SkiSmart-Ski Resort API Documentation

Overview
The SkiSmart-Ski Resort API is a comprehensive resource that consolidates data from multiple sources, including Google APIs (Places API, Place Details API, Geolocation API) and Visual Crossing Weather API. Our API processes this data, which involves calculating distances using the Haversine function and compiling weather information. This API is designed for ski enthusiasts, offering detailed information about ski resorts and weather conditions based on specific postal codes and dates within a hardcoded search radius of 150km.

Authentication
No authentication is required for users to access the endpoints. Internally, our system authenticates with Google APIs and Visual Crossing Weather API, processes the data, and responds to user queries.

Rate Limit
Currently, there is no rate limit in place for accessing our endpoints.

EndPoints
1. Ski Resorts by City
Endpoint: /skicities

Description:
Retrieves a list of ski resorts based on one of the six predefined cities. The results are based on the hardcoded radius of 150km from the city center, calculates the distance from there to the ski resort and provides weather information for each resort.

Supported Cities:
Montreal (H3G 1M8)
Toronto (M5N 2H9)
Calgary (T2G 0P3)
Ottawa (K2J 5P8)
Edmonton (T6G 0J9)
Vancouver (V6G 1Z4).

Query Parameters:
city: Name of the city (must start with a capital letter).
date: Desired date for the ski trip (format: DD-MM-YYYY).

Attributes Returned:

name: Name of the ski resort.
address: Physical address of the resort.
website: Website URL of the resort.
photo: Photo URL of the resort.
rating: User rating of the resort.
mapUrl: Google Maps URL of the resort.
postalCode: Postal code of the resort.
date: Date for which the information is provided.
distance: Distance from the city to the resort.
weatherData: Contains weather-related attributes such as temperature, forecast, wind speed, and snow depth.

Sample Calls:

JavaScript

```
fetch('http://3.145.39.202/skicities?city=Vancouver&date=05-11-2023')
  .then(response => response.json())
  .then(data => console.log(data));
```

Browser

```
http://3.145.39.202/skicities?city=Vancouver&date=05-11-2023
```

Terminal

```
curl http://3.145.39.202/skicities?city=Vancouver&date=05-11-2023
```

Sample JSON Response:


```
[
  {
    "cityData": {
      "name": "The Destination Slope and Surf Outfitters",
      "address": "267 Pemberton Ave, North Vancouver, BC V7P 2R4, Canada",
      "website": "https://www.thedestination.ca/",
      "photo":
"https://lh3.googleusercontent.com/places/ANXAkqHcolrqit1nchKsdm0QOiK1jy6J3FG2nD_M
kMJSnUxcAG8XozL5SqCFWOijVPAtroRZyUWA6T5SELr7vtW4rVVh8hvojYcQ5Js=s1600-w
800",
      "rating": 4.4,
      "mapUrl":
"https://www.google.com/maps/embed/v1/view?key=AIzaSyBq_hYo4JWsz1s6HglTs1doKeGdA
hHlt3U&center=49.3198385%2C-123.1079751&zoom=15",
      "postalCode": "V7P 2R4",
      "date": "2023-11-05",
      "distance": 5.9
    },
```

```
   "weatherData": {
     "temp": 10.6,
     "wForecast": "Rain, Partially cloudy",
     "wind": 15.8,
     "sDepth": 0
   }
  }
]
```


Error Codes:
400 Bad Request: City name and date are required.
400 Bad Request: City not found.
500 Internal Server Error: Any server-side error.

2. Ski Resorts by Distance
Endpoint: /skiresortsbydistance

Description:
Provides a list of ski resorts sorted by distance, from the closest to the furthest from the user's postal code. Users need to specify the limit to the number of resorts returned. For example, limit=3, will return the 3 closest ski resorts to the user's postal code.

Query Parameters:
postalCode: Postal code of the user's location.
date: Desired date for the ski trip (format: DD-MM-YYYY).
limit: Number of resorts to return.

Attributes Returned:
name: Name of the ski resort.
address: Physical address of the resort.
website: Website URL of the resort.
photo: Photo URL of the resort.
mapUrl: Google Maps URL of the resort.
postalCode: Postal code of the resort.
date: Date for which the information is provided.
distance: Distance from the postal code to the resort.

Sample Calls:
JavaScript

```
fetch('http://3.145.39.202/skiresortsbydistance?postalCode=H3S2L4&date=15-10-2023&limit=3
')
  .then(response => response.json())
  .then(data => console.log(data));
```

Browser
http://3.145.39.202/skiresortsbydistance?postalCode=H3S2L4&date=15-10-2023&limit=3

Terminal
curl http://3.145.39.202/skiresortsbydistance?postalCode=H3S2L4&date=15-10-2023&limit=3

Sample JSON Response:


```
[
  {
    "name": "Poubelle du Ski",
    "address": "8278 Boul. Saint-Laurent, Montréal, QC H2P 2L8, Canada",
    "website": "https://www.poubelleduski.ca/",
    "photo":
"https://lh3.googleusercontent.com/places/ANXAkqHVgqPHBYaZ3FJb_TNa3zaE3E5SnIurrn2c
N4I8CYAUST4u5zcPubhSJZnEHXXHkt_suNIRcD3qoL1q4MXszmRUEf7lYskPT6U=s1600-w
800",
    "rating": 4,
    "mapUrl":
"https://www.google.com/maps/embed/v1/view?key=AIzaSyBq_hYo4JWsz1s6HglTs1doKeGdA
hHlt3U&center=45.5395366%2C-73.63423&zoom=15",
    "postalCode": "H2P 2L8",
    "date": "15-10-2023",
    "distance": 3.2
  },
  {
    "name": "Ski Benoit",
    "address": "4632 Rue Richard Hewton, Lachine, QC H8T 1P3, Canada",
    "website": "http://skibenoit.com/",
    "photo": "./placeholder.jpg",
    "mapUrl":
"https://www.google.com/maps/embed/v1/view?key=AIzaSyBq_hYo4JWsz1s6HglTs1doKeGdA
hHlt3U&center=45.4381061%2C-73.7079188&zoom=15",
    "postalCode": "H8T 1P3",
    "date": "15-10-2023",
    "distance": 10.4
  },
  {
    "name": "SKI TOWN Brossard",
    "address": "8025 Boulevard Taschereau unité K, Brossard, QC J4Y 1A4, Canada",
    "website": "http://skitown.ca/",
    "photo":
"https://lh3.googleusercontent.com/places/ANXAkqF4QuNFOMHjqq4jn7F1pMePgu8uIjVIc1Js
DS13budyRSV_Wh4NMr02Jkp7Yks8l7DTKWJV8iK7idXbGywlaUc2MFp5PUgXvqw=s1600-
w800",
    "rating": 4.4,
```

```
   "mapUrl":
"https://www.google.com/maps/embed/v1/view?key=AIzaSyBq_hYo4JWsz1s6HglTs1doKeGdAhHlt3U&center=45.4504472%2C-73.4671652&zoom=15",
   "postalCode": "J4Y 1A4",
   "date": "15-10-2023",
   "distance": 14.1
  }
]
```


Error Codes:
400 Bad Request: Postal code and date are required.
400 Bad Request: No resorts found near the postal code.
500 Internal Server Error: Any server-side error.

3. Weather Information
Endpoint: /weather/:postalCode

Description:
Provides the current weather information based on the postal code and date provided. The data includes temperature, weather forecast, wind speed, and snow depth.

Path Parameter:
postalCode: Postal code for which the weather information is required.

Query Parameter:
date: Desired date for the weather information (format: DD-MM-YYYY).

Attributes Returned:
temp: Current temperature.
wForecast: Weather forecast description.
wind: Wind speed.
sDepth: Snow depth.

Sample Calls:
JavaScript
```
fetch('http://3.145.39.202/weather/:postalCode=H3S2L4&date=15-11-2023')
  .then(response => response.json())
  .then(data => console.log(data));
```

Browser
http://3.145.39.202/weather/:postalCode=H3S2L4&date=15-11-2023

Terminal
```
curl http://3.145.39.202/weather/:postalCode=H3S2L4&date=15-11-2023
```

Sample JSON Response:


```
{
  "temp": 5,
  "wForecast": "Partially cloudy",
  "wind": 11.5,
  "sDepth": 0
}
```


Error Codes:

400 Bad Request: Postal code is required.
404 Not Found: No weather data found for the postal code.
500 Internal Server Error: Any server-side error.