Question 3.

What are the possible reasons for the following defect? How would you go about debugging the problem and gathering more information?

On a web application, a user adds a phone number to their account. The user then changes the phone number. Upon trying to re-enter the first phone number, the user is allowed to click Save, and it seems to work, but the saved number remains the second number rather than updating to the more recently entered number. A page refresh does not change the result.

Answer:

When a user adds the new phone number, changes the existing values and populates the same latter value, it might not save for a couple of reasons as follows:

1. Data Load Delays:

There might be inconsistency in the speed of upload of information sent to the backend and processing time for the server. When there is a delay in corresponding functions, the front-end may be working ahead of backend servers. In this case, we first need to make sure that the value is updating through frontend using developers tool or networking tool. We can look at the updates in the network calls and their data preview in response body.

2. No Reuse Policy:

There might be a possible constrain applied on the field which is just not visible to the user for the corresponding phone number field. There may be a session, time constraints on the field only after which user may be allowed to update the give phone number. The user might have to wait for a couple of minutes/hours or defined period of time to re-populate the phone number field with previous values.

3. Initial Value Does not Update at all

There might be another reason to which the phone number does not update back to previous values. Which might be that post creating an account the database is indexing the user account credentials with initial value as primary key. This does not update the user phone number when

changing the phone number, instead created a new user account with phone number2 credentials in the database as primary key. In this effect, this might amplify to be a critical defect.

Debugging and Gathering Information:

We can gather more information through manually testing front end, looking at networking calls, running backend queries, and test for API.

1. Manual Verification: We can use developer's tool in the frontend to gather information regarding the network calls, and look into the responses for every request made by updating the user phone number. We should get ample amount of information from this spot, to classify this issue as frontend or backend.

2. Running through API: We can write test driven test cases to perform operation on the API calls. We can use postman or use any tool that will return us a response with endpoints and their values.

3. Code and Structure: Adding toggling points at this instance in the server side scripting code to update the phone number, and call through simple stdout calls will return us the flow of the events, and their return functionality.

4. Database Investigation: We can query out the resultant database by identifying the customer, given metro, and/or email address which is ideally supposed to be the primary key and retrieve the result of the value of phone number that is stored.