Perguntas e respostas:

**1. Definir o Contexto do App**

**Pergunta:**  
Descreva em uma frase qual será o propósito do app que sua equipe decidiu desenvolver.

**Resposta:**  
O propósito do app é gerenciar usuários, permitindo o cadastro de informações pessoais (nome e CPF) e o controle do saldo financeiro de cada usuário.

**2. Definir as Entidades**

**Pergunta:**  
Escreva como vocês definiriam as entidades (classes) e descreva quais campos cada entidade deve conter.

**Resposta:**  
A entidade principal no app é a classe Usuario, que representa a tabela Usuario no banco de dados. A definição da entidade é a seguinte:

* id: Int, chave primária auto-gerada para identificar cada usuário de maneira única.
* nome: String, armazena o nome do usuário.
* cpf: String, armazena o CPF do usuário.
* saldo: Double, representa o saldo financeiro associado ao usuário.

**3. Criando o DAO (Data Access Object)**

**Perguntas:**  
Quais métodos o DAO precisa ter? Exemplifique com pelo menos 3 métodos (por exemplo: inserir, buscarTodos, remover).  
Como cada método vai atuar sobre os dados das entidades?

**Respostas:**  
O DAO (UsuarioDAO) precisa fornecer métodos para manipulação dos dados, incluindo operações de inserção, consulta e exclusão. Aqui estão três exemplos de métodos implementados:

* insertUsuario(usuario: Usuario): Insere um novo Usuario no banco de dados. Esse método atua criando uma nova linha na tabela Usuario com os dados do usuário fornecido.
* getAllUsuarios(): Retorna todos os usuários cadastrados no banco de dados. Esse método atua lendo todos os registros da tabela Usuario.
* deleteUsuario(usuario: Usuario): Remove um usuário específico do banco de dados. Esse método localiza o usuário pelo ID e o remove.

**4. Definindo o Banco de Dados Room**

**Pergunta:**  
Descreva como sua equipe configuraria a classe de banco de dados, mencionando quais DAOs ela conteria e como a instância seria gerenciada.

**Resposta:**  
A classe AppDatabase é definida como uma extensão de RoomDatabase e contém a tabela Usuario e o DAO UsuarioDAO. A configuração da classe está assim:

* **Entidades**: Inclui a entidade Usuario.
* **DAOs**: Contém UsuarioDAO, que gerencia as operações de CRUD.
* **Padrão Singleton**: A instância do banco de dados é gerenciada com o padrão Singleton usando INSTANCE e getDatabase, garantindo que apenas uma instância do banco seja criada ao longo do ciclo de vida do app.
* **Versão do Banco**: Está definida como 1, o que facilita futuras migrações.

**5. Simulação de Fluxos de Inserção, Consulta e Exclusão**

**Perguntas:**  
Descrevam como seria o fluxo completo de inserção de um item no banco, desde o momento em que o usuário interage com o app até o momento em que os dados são salvos.  
Descrevam como os dados serão exibidos no app. Onde eles aparecerão?  
Como o app permitirá que o usuário exclua um item? Explique o processo.

**Respostas:**

* **Fluxo de Inserção**:
  + O usuário preenche os campos “Nome do Usuário”, “CPF do Usuário” e “Saldo Inicial”.
  + Ao clicar no botão "Salvar Usuário", os dados inseridos são coletados, e um novo objeto Usuario é criado.
  + O app usa o método insertUsuario para inserir o novo Usuario no banco.
  + Após a inserção, os campos de entrada são limpos.
* **Exibição dos Dados**:
  + Os dados são exibidos ao clicar no botão "Listar Usuários", que chama o método getAllUsuarios.
  + A lista retornada é exibida usando um LazyColumn, mostrando cada usuário com seu saldo na tela.
* **Exclusão de um Item**:
  + Ao lado de cada usuário na lista, há um botão “Deletar”.
  + Quando o botão é clicado, o método deleteUsuario é chamado para remover o usuário selecionado.
  + Em seguida, a lista é atualizada para refletir a exclusão, chamando getAllUsuarios novamente.

**6. Desafios Criativos**

**Pergunta:**  
Descreva uma melhoria criativa que sua equipe poderia implementar no app, explicando o impacto que ela teria.

**Resposta:**  
Uma melhoria criativa seria implementar uma validação que impede salvar um usuário sem nome ou CPF, mostrando uma mensagem de erro para o usuário preencher os campos obrigatórios. Outra melhoria seria ordenar os usuários por nome ou saldo, oferecendo uma opção de filtro para facilitar a navegação. Esses ajustes melhorariam a experiência do usuário ao tornar a aplicação mais organizada e fácil de usar.