1. **Patients can message Karuna from various different mediums (Facebook Messenger, SMS, etc.)**

   Questions:

   1. **How to communicate with different mediums?**
      - There should be an API which we can talk to. And each message can sync with a Karuna web messaging app. It could be a one to many relationship. Ex) Many mediums can be used to communicate and deliver messages to one messaging center within Karuna.
   2. **Should Karuna keep track of the different accounts the user has linked up?**
      - A database table can be created which can keep track of the different linked accounts and the userId can serve as a foreign key. To simplify things even more, within the users table, there can be a column which displays the different accounts they have set up. Based on that, messages coming from those accounts can be received by Karuna.
      - There should be a process where the user links their different accounts and they should be able to go back and add accounts later also.
2. **The same patient can contact us on different mediums at different times, and receive responses on the last medium they used**

   Questions:

   1. **How to architect the messaging DB?**
      - There should be a conversation ID which links to the different mediums. So a message sent from facebook to conversation ID ab123 will display the message in the interface. And if an SMS is also sent with that same conversation ID then that message will also link to the same conversation. The type of way the messaged is received might be different but because the conversationID is the same, it links to the same conversation.
   2. **How to manage the different relationships?**
      - Each medium will have a different relationship. If we can to respond based on the medium the user last messaged Karuna from then there should be a ‘medium type’ that we store and also have a

service which will hit that api endpoint to send the coordinator's response to that particular medium.
    - I suggest we also send the message within the Karuna web app. So if they check the Karuna app before the last used Medium, the patient/user can view the message on both applications.
3. **Messages from patients can get routed to the correct coordinator, based on a set of rules**

   Questions:

    1. **Build a system that evaluates based on rules or conditions?**
        - There should be a set of conditions, similar to if and else and we can have metadata which can have boolean responses and based on a combination of statements, the patient can be routed to the correct coordinator.
    2. **What should the rules be?**
        - Rules should be defined by coordinator. It can be a set of conditions which must be true or false and based on the user criteria, it will be easy to determine which coordinator they should be routed to.
4. **Coordinators have a UI which shows all their routed messages in one place**

   Questions:

    1. **How to decide if a user is an coordinator?**
        - We can add a 'type' to the users database schema and if it says coordinator then that user will have certain permissions.
    2. **How to decide if the logging in user is a coordinator and present them a particular interface?**
        - When the user is logging in, that login endpoint should return the 'type' of that user nad based on that, we can route the user to UI.
5. **Coordinators can respond to messages without worrying about which medium will be used**

   Questions:

    1. **How create a system or interface which allows messaging from different mediums?**
        - If the user can send messages to karuna from different mediums then the coordinator can be given similar permissions and rights.

- They can also just message on the karuna interface or app and it can be configured to notify the user via email or a message of some sort.

6. **Coordinators can "hand off" a patient to another coordinator**

   Questions:

   1. **How to create a relationship between a coordinator and a patient?**
      - This can be a one - to - one relationship where the user
   2. **Can one patient have multiple coordinators?**
      - Initially let's have one coordinator and maybe a secondary coordinator if we really need to have two of them. So if your coordinator is not available and it is an emergency then the secondary is informed.

7. **Audio and video interactions are recorded and transcribed**

   Questions:

   1. **Where and how to store this?**
      - We can store links and meta data in a database table and store the actual media in a S3 bucket.
   2. **How to decide which video and audio links to which conversations and which people?**
      - We can create a database table called 'media' and the columns can include the user and coordinator ids, timestamps of when the conversation took place, along with a link to the media itself. This way, we will know when and between whom these conversations took place.

8. **Both incoming and outgoing messages are delivered in-order and exactly-once**

   Questions:

   1. **How to keep track of the ordering of messages?**
      - There should be a timestamp that sticks to ever message so that way, it can be queried in order of relevance.
   2. **How to know a message was delivered?**
      - Once we send a message, we should receive a response from like a '200' or a 'delivered' so we know the server has received the message. There should also be other responses, such as it has

been sent from our end, but not received from there so if something isn't delivered then it is easy to know if the delivery failure is from which end.

3. What to do if the message isn't delivered in the first try?
    - If the message isn't delivered, we can set up a service which tries to hit their server again in 5 minutes or so?

9. **The external systems for delivering messages can be easily changed**

    Questions:

    1. **How to set things up so if one external delivery system is removed, it does not affect the rest of the system?**
        - The links to external systems should be via an API or other external services so the system isn't something stored within our system. There should be a secure way to communicate and utilize the services without having to keep anything on our end. The only thing we should need to build are services to communicate between our database and their APIs.
    2. **How to structure the overall infrastructure for delivery messaging?**
        - We should structure everything in pieces. Each delivery system should have it's own service so if one is extracted, removed or modified then it does not affect the whole system.

10. **Various metrics (such as the maximum time a patient waits for a response) can be reported and alerted on.**

    Questions:

    1. **How to keep track of a long a patient was waiting?**
        - This is simple meta data we can keep track of. A simple but probably not them most efficient way to do this is storing a timestamp of the time the user first contacted us to the time when we were able to present a response.
    2. **How to alert/report this informations?**
        - There can be reports and analysis of how system as well and this metadata and statistics can be presented in that. We can also set thresholds to where we are alerted if a user waited more than a certain period of time.

- We can also set up alerts for improvements of the response time. For example, if the system responded sooner than it usually does, then that’s an improvement in efficiency and response time.