Missing Field Identification

The maximum amount of time allocated for this assignment is 4 hours. If you find yourself spending more than that please pause work and submit what you've accomplished.

- We want to be respectful of your time as a candidate and NOT force you to rush and try and complete a polished project in an unreasonable amount of time.
- This project is meant to provide you opportunities to demonstrate decision making and actual coding in a short amount of time.
- This is exactly the type of feature we would be asking you to implement if you were to take on the role, so imagine you were doing this as your first day task. Consider what would be high code quality - and strive for it (within the time limit, of course).

Background

At Trellis, we do our best to collect as much policy information as possible from each insurer's website. However, since each insurer has a different site, some fields are not always available from each insurer (e.g. an insurer might not display a driver's date of birth on their page) and we might get different levels of completeness for the fields (e.g. we only get a partial driver's license).

Without a complete set of data, we cannot offer customers competitive quotes from other providers - so we need to collect that information through outreach from our customer success department. To make it as easy as possible for our customers, we only want our customer success team to ask for the information that is missing or incomplete from the customer's profile.

Problem

In order to request an insurance quote from The Colonel Insurance, they require the following fields:

- the address of the policy holder
- the name of the policy holder
- the email address of the policy holder
- the complete driver's license number of the policy holder
- the exact date of birth of all operators
- the gender of all operators

Please submit a function that takes in a single policy and returns a list of the required fields that are not available in the policy. The file `policies.json` contains an array of example policies you can use to aid you in the development of your function.

Your function should return *exactly what information* our support team needs to ask for in order to have a complete policy that we can use to request quotes from The Colonel Insurance.

Other Considerations

Suppose that in addition to generating quotes from The Colonel Insurance, we also wanted to request quotes from Ranchers Insurance. However, the set of fields that Ranchers Insurance requires to generate quotes is slightly different - e.g. don't require gender, but they want the license numbers for all operators and also want the policy holder's phone number. How would this affect your solution?

Suppose that instead of being consumed by a support team member, the list of questions was going to be used to power a programmatic, online interview flow. Would there be anything you would change in your solution?

Submission and Evaluation

Please upload your submission as a ZIP file to us via email at tech-lead+code@trellisconnect.com or to the provided submission link. Be sure to include instructions inside the ZIP file regarding how to run your submission.

Your submission will primarily be evaluated on the code quality (readability, maintainability, scalability, testing, etc.), mastery of the language used, and your ability to identify and appropriately handle the nuances in the data. Consideration will also be given towards satisfaction of business requirements and communication.

Data Definition

```
{
  policyId: unique identifier for policy,
  issuer: company that issued the policy,
  issueDate: date policy starts, YYYY-MM-DD
  renewalDate: date policy ends, YYYY-MM-DD
  policyTermMonths: duration of policy in months, 6 or 12
  premiumCents: total premium (cost) for the term of the policy, in cents
  policyHolder: {
    name: {
      firstName: policy holder's first name
      middleName: policy holder's middle name
      lastName: policy holder's last name
    }
    address: {
      number: street number for policy holder's address
      street: street name for policy holder's address
```

```
    type: street type for policy holder's address
    sec_unit_type: secondary unit type for policy holder's address
    sec_unit_num: secondary unit number for policy holder's address
    city: city for policy holder's address
    state: state for policy holder's address
    zip: zip code for policy holder's address
  }
  email: policy holder's email
  phone: policy holder's phone number
}
operators: [{
  isPrimary: is the operator the policy holder
  name: {
    firstName: operator's first name
    middleName: operator's middle name
    lastName: operator's last name
  }
  birthdayRange: {
    start: start of operator's birthday range
    end: end of operator's birthday range
    // sometimes we can't extract the exact birthday
    // (e.g. driver is 35 years old), so we save the implied range in this case
  }
  gender: operator's gender
  driversLicenseStatus: operator's driver's license status (valid, invalid, etc.)
  driversLicenseState: operator's driver's license state
  driversLicenseNumber: operator's driver's license number
  relationship: operator's relationship to policyholder
}]
```