AWS services that can be used for the project:

Element	**AWS Service**		Free Tier (source link)
User Login/Authentication (Parents login with email/password)	AWS Cognito		https://aws.amazon.com/cognito/pricing/
Database (Cloud) (stores profiles from sync with SQLite/RoomDB)	Amazon DynamoDB		https://aws.amazon.com/free/database/ This always free service is on the **Free and Paid plan**. Use your credits to evaluate beyond these monthly limits: 25 GB of storage 25 provisioned Write Capacity Units (WCU) 25 provisioned Read Capacity Units (RCU)
QT Robot access to database (generating access keys)	AWS IAM		Included in the AWS Free Tier
Communication from robot and app with cloud DB	API – GraphQL schema file (open-source query language for APIs)		Open-source

Option: use a developer tool to get working and manage services above easier	AWS Amplify (with AppSync – created automatically)		https://aws.amazon.com/amplify/pricing/

AWS Amplify simplifies app (or web) development by handling backend configuration, allowing for faster development and deployment. It can work with other AWS services and simplify tasks such as adding user login, storing data, hosting an app, automatically handling the backend, while allowing to focus on building the app itself.

Why to use: saves time, we can focus on user experience part more, can work across platforms, provides more scalability

Main Features: authentication service, serverless architecture, data sync and storage (integration with Amazon S3 and AppSync), offline sync, CI/CD, real-time data and notifications system.

AWS Cognito handles user authentication, sign-up, sign-in, MFA and access control for mobile app, and user synchronization across devices. Cognito offers integration with social identity providers (Facebook, Google, and Amazon) so users can sign in using their existing accounts.

Why to use: it will help us to focus on the app logic rather than login system complexity. Enhances app security. It can be easily integrated with other AWS services.

GraphQL is used to allow app on the tablet and physical robot to talk to the Cloud (DynamoDB in our case). Without it, if robot asks for the child's data from Cloud, it just dumps the entire file and wastes data and time for things it does not need. With it we can specify the user to return based on condition (e.g., return only name and sound sensitivity level). We write GraphQL code in Amplify as a blueprint (schema.graphql file, lists every data type and their relationships), blueprint is deployed to so that Amplify reads(via AppSync) it and builds DynamoDB tables according to it.

Note: It is like a swagger.yml OpenAPI file we used in COMP2001.

Why we are most likely do not need AWS S3:

S3 is mostly for files storage, while DynamoDB is for text data type. Our images in app (e.g., avatars) are static and live inside the Android app (R/drawable/). So, they can be installed on the device when user downloads an app. And it will work offline. And it is a lot cheaper, faster to find and return matching TEXT record in cloud than finding FILE data. For privacy, we store face images as an embedded file locally, and do not store them in the cloud. Thus, we won't need S3 cloud bucket to store images. We would

need S3 bucket if we, for example, in case we generate pdf reports in app and send them to dentist office via cloud sync.

Face recognition

Face recognition happens 100% on the tablet. Do not send face images to AWS cloud.

Face Recognition library can become a QR code scanner temporarily for test purposes.

Possible user recognition flow without camera (an idea only, abstract)

Child logs into profile via avatar -> app syncs profile to AWS DynamoDB

App can generate QR code on the screen containing user ID -> robot's camera always looking for a QR code -> reads user ID from QR code -> sends request to DynamoDB (userID – any match? Who is that?) -> pulls user info and user preferences? (e.g., username, sound off/on) and then acts based on those.

1 - Android Tablet (app)

Possible tool set: Android Studio (Kotlin, Jetpack Compose, RoomDB or SQLite(for profiles), CameraX, AWS Amplify SDK(+ Amplify DataStore), Jetpack DataStore (instead of SharedPrefferences)).

Parent Login: AWS Cognito via Amplify auth command. Parent can create an account with email and password (and enable MFA) to manage multiple children's profiles.

Child Profile: parent can enter child's name and settings (e.g., sound, animations, dark mode, colour scheme) and potentially take a photo of the child.

Local Storage for face imagery: app converts child's photo in a secure file format and saves it on the device (android/ios internal storage), but not the cloud!!!

How child can log in: takes a tablet -> camera captures video piece -> face library locally analyses frames and matches the face to the locally stored file. If matched -> the app retrieves user profile from user entities stored in RoomDB (or SQLite) file and logs the child.

Offline data storing and cloud sync: RoomDB (or SQLite) saves all mood, rewards, settings on the profile locally first. (In case app goes offline). Then Amplify DataStore pushes local changes in RoomDB (or SQLite) via the AWS AppSync interface to cloud AWS DynamoDB whenever Wi-Fi connection is available.

Recognition by QR code: when child meets the robot in dental office, the app can generate the QR code (full screen, brightness up) containing the child's profile user ID.

2 – The Bridge to AWS Cloud

Main tool set: AWS Amplify CLI, AWS AppSync (API layer between GraphQL and Amplify), DynamoDB, IAM

Database: DynamoDB stores a COPY of child profiles and mood, hygiene, reward logs, etc.

Cloud Security: we created and specified a new IAM User (QT Robot - physical). It provides robot with permissions to **READ only** the DynamoDB, but not edit or delete it.

Note: DynamoDB (most likely) will have **no logic** (e.g., calculating child's age, deciding what robot is going to say, translate text, push notifications to the robot screen, etc.), it just holds the fetched data. Robot just runs a loop, e.g., every 5 sec checking if there is a data for it to process.

3 – The Physical QT Robot (potentially QR code scanner, or face scanner later)

(Due to lack of expertise in Robotics, suggestions in these sections are generated by AI and are subject to further review and changes)

Main toolset: Python, OpenCV, Boto3 – AWS SDK, QR Reader (e.g., PyZbar).

The robot always runs python script using OpenCV to continuously analyse video feed from its head camera. When it detects QR Code generated on tablet in full-screen, PyZbar decodes it into user ID. Then the script uses Boto3 to query DynamoDB for matching that user ID.

If matched: Robot pulls/downloads profile (child's name, sound settings, etc.) -> Robot adjusts its internal volume -> Robot triggers text-to-speech to say: "Hello [child's name]!" -> the script pauses for 5 seconds after greeting to prevent looping.

4 – Possible ways Dentist can view the data

Option 1: create a Dentist Practitioner view for the app.

Login system is still delegated to AWS Cognito with **mandatory MFA.**

Child's profile local storage is synced to cloud. Dentist's tablet pulls data from the Cloud.

Changes to make: authorize dentist group to pull data created by other users using GraphQL (schema file?)

Dentists logs in via Cognito (to the app, email/password) -> app detects user as a part of the Dentists group -> app shows a search bar or patients chart instead or "Select profile stickers".

Optional: dentist can scan QR code from the child's tablet to get user ID and pull that specific profile and associated data from DynamoDB

Option 2 – No UI (most optimal for time): just show the functionality working via AWS Dashboard

Use AWS Console as an admin view:

Open laptop with the AWS DynamoDB console -> Explore items -> it lists the live database view (in raw JSON text) of the child's anxiety levels/hygiene routine/reward system progress. It has a text "Dentist front end is not built yet, but the functionality and architecture works".

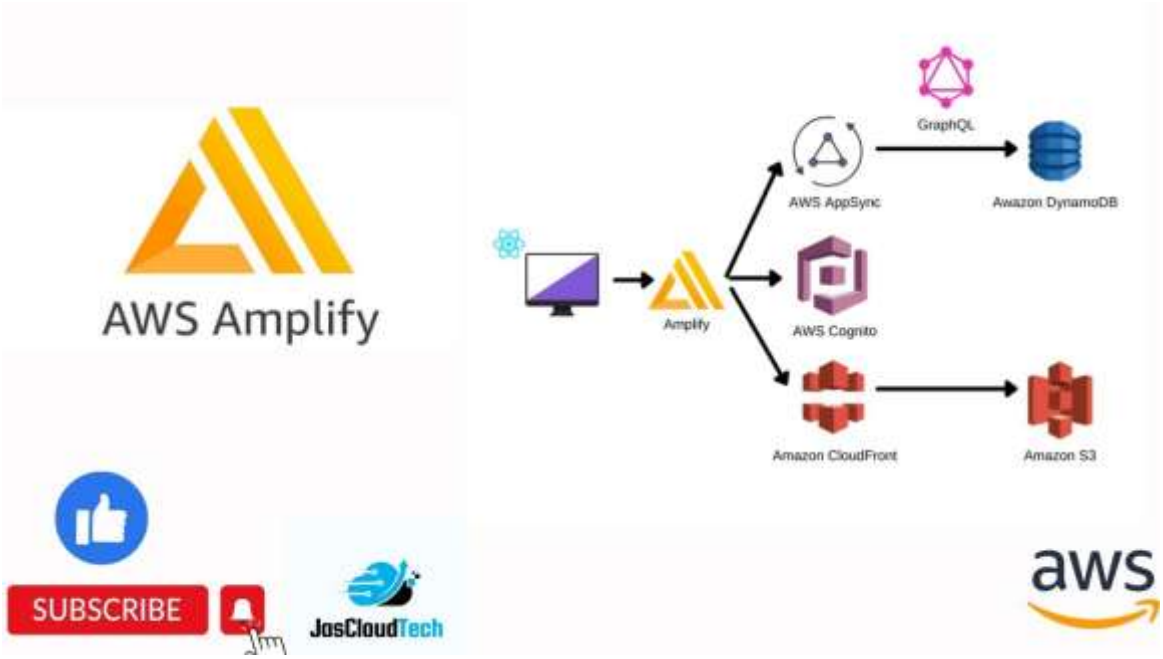
Reference links

AWS Amplify: https://www.geeksforgeeks.org/devops/introduction-to-aws-amplify/

https://aws.amazon.com/amplify/

AWS Cognito: https://www.geeksforgeeks.org/devops/what-is-amazon-cognito/

Video about AWS Amplify: https://www.youtube.com/watch?v=TmDWNSJVuvM



AWS Amplify Console for App Development: https://medium.com/codex/accelerating-mobile-app-development-with-aws-amplify-console-a-streamlined-approach-for-success-0b352f0db05c

Amazon Free Tier: https://aws.amazon.com/free/?nc2=h_pr_ft