2.Proposed software architecture

2.1 Overview

The MyAiTherapist system software architecture is a comprehensive document that describes the layout and component parts of the system. It acts as a blueprint for the design and development of the system, directing the team's technical choices and ensuring that all of the system's components function properly.

The software architecture contains details about numerous classes and modules discovered during the project's analysis phase. To make the system simpler to comprehend and use, these classes and modules—which serve as the system's building blocks—are arranged logically and hierarchically.

Software architecture defines the structure and organizing principles of the system by describing the relationships between subsystems and the technical choices made up to this point. This covers the technologies and frameworks employed in the system's development, as well as the design patterns and best practices that were used.

The software architecture gives the development team a roadmap and specifies classes and modules for implementation, ensuring that all required parts are present. By offering this level of detail and direction, the software architecture helps the MyAiTherapist system's development and deployment go smoothly.

2.2 Subsystem decomposition

To achieve effective design and development for MyAiTherapist web application, it will be splitted into several subsystems based on functional and non-functional requirements besides technical considerations. Each of these subsystems will be responsible for specific tasks and functions. Subsystems are going to work together and provide a cohesive and efficient overall system.

**Front-end subsystem** will be in charge of the User Interface of the MyAiTherapist. It'll provide a user interface that is easy to use. We'll use Nuxt 3 built on a javascript framework Vue.js to implement server-side rendering. The application will provide a smooth and responsive experience for users. Front-end module will handle all engagement with the user. It'll communicate with the backend subsystem via APIs and represent the changes to users. AWS Amplify will also be used by the front-end subsystem to make it easier to integrate AWS cloud services, such as authentication, authorization, and data storage. Through APIs, it will communicate with the backend subsystem and manage all user interactions, including input and output. Through AWS Cognito, user authentication and authorization will be managed.

**Backend subsystem** will be responsible for the business logic and handle the data management. This subsystem consists of AWS cloud services, which are Lambda for functionalities, DynamoDB for database and S3 for storing static files, in order to fully scalable and well founded infrastructure for the system. This system will provide APIs to speak with the front-end subsystem and other external systems as well. It'll make sure the integrity and security of the data we store in the system.

**The security subsystem** is in charge of the web application's security and privacy, which includes data encryption, access controls, and user authentication and authorization. To offer strong security measures and guard against unauthorized access, it will integrate with AWS Identity and Access Management (IAM).

**The monitoring and logging** **subsystem** will be in charge of keeping track of the web application's performance and availability as well as gathering and storing logs for troubleshooting and analysis. With real-time visibility and insight into the web application's behavior provided by AWS CloudWatch and AWS X-Ray, the team will be able to quickly identify and resolve any potential problems.

**The therapy analysis subsystem** will be in charge of using AWS Rekognition to analyze the therapy sessions in order to draw conclusions and spot trends. In order to enhance the therapeutic process and support wise decision-making, it will give therapists and patients data and reports.

2.3. software hardware mapping

AWS S3: This service will be used to store and access user data, such as notes from therapy sessions and analysis findings.

User data, such as patient lists and analysis results, will be stored using the AWS DynamoDB service.

AWS Lambda: This service will be used to carry out the AWS Rekognito-based analysis of therapy session records.

For the web application's user authentication and authorization, AWS Cognito will be used.

To make the development and deployment of the web application easier, AWS Amplify will be used.

AWS CloudWatch: This service will be used to keep an eye on the performance and availability of the web application and the services that are connected to it.

Debugging and troubleshooting the web application and its related services will be done using AWS X-Ray.

The web application will manage access to the various AWS services it uses by using the AWS Identity and Access Management (IAM) service.

AWS API Gateway: This service will be used to create, publish, and secure APIs for the web application.

Nuxt.js: This framework will be used to build the server-rendered components of the web application using Vue.js.

**2.4 Persistent data management**

We'll be utilizing a mix of Amazon DynamoDB and Amazon S3 for persistent data management.

The documentation of therapy sessions and the findings of the analysis produced by Amazon Rekognition will both be stored in DynamoDB. Each record in a DynamoDB table that stores the records of therapy sessions will correspond to a single session. The name of the patient, the date and time of the session, and any notes or observations made by the therapist will all be attributes of the table, with the session ID serving as the primary key. An additional DynamoDB table will be used to store the analysis results produced by Amazon Rekognition. This table's primary key will be the session ID, and its attributes will include the analysis results and any pertinent metadata.

The video recordings of the therapy sessions and any accompanying paperwork or media files will be kept on S3. S3 offers a dependable, expandable, and safe storage option for these files. We will also use Amazon CloudFront, a content delivery network (CDN), to deliver the video files to users with low latency in order to maximize their performance.

We will use AWS Identity and Access Management (IAM) to regulate access to the DynamoDB tables and S3 bucket in order to guarantee data security and integrity. The data won't be accessible to anyone besides authorized users like administrators and therapists.

The performance and availability of the DynamoDB tables and S3 bucket will also be monitored using AWS CloudWatch, and alerts will be set up for any potential problems. AWS X-Ray will also be used to track requests and troubleshoot any data management system problems that may occur.

**2.5. Access control and security**

We will use Amazon Cognito for user authentication and AWS Identity and Access Management (IAM) for access control to ensure the security and privacy of the therapy session records and analysis results.

User account creation, management, and the procedure for verifying users' identities when they access the application will all be handled by Cognito. Access to the application and viewing the analysis and therapy session records will only be available to authenticated users.

The DynamoDB tables and S3 bucket that house the session logs and analysis results will be protected by IAM. IAM will enable us to specify which users or groups have access to which resources and at what level (e. g. read-write, read-only, or both). This will guarantee that only authorized users can access the data and that they can only carry out the authorized tasks.

We will also use Amazon Amplify to safeguard against widespread web vulnerabilities like cross-site scripting (XSS) and cross-site request forgery (CSRF), as well as to secure the communication between the web application and the backend services.

Last but not least, we will use AWS CloudWatch to keep an eye out for any potential problems with the security and access control systems and to set up alerts if necessary.

2.6. Global Software Control

We'll be utilizing a number of AWS services for global software control to make sure the web application is scalable, dependable, and responsive.

The web application or other AWS services will trigger events, and Amazon Lambda will be used to carry out backend logic in response. We won't have to provision or manage servers in order to respond to these events with code thanks to Lambda. This can enable us to easily and quickly scale the application to accommodate demand.

Amazon CloudWatch will also be used to set up alerts for any potential problems as well as to track the application's availability and performance. This will enable us to spot issues early on and fix them before they negatively affect the user experience.

Finally, in order to track requests and troubleshoot any application problems that may arise, we will use AWS X-Ray. With the help of X-Ray, we will have a thorough understanding of the request path and be able to spot issues early and fix them.

We can make sure that the web application can support a large number of users and requests without any downtime or performance issues by using these services for global software control.

2.7 Boundary conditions → burayı paraphrase et direkt aiden aldım

2.7.1 Initialization

During the initialization phase, the web application will perform the following tasks:

Establish a connection to the DynamoDB tables and S3 bucket for storing the therapy session records and analysis results.

Load the list of authorized users from Cognito and ensure that only these users are able to access the application.

Load any necessary configuration values and settings from a configuration file stored in S3.

Load any necessary libraries and dependencies required by the application.

2.7.2 Termination

During the termination phase, the web application will perform the following tasks:

Close the connection to the DynamoDB tables and S3 bucket.

Save any necessary data or state to persistent storage (e.g., DynamoDB, S3).

Release any resources that are no longer needed (e.g., open connections, allocated memory).

2.7.3 Failure

In the event of a failure, the web application will perform the following tasks:

Log the error and any relevant details to a log file stored in S3.

Notify the appropriate parties (e.g., administrators, developers) of the failure via email or SMS.

Attempt to recover from the failure by re-initializing any affected components or services. If recovery is not possible, the application will shut down gracefully and alert the appropriate parties.