiday, if the traveler were to study language in a city. However, it would be difficult to combine a City holiday and a Skiing holiday, as most ski resorts and mountains are not located in cities. Therefore, the similarity score between City and Language should be higher than the score between City and Skiing. But if a hierarchical tree — such as the one presented below — were used, then both combinations would have the same similarity score.



*Figure 1 Hierarchical Tree for holiday types, where the similarity score of the lowest common ancestor of the new case holiday type and the existing case holiday type is used*

Therefore, a symmetrical decision table is used instead to represent similarity between the different types.

Table 1 Symmetrical Decision Table for Holiday Type

	Bathing	Active	Education	City	Recreation	Wandering	Language	Skiing
Bathing	1	0.1	0.1	0.3	0.7	0.65	0.3	0.2
Active	0.1	1	0.2	0.2	0.5	0.75	0.45	1
Education	0.1	0.2	1	0.8	0.5	0.5	0.9	0.45
City	0.3	0.2	0.8	1	0.7	0.9	0.75	0.15
Recreation	0.7	0.5	0.5	0.7	1	0.8	0.4	0.9
Wandering	0.65	0.75	0.5	0.9	0.8	1	0.65	0.45
Language	0.3	0.45	0.9	0.75	0.4	0.65	1	0.3
Skiing	0.2	1	0.45	0.15	0.9	0.45	0.3	1

The similarity scores were made up based on the similarity and the compatibility of the holiday types.

The weight given to the holiday type attribute is 0.2, which is the median of all feature weights.

## Price

Price similarity is calculated using a cosine function when the price of the existing case is within the price threshold of the new case. If the existing case were cheaper than the new case, then a similarity of 1.0 would be returned. If the price of the existing case were above the price threshold of the new case, then 0.0 would be returned.

The price threshold is 125% of the price of the new case. This is used as a \$100 increase in price matters a lot more to a holiday that was meant to cost \$150 than a \$2000 holiday.

The formula for calculating similarity in price within the threshold is:

$$similarity = \frac{1}{2} \cos \left( \frac{existing\ price - new\ price}{0.25 * new\ price * \pi} \right) + 0.5$$

This can be represented by a graph:

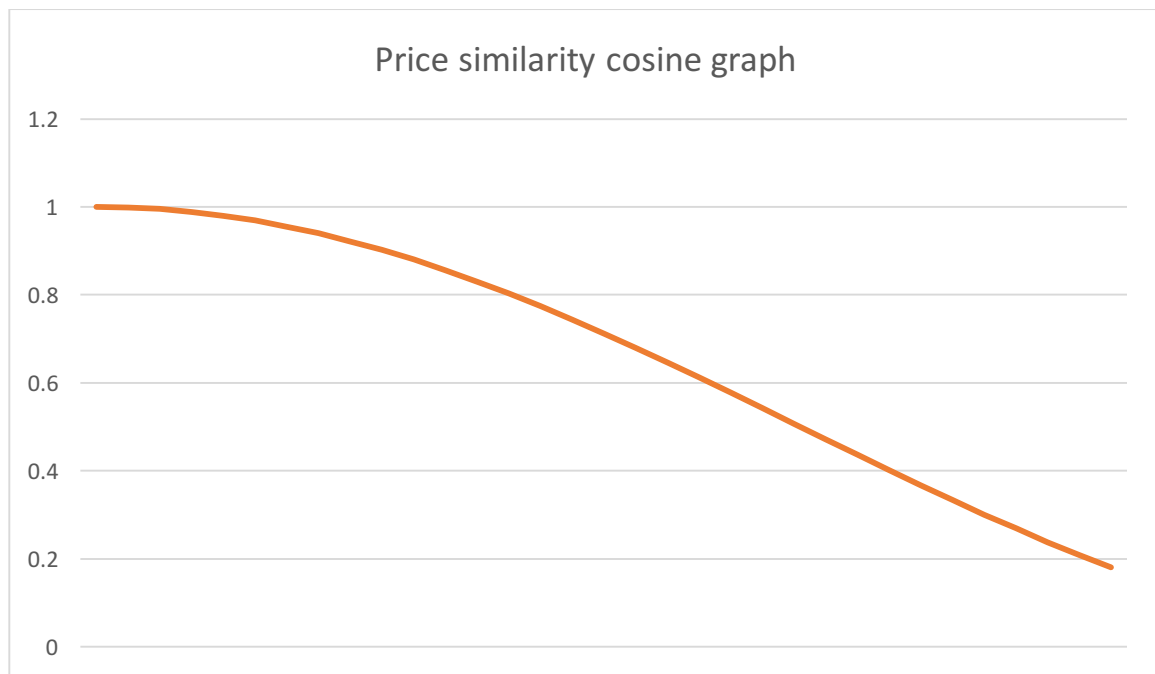


Figure 2 Price similarity cosine graph, where the x axis intercept point represents the maximum price threshold

The weight given to the price attribute is 0.3, which is slightly higher than the median weight of 0.2. This is because price is usually a very important factor, and when given a budget, the price should definitely not exceed the upper threshold.

## Number of Persons

Number of persons similarity is calculated using an asymmetric function. If the number of persons in the existing case were less than the number of person in the new case, 0.0 is returned, as people are not likely to cut someone out of their travel plans. If the number of persons in the existing case were more than the number of person in the new case, then a straight-line function is used. The threshold that the number of persons in an existing case can increase by is 130% of the number of persons in the new case. This is used as a trip with a higher number of persons would not be affected as much if the number of persons for the trip were to change compared to a trip with a lower number of persons.

The formula for calculating the similarity for number of persons, if within the threshold, is:

$$similarity = 1.0 - \frac{existing\ case\ persons - new\ case\ persons}{new\ case\ persons * 1.3}$$

The weight given to the number of persons attribute is 0.2, which is the median weight.



## Transportation

There are four different transportation types in the case base:

- Car
- Coach
- Plane
- Train

An asymmetric decision table is used to represent similarities between the types of transportation, as there could be different similarity scores between two transportation types depending on the transportation type in the existing case.

*Table 2 Transportation similarity stored in asymmetric decision table, where each row represents a transportation type in the existing case*

	Plane	Car	Train	Coach
Plane	1	0	0	0
Car	0	1	0.8	0.5
Train	0	0.3	1	0.7
Coach	0	0.5	0.7	1

The transportation type attribute is given a weight of 0.1, slightly lower than the median weight of 0.2, as transportation is usually quite flexible compared to the other attributes. For example, someone who doesn't own a car could rent a car if other transportation types are not available.

## Duration

Duration is calculated using a symmetrical straight-line function, if the duration in an existing case is within the threshold. A threshold of 30% longer or shorter than the new case is used, as the longer a trip is, the less effect modifying a certain number of days has. If the duration in the existing case is outside of the threshold, either too long or too short, then 0.0 is returned.

If the duration of the existing case were longer than that of the new case, but still within the 130% of the duration of the new case threshold, then the formula used is:

$$similarity = \frac{1}{\frac{new\ case\ duration * 0.3}{\times (existing\ case\ duration - new\ case\ duration)} + 1}$$

If the duration of the existing case were shorter than that of the new case, but still within the 70% of the duration of the new case threshold, then the formula used is:

$$similarity = \frac{-1}{\frac{new\ case\ duration * 0.3}{\times (existing\ case\ duration - new\ case\ duration)} + 1}$$

The duration attribute is given a weight of 0.2, the median weight.

## Season

Season is calculated by first mapping the months to integers. January is converted to 0, February to 1, and so on. The season in the existing case and the season in the new case are both converted, and then compared. If the values are the same, i.e., the cases both have the same month, then 1.0 is returned as the similarity. If the values are different, then the difference in seasons is calculated using the formula:

$$difference\ in\ seasons = \left| \frac{new\ case\ month\ \% 12}{3} - \frac{existing\ case\ month\ \% 12}{3} \right|$$

This gives us an integer representing the number of seasons the months differ by. For example, Spring and Summer have a difference of 1.

If the new case month and the existing case month are in the same season, then 0.8 is returned as the similarity.

If there is a seasonal difference of 1 or 3, meaning that the season is either the season before or after the desired season, then 0.4 is returned as the similarity.

If the existing case has the opposite season, then 0.0 is returned as the similarity.

The season attribute is given a weight of 0.2, the median weight.

## Accommodation

There are six different accommodation types in the case base:

- HolidayFlat
- OneStar
- TwoStars
- ThreeStars
- FourStars
- FiveStars

It is assumed that HolidayFlat has a lower rating than one star.

Each accommodation type is mapped to an integer. HolidayFlat is mapped to 0, OneStar is mapped to 1, TwoStars to 2, and so on.

An asymmetric straight-line function is used to calculate the similarity score. If the existing case accommodation were rated higher than that of the new case, then 1.0 is returned as the similarity score. However, if the existing case accommodation were rated lower than that of the new case, then the following formula is used:

$$similarity = -\frac{1}{3}(difference\ in\ ratings) + 1$$

This means any existing case with an accommodation more than 2 ratings lower than the new case accommodation would return a similarity score of 0.0.

The accommodation type attribute is given a weight of 0.2, the median weight.

## Design & Implementation

The application was developed using Java, with the graphical user interface implemented using Swing.

### Class Structure

There are three packages containing a total of seven classes.

In package `gui`, the class `MainRunner` is responsible for setting up the graphical user interface and initialising k-nn calculator.

In package `retrieval`, the class `Cases` is responsible for reading the travel case base CSV file (converted from the Excel file format) and transforming the content into objects of the class `TravelCase`. All cases are stored in an arraylist. The class `TravelCase` represents a case in the case base, and contains all attributes such as “caseName” and “accommodation”.

In package `similarityMetrics`, the class `kNearestNeighbour` is responsible for calculating the similarity scores of each existing case to the new case, and ordering the cases from the most similar to the least similar using a priority queue.

The class `DefaultMetrics` is used to store all the attribute local similarity scores and calculation methods. It contains all the thresholds for different attributes, and contains values for populating decision tables.

The class `DecisionTable` is used to store the similarity values of all attributes that need a decision table for similarity calculation. It contains methods for updating, retrieving, and displaying values in the decision table.

The class `SymmetricalDecisionTable` is a subclass of class `DecisionTable`, and is used to represent a symmetrical decision table. It overrides the two methods in class `DecisionTable` for updating values.

A UML diagram is presented below to illustrate the class structure.

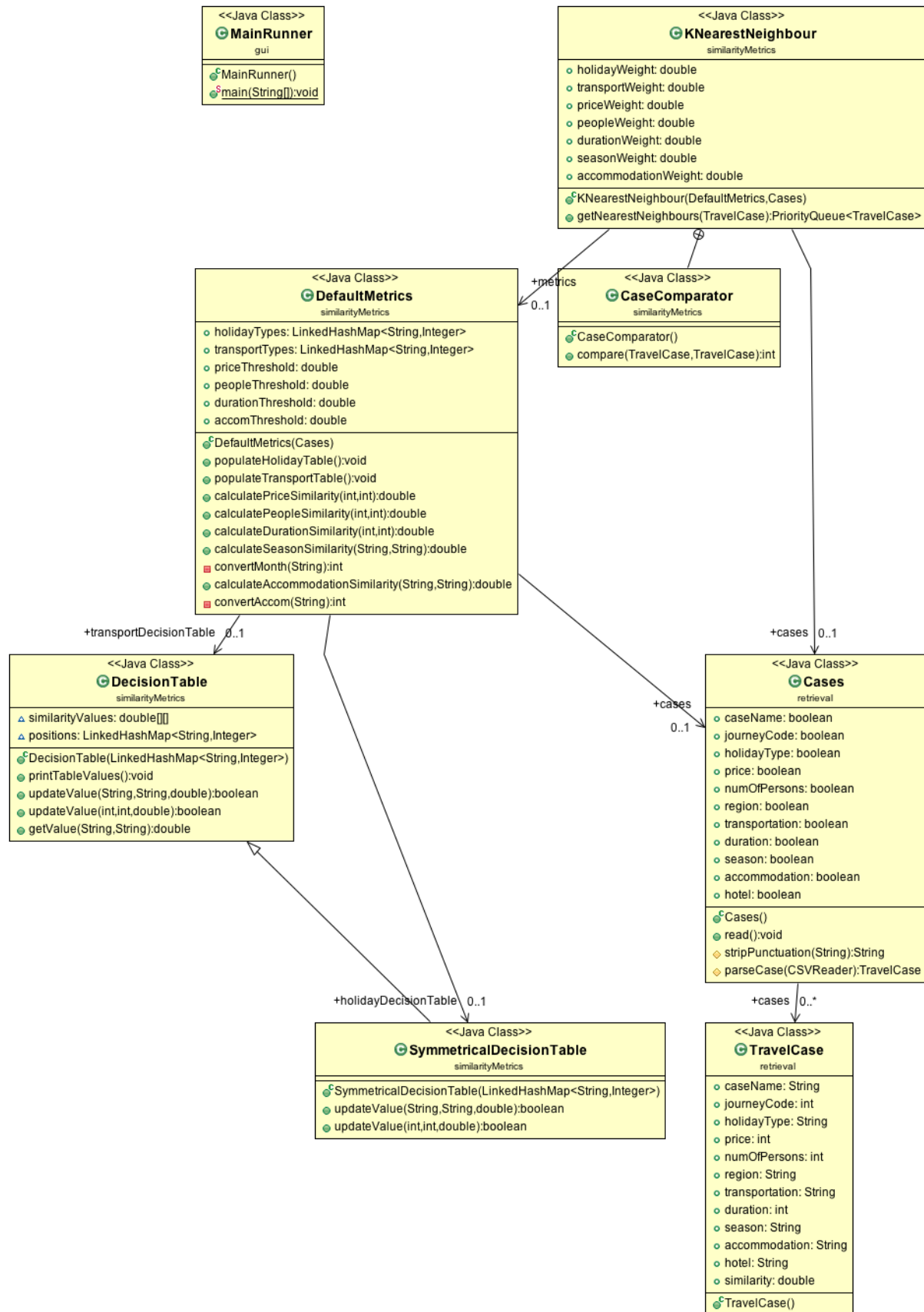
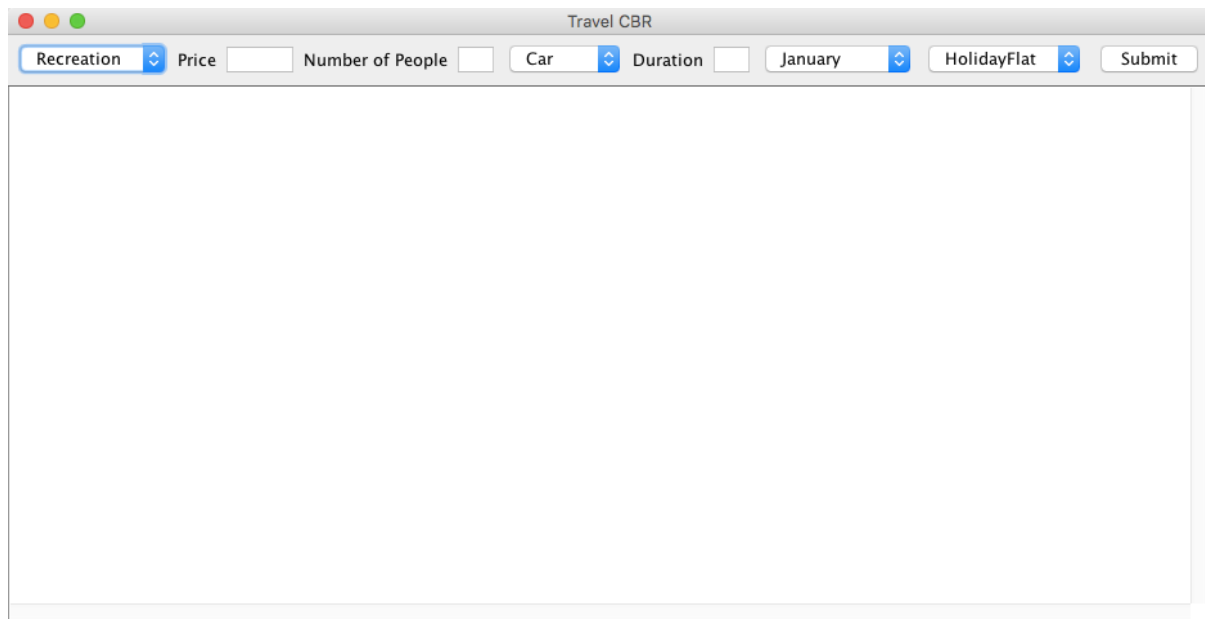


Figure 3 UML Diagram of the application

## Graphical User Interface

The graphical user interface is implemented using Swing.

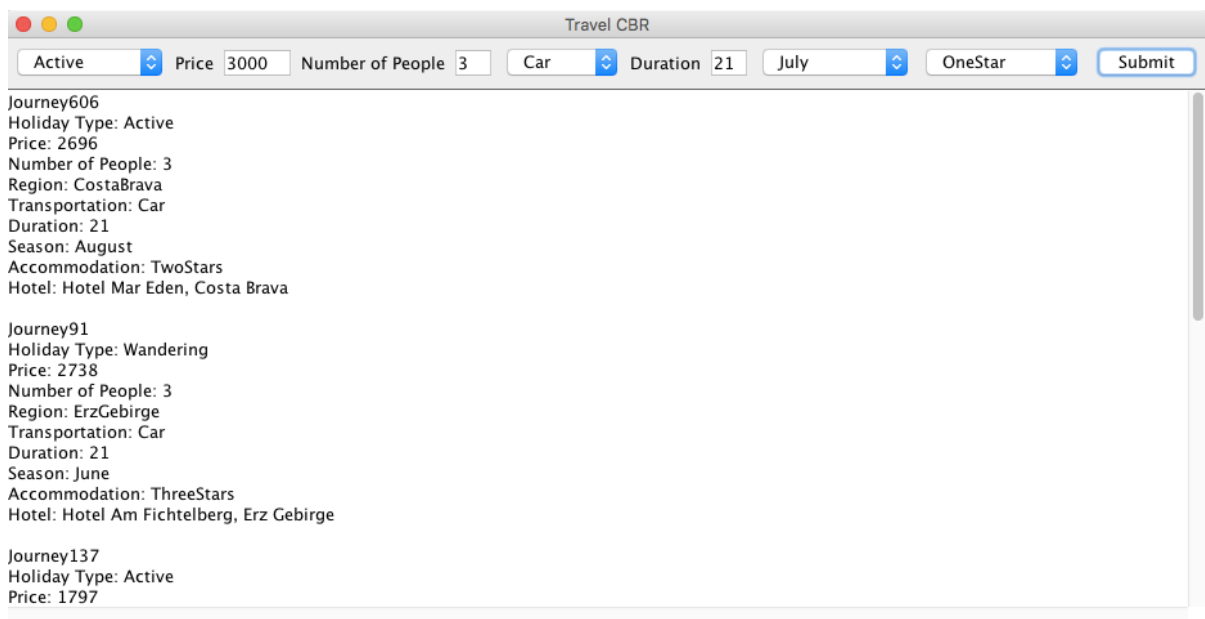
Input fields for entering the attributes for the new case are at the top. Where values are limited, such as HolidayType, Transportation, Season, and Accommodation, a dropdown box is used. This allows the user to see the available options, and prevents them from entering incorrect values and causing errors.



The screenshot shows a Java Swing window titled "Travel CBR". At the top, there is a horizontal bar containing several input fields and a "Submit" button. The fields are: "Recreation" (a dropdown menu with a blue arrow icon), "Price" (a text input field), "Number of People" (a text input field), "Car" (a dropdown menu with a blue arrow icon), "Duration" (a text input field), "January" (a dropdown menu with a blue arrow icon), and "HolidayFlat" (a dropdown menu with a blue arrow icon). Below this bar is a large, empty rectangular area intended for displaying the results of the search.

*Figure 4 Graphical User Interface*

Once all attribute values are given and the submit button is clicked, a list of the 10 most similar cases are retrieved and displayed in the main text area.



The screenshot shows the same "Travel CBR" window, but now the input fields are populated with values: "Active" (dropdown), "Price" (3000), "Number of People" (3), "Car" (dropdown), "Duration" (21), "July" (dropdown), and "OneStar" (dropdown). The "Submit" button is highlighted. The main text area now displays a list of 10 retrieved cases, each with its ID and attributes. The first three cases are visible:

- Journey606  
Holiday Type: Active  
Price: 2696  
Number of People: 3  
Region: CostaBrava  
Transportation: Car  
Duration: 21  
Season: August  
Accommodation: TwoStars  
Hotel: Hotel Mar Eden, Costa Brava
- Journey91  
Holiday Type: Wandering  
Price: 2738  
Number of People: 3  
Region: ErzGebirge  
Transportation: Car  
Duration: 21  
Season: June  
Accommodation: ThreeStars  
Hotel: Hotel Am Fichtelberg, Erz Gebirge
- Journey137  
Holiday Type: Active  
Price: 1797

*Figure 5 Top 10 retrieved cases most similar to input case*

## User Manual

No installation or setup is required. The application can be used by opening the rcia861.jar file.

The fields Price, Number of People, and Duration require numerical inputs.

When all inputs are correctly given, click on the Submit button to retrieve the top 10 most similar cases.



## Works Cited

- [1] I. Watson, "CBR is a methodology not a technology," *The Knowledge Based Systems Journal*, vol. 12, no. 5-6, pp. 303-8, Oct 1999.
- [2] M. M. Richter and R. O. Weber, Case-Based Reasoning, Berlin: Springer Berlin Heidelberg, 2013.
- [3] I. Watson, Applying Case-based Reasoning: Techniques for Enterprise Systems, San Francisco, CA: Morgan Kaufmann, 1997.