Hello! If you are reading this, we're likely in the process of a Senior Backend Developer role at Betsson Group. If so, congratulations!

In this assignment, we are allowing you to showcase your talent on the technical side.

Assignment Details

Weather API Integration with Database Enrichment
We create an API for users to get the weather conditions of the country where they are registered in our system.

What are the requirements?

1. Create an API endpoint for user authentication and weather information retrieval based on the user's country.

2. Upon registration, validate user data against Countries API.

3. Generate unique usernames automatically.

4. Secure the endpoint with Basic Authorization.

5. Cache weather data for performance optimization.

What we look for

Clean, **readable,** and easily **testable** code.

Time & Delivery

- Kindly limit the development time 1 week

- Please submit the finished assignment in a zip file and reply to HR's email.
- Make sure the project builds even if the solution is not complete.

Endpoints

1. *url: /api/registration* , HTTP Verb : POST , Authentication is no required

The clients will register themself via this endpoint. Username will be generated automaticlly

For example: if the user full name is Jack Doe then username will look like this "jado12345"

2. *url: /api/login* , HTTP Verb : POST , Authentication is no required

The clients can be login via this endpoint. After the login other endpoint must used generated access token.

3. *url: /api/weather/{username}* , HTTP Verb: GET , Authentication is required

Returns weather data for a location provided in the requestt

Request Validation: Validate the request to ensure a location is provided.

API Integration: Integrate with an external weather API (e.g., OpenWeatherMap or Open-Meteo) to fetch current weather data.

Database Enrichment: Enrich response with additional data from a SQLite database (e.g., historical weather, local attractions).

Output Caching: Cache response for one hour in memory to improve performance.

Unit Test: Provide a test ensuring correctness of the weather endpoint.

Documentation: Provide clear setup instructions, environment configuration, and API access documentation.

Repository Link: [Provide Repository Link Here]

Plus points:

Error Handling: Implement global exception handling for stability.

Testing: Write comprehensive tests covering validation, API integration, database enrichment, and caching.

OpenWeatherMap: https://openweathermap.org/current

Open-Meteo: https://open-meteo.com/en/docs

(Or similar publicly available api)

SQLlite: https://www.sqlite.org/ (Or you can use any one of your choice PostgreSQL,MySQL etc. , provided that you share the necessary scripts in the repo)

Countries API: https://restcountries.com/v3.1/alpha?codes={countryode},{countryode},{countryode}

Example: https://restcountries.com/v3.1/alpha?codes=MLT,SWE,GBR

Additional Information: The idd property in the response payload represents the country phone code info.

Tables:

Users{

Id,

Firstname,

Lastname,

Username,

Email,

Password,

Address,

Birthdate,

PhoneNumber, (Must be with prefix)

LivingCountry,

CitizenCountry

}