Functionality

Amplify hosts the front end, backend is deployed on Elastic beanstalk and is connected to RDS to store data.

Options

There were choices to be made for the version of Amplify, the service that we are using to host our front end and a choice to be made for storing our data, where we considered RDS, but there is another service called Aurora, so we will have to figure out which one works better for our webapp.

Amplify has a new version, gen 2, which supports typescript and that is something to be taken into consideration. It supports full stack deployments from git for web-applications using typescript. It also has support for implementation of custom pipelines. However, this service is still in preview mode, and we are not super sure if everything will work as well as the previous version of Amplify. As for Gen 1 of Amplify, we have all the regular functionality provided by Amplify like support for multiple languages. Although our code is written in typescript for the frontend, we will still be able to use gen 1 of Amplify.

Next, we had to make a decision on whether we wanted to use RDS or Aurora since both of them have their benefits. Aurora is a relational database that is compatible with MySQL and PostgreSQL, it however stores this information in the form of InnoDB. RDS on the other hand is simple to set up and handles multiple databases like MySQL, PostgreSQL, MariaDB, Microsoft SQL server, Oracle and it also supports Amazon Aurora, so it makes perfect sense to use RDS as our service. Also, aurora does not have a free tier or trial, so we do not want to spend resources on it as of now.

Things to take into consideration.

1. Upon learning how to deploy the backend on AWS, I found out that it is better to deploy the backend by using a service that helps in containerization, such as Docker. This not only makes things easier to manage later on down the line, but docker also helps in roll backs to previous versions of the applications. When I further started learning more about docker, I found out that there is a service called Kubernetes. This is used for managing containerized services, however upon further research, I also found that AWS has services that deal with containerized services. The second way will be better since we won't have to deal with Kubernetes and can directly work within AWS.
2. We need to learn more about how we can configure our AWS. We have 2 options: Use Amplify for the front end and Elastic beanstalk for our backend. Our second option is to use just Amplify for both front-end and back end. This option is slightly unlikely to happen since using Elastic beanstalk is a better option when it comes to scalability. Our third option is to use just Elastic beanstalk. To decide which will be a better option, we will have to carry out some tests to see what configuration of the deployment makes sense based on scalability, performance, and cost(once we are done with the free trial).