ByteGenie FullStack Developer Test

The objective of this test is to create an AI-powered application that allows users to interact with events, company, and people data to

- Identify companies that are attending certain types of events, e.g. event over a given period, or events in a certain industry;
- Find people working for companies who are attending events of certain types;
- Find events that are being attended by companies of a certain type, e.g. companies with revenue above or below some threshold, or companies that have number of employees above or below a certain threshold, etc;
- Find email addresses of people with certain type of job descriptions working for certain type of companies (e.g. “sales people working in tech companies”);
- etc.

The user should be able to interact with the application in the form of natural language, and the app should query the relevant data, and generate the desired output based on user’s prompt/instruction in natural language.

Some examples of relevant user queries are

- Find me companies that are attending Oil & Gas related events over the next 12 months;
- Find sales people for companies that are attending events in Singapore over the next 9 months.
- Find me events that companies in Pharmaceuticals sector are attending;
- I need the email addresses of people working for companies that are attending finance and banking events;

You are free to implement any number of user instruction of this form that you think are useful, and can be implemented based on the data. However, it would be better to have at least a few instructions that work well than having too many instructions, neither of which works very well.

Based on this instruction the app needs to decide which datasets to query, and how to generate the right output. One way to do this is to store that data into an SQL database, and use an LLM to convert the user query into an SQL query to retrieve the relevant data, and show it to the user. Another way to do it is to use an LLM to parse the user query to extract key attributes, and then construct one or more SQL query algorithmically, and retrieve and combine the data, and show it to the user. There can be other ways to do this, and you are free to use whichever approach seems best.

The relevant data for this test can be found in this folder. This contains a few CSV files, one containing events data, one containing company info, one containing people info. Events and company data can be merged using ‘event_url’ column. Company and people data can be merged using ‘homepage_base_url’ column. Each event_url corresponds to a unique event, and each homepage_base_url can be interpreted as a unique company.

You are welcome to add any other columns in the data that can be derived from existing data. For example, you can use an LLM to tag each event to an event industry based on event name and description. Similarly, you can use an LLM to create people emails based on the email pattern, and person's name provided in the data. You can also feel free to standardise any columns, like company_industry, company_revenue, number of employees, etc. that might help in effectively querying the data.

The final application should consist of a Python API and a React App, and a database.

Please submit your application as links to one or more GitHub repos, which should contain all the code needed for running the application, in an email to hr@esgnie.org

The API GitHub repo should include a README.md file that summarises:
- main steps and motivation for any data engineering/processing on the raw data before making it available to the API;
- the main functionalities of the API;
- the key challenges you faced in solving the problem;
- how would you improve the backend, if you had more time to work on it?

The UI GitHub repo should include a README.md file that summarises:
- key functionalities of the UI;
- the API endpoints the UI uses;
- the key challenges you faced in building the front-end;
- how would you improve the front-end, if you had more time to work on it?

The database GitHub repo should include a README.md that summarises:
- database schema;
- The challenges you faced in working with this data;
- How would you improve the database design if you had more time to work on it?

If you have any questions, please feel free to send them to hr@esgnie.org