AWS Deployment Steps

a. AWS Deployment Plan

The following steps outline the AWS services used for deploying the Intelligent Medical Assistant system and scaling it for real-world applications.

1. Data Storage:

- **Amazon S3** is used for storing datasets such as MedQA and the medical text corpus. S3 provides durable, scalable, and secure storage, ensuring that medical datasets are easily accessible for model training and evaluation.

2. Model Hosting:

- The fine-tuned **T5 model** is deployed using **AWS SageMaker**, which provides a fully managed environment for hosting machine learning models. SageMaker enables easy scaling of model inference, allowing it to handle multiple concurrent user queries.

3. API Development:

- **AWS Lambda** is used to handle incoming requests. Lambda functions execute the retrieval and model invocation logic when a user submits a query. This serverless approach allows the system to scale automatically, only running when needed, and ensures cost-effective management of computing resources.

4. Frontend Hosting:

- If a frontend application is required, **AWS Amplify** can be used to host the web-based user interface. Amplify integrates seamlessly with AWS backend services and provides a fast and secure way to deploy the UI, allowing users to input medical queries easily.

5. Database:

- **AWS DynamoDB** manages the storage of user queries and responses. It offers a fast, flexible, and scalable NoSQL database service, which is essential for storing and retrieving large volumes of user interactions efficiently.

6. Monitoring:

- **AWS CloudWatch** is used to monitor the application's performance and logs, ensuring the system runs smoothly. It helps track API Gateway and Lambda function metrics, logs errors, and monitors SageMaker inference requests.

7. Scaling:

- **Auto-scaling** policies are configured across AWS services (e.g., SageMaker, Lambda, and DynamoDB) to automatically scale resources based on user load. This ensures the system can handle varying numbers of requests, providing both flexibility and cost-efficiency.

b. Technical Architecture Diagram

The system's technical architecture can be represented in a diagram, illustrating the flow from the user query to model inference and response generation. Below is a high-level description of the architecture:

1. User Interface (Frontend App):

- A web-based interface (potentially hosted on **AWS Amplify**) where users input their medical queries.

2. API Gateway:

- **AWS API Gateway** manages incoming HTTP requests from the frontend. It triggers **AWS Lambda** functions to process the requests.

3. AWS Lambda:

- **AWS Lambda** handles the logic for processing the query, retrieving relevant medical passages, and invoking the fine-tuned **T5 model** hosted on **AWS SageMaker**.

4. AWS SageMaker:

- **AWS SageMaker** hosts the fine-tuned **T5 model** that generates responses based on the retrieved medical passages and the user query.

5. Amazon S3:

- **Amazon S3** stores the medical datasets and other relevant files required for the model's operation (e.g., pretrained embeddings and model weights).

6. DynamoDB:

- **AWS DynamoDB** manages the storage of user queries and generated responses, providing efficient and scalable database service.