Sonia Ortiz
SWDV 691

Database Technology

This app will use a document database such as MongoDB. This database will house the information output generated by a API that the users input their location information. In this scenario there are no entity relationship diagrams, the document database would store all the information for the "weather" this way every stored object can be different from every other. This eliminated the need for object relational mapping while loading data into the database.

Database Design

The example below is the is the output information after a user inputs their location (city, state, or zip), this information will be displayed on the webpage as well as stored in the document database. Since there is no user sign in if a user were to make another location request another document would be populated into the database.

MVP will only include current weather information on home page.

```
{
```

LOCATION INFORMATION

```
  "coord": {
    "lon": Number,
    "lat": Number
  },
```

WEATHER DATA

```
  "weather": [
    {
      "id": Number,
      "main": String,
      "description": String,
      "icon":
    }
  ],
```

BASE WILL NOT BE DISPLAYED
WEATHER SPECIFIC DATA WILL BE DISPLAYED

```
  "base": "stations",
  "main": {
    "temp": Number,
    "feels_like": Number,
    "temp_min": Number,
    "temp_max": Number,
    "pressure": Number,
    "humidity": Number
  },
  "visibility": Number,
  "wind": {
    "speed": Number,
    "deg": Number
  },
```

Sonia Ortiz
SWDV 691

DATE/SUNRISE/SUNSET
```
 "dt": Number,
   "sunrise": Time,
   "sunset": Time
 },
```
TIME ZONE INFORMATION
```
 "timezone": String,
 "name": String,
   }
```

Here is a detailed break down of the columns for document database

- coord
  - coord.lon City geo location, longitude
  - coord.lat City geo location, latitude
- weather (more info Weather condition codes)
  - weather.id Weather condition id
  - weather.main Group of weather parameters (Rain, Snow, Extreme etc.)
  - weather.description Weather condition within the group. You can get the output in your language.
  - weather.icon Weather icon id
- base Internal parameter
- main
  - main.temp Temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  - main.feels_like Temperature. This temperature parameter accounts for the human perception of weather. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  - main.pressure Atmospheric pressure (on the sea level, if there is no sea_level or grnd_level data), hPa
  - main.humidity Humidity, %
  - main.temp_min Minimum temperature at the moment. This is minimal currently observed temperature (within large megalopolises and urban areas). Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  - main.temp_max Maximum temperature at the moment. This is maximal currently observed temperature (within large megalopolises and urban areas). Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  - main.sea_level Atmospheric pressure on the sea level, hPa
  - main.grnd_level Atmospheric pressure on the ground level, hPa
- wind
  - wind.speed Wind speed. Unit Default: meter/sec, Metric: meter/sec, Imperial: miles/hour.
  - wind.deg Wind direction, degrees (meteorological)
  - wind.gust Wind gust. Unit Default: meter/sec, Metric: meter/sec, Imperial: miles/hour

- clouds
  - clouds.all Cloudiness, %
- rain
  - rain.1h Rain volume for the last 1 hour, mm
  - rain.3h Rain volume for the last 3 hours, mm
- snow
  - snow.1h Snow volume for the last 1 hour, mm
  - snow.3h Snow volume for the last 3 hours, mm
- dt Time of data calculation, unix, UTC
- sys
  - sys.type Internal parameter
  - sys.id Internal parameter
  - sys.message Internal parameter
  - sys.country Country code (GB, JP etc.)
  - sys.sunrise Sunrise time, unix, UTC
  - sys.sunset Sunset time, unix, UTC
- timezone Shift in seconds from UTC
- id City ID
- name City name
- cod Internal parameter

Stretch Feature

Sonia Ortiz
SWDV 691

The 30 day forecasts will be part of the almanac page which has been changed to a stretch feature. This information will still be stored in a document database but will have a different table/column layout (see below). Below is a labeled JSON response, location details are gathered from the user with the weather details being the output. Based on location input the weather information for the next 30 days will be outlined for the user.

```
{
```

LOCATION DETAILS BASED ON USER INPUT / THIS IS AN OUTPUT RESPONSE FROM ZIP CODE OR CITY STATE INFORMATION

```
  "cod":"Number",
  "city":{
    "name":"String",
    "coord":{
      "lon": Number ,
      "lat":Number
    },
    "country":"String"
  },
```

LIST OF 30 DAYS WITH WEATHER DETAILS OUTPUT INFORMATION

```
  "list":[
    {
```

TIME

```
      "dt":number,
      "sunrise":Time,
      "sunset":Time,
      "temp":{
        "day":Number,
        "min":Number,
        "max": Number,
        "night": Number,
        "eve": Number,
        "morn": Number
      },
      "feels_like":{
        "day": Number,
        "night": Number,
        "eve": Number,
        "morn": Number
      },
      "pressure": Number,
      "humidity": Number,
      "weather":[
        {WEATHER DESCRIPTIONS AND ICONS OUTPUT
          "id":number,
          "main":"string",
          "description":"string",
          "icon":"string"
        }
      ],
      "speed":number,
      "deg":number,
      "clouds":number,
      "rain":number
    },
```

Here is a detailed break down of the columns for document database

- city
  - city.id City ID
  - city.name City name
  - city.coord
    - city.coord.lat City geo location, latitude
    - city.coord.lon City geo location, longitude
- country Country code (GB, JP etc.)
- population City population
- timezone Shift in seconds from UTC
- cod Internal parameter
- message Internal parameter
- list
  - list.dt Time of data forecasted
  - list.sunrise Sunrise time, Unix, UTC
  - list.sunset Sunset time, Unix, UTC
  - list.temp
    - list.temp.day Day temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
    - list.temp.min Min daily temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
    - list.temp.max Max daily temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
    - list.temp.night Night temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
    - list.temp.eve Evening temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
    - list.temp.morn Morning temperature. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  - list.feels_like
    - list.feels_like.day Day temperature.This temperature parameter accounts for the human perception of weather. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.

    - list.feels_like.night Night temperature.This temperature parameter accounts for the human perception of weather. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
    - list.feels_like.eve Evening temperature.This temperature parameter accounts for the human perception of weather. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
    - list.feels_like.morn Morning temperature. This temperature parameter accounts for the human perception of weather. Unit Default: Kelvin, Metric: Celsius, Imperial: Fahrenheit.
  - list.pressure Atmospheric pressure on the sea level, hPa
  - list.humidity Humidity, %
  - list.weather (more info Weather condition codes)
    - list.weather.id Weather condition id
    - list.weather.main Group of weather parameters (Rain, Snow, Extreme etc.)
    - list.weather.description Weather condition within the group. You can get the output in your language.
    - list.weather.icon Weather icon id
  - list.speed Wind speed. Unit Default: meter/sec, Metric: meter/sec, Imperial: miles/hour.
  - list.deg Wind direction, degrees (meteorological)
  - list.clouds Cloudiness, %
  - list.rain Precipitation volume, mm
  - list.snow Snow volume, mm
  - list.cnt Number of lines returned by this API call