Task story

Contacts table holds the information about the contacts of a company. This table can have data being pushed from different sources. Source tables for example are described as leads, customers etc. Data from these source tables can be pushed to the contacts table via an API.

For example:

https://{url}/contacts?lead_id=1
https://{url}/contacts?customer_id=1

Above endpoint will get the corresponding lead or customer info from respective tables and inject the relevant information to the contacts table.

You only need to write an injection service that will handle the fetching of data from the corresponding source table and add it to the contacts table.

Consider below DB schema

contacts table:

- id
- first_name
- last_name
- postal_code
- city
- street_name

leads table:

- id
- first_name
- last_name
- postal_code

customers table:

- id
- first_name
- last_name
- postal_code
- city

Technical todo's

- Create an injection service class that handles the injection of data into the contacts table.

- Injection of data into the contacts table should only happen for classes with proper class signature (perhaps with interface).

- Injection of data from each table like leads, customers should be handled in its own class.

- The main service which handles the injection should not have any trace of any source injections (i.e LeadInjection, CustomerInjection service should not have direct implementation in this class).

- Data that is injected into the contacts table should be always via data transfer objects.

- The injection service should be extendable which can accept other data injections into the contacts table in future. For example if there is a new table called 'debtors', that has a similar structure, then with a new class for this service, it should be possible to get data from the 'debtors' table into the contacts table.

- Write proper unit tests to showcase the working of the injection service. For example tests like:

  testItInjectsLeadToContactsTable

  In this test, a lead record will be fetched from the leads table and passed into the lead injector service. This injector service when passed to the main service class should assert the contact with the lead data.

Requirements

- Send the code in a zip file
- Add a proper README file containing clear simple instructions on how to execute the code and tests.
- If any clarifications are needed which are not explained here, make an assumption and note these in the README file.
- Preferably use Laravel framework
- Preferably implementation of a design pattern

What we look at

This task is designed to give us an idea of:

- How you structure your code.
- Your ability to deliver an appropriate, simple solution to a given problem.
- How you work when faced with limited time to solve a problem.

Expected Impact

4 - 6 hours