ワークサンプルテスト（バックエンドエンジニア）

損害保険会社は、様々な種類の保険をお客様に提供しています。お客さまは、各保険代理店で保険を契約することが多いのですが、代理店ごとに独自のDBを持ち、情報が散在しています。現在、顧客レベルで全ての情報を統合する、顧客一元化DBをおよび、そのAPIを企画中です。本ワークサンプルテストでは、その企画のディスカッション、およびプロトタイプを実施します。

**仕様**

1. 保険代理店A,B,Cは、以下のような保険契約DBを持つ。

**代理店契約DB　(agent\_a.csv, agent\_b.csv, agent\_c.csv)**

契約ID、商品ID、氏名、電話番号、郵便番号、住所

1. 保険商品を定義したDBもある。

**契約商品DB (products.csv)**

商品ID、商品名

1. 各代理店の情報を統合し、顧客起点となる以下のようなDBを作成したい。上記代理店契約DBから、以下のDBをつくるためには、契約情報から「名寄せ」を行い、名寄せされた顧客に顧客IDを紐付ける必要がある。

**顧客契約DB (transactions.csv)**

顧客ID、商品ID、代理店ID

**顧客DB (customers.csv)**

顧客ID、氏名、電話番号、郵便番号、住所

1. 顧客IDをキーに、顧客が現在契約中の保険商品一覧と、その詳細を取得するAPIを作成したい。ただし、以下のセキュリティ要件がある
   * 1. 代理店は他の代理店で契約されたものにはアクセスできない
     2. 保険会社は全ての情報にアクセスできる

**ディスカッションポイン**ト

エンジニアとして、上記仕様を元に、ビジネスサイドの人間と、プロトタイプする内容を詳細に詰めてください

* 1. 名寄せロジックをどうするか、サンプルデータを見ながら詳細を詰めてください。名寄せは今後想定しないようなパターン（特に住所）が登場する可能性もあります
  2. APIのセキュリティの実現方法ついて議論してください。プロトタイプでどこまで実装するかも決めてください

**プロトタイプ**

ディスカッションで決まった内容をもとに、実際に動くプロトタイプ（サンプルデータ作成を含む）を作成してください。以下プロトタイプの要件となります

1. 代理店契約DB (A、B、C)のデータを名寄せし、顧客契約DBと顧客DBを作成するプログラムを用意してください
2. ２を実行後、顧客IDをキーとして契約情報一覧が取得できるAPIを実装してください（可能な範囲でセキュリティ要件を実装してください）
   1. Input: 顧客ID
   2. Output: 商品一覧＋代理店情報
      1. 商品ID、商品名、代理店ID
3. 実装には以下のテクノロジーの何かを利用して下さい
   1. Java
   2. Python
   3. Javascript (Node.js)
   4. Rust
   5. Go
   6. Kotllin
4. REST APIの実装には、既存のメジャーなFrameworkを使うこと（可能であればGraphQLを推奨）
5. PostgreSQL v14がローカルで実行されていることを想定して下さい
6. 名寄せロジックのUnit Testを実装してください
7. APIのUnit Testを実装してください

上記条件を満たすプログラムを作成し、githubで公開してください。Readmeに、プログラムを実行するための各種前提条件と、実行方法を詳細に記述してください。Mac環境で検証します。

For 面接官

**技術的検証ポイント**

* 名寄せはある程度のロジック能力チェック
* APIだけじゃなくてone shotのプログラムもかけるか
* DBの基本的知識
* セキュリティのところ
* API作成
* サンプルデータ作成と、Unit Testで品質への対応が見える

**ディスカッション検証**

* 名寄せの調整
  + 正規化
    - 前後スペースの削除
    - 全角、半角とか
    - 各種区切り文字 / とか –
    - 電話番号は(090)の括弧とか
    - ３番地二上目 -> 3-2とか
  + 住所の名寄せ
    - 正規化した上で
    - 完全一致、前方一致、後方一致
    - プラガブルにして欲しい
  + 名寄せした上で何を正しいデータとするか
    - 電話、住所とも長い方を優先
* APIセキュリティのところ
  + 認証・認可の話がわかってるか。よりセキュアにするにはどうするか。
  + PostgreSQLなのでRow Level Securityとかも使える。知ってるか。