Technical Design Document: Pokémon Deck Building App

Author: Anna Keramaty

Overview

The goal of this project is to build a web and mobile browser-based app where Users can build and save Pokemon decks that they can return to over time.
Main requirements: The app is web and mobile browser based, the app should support scale to 100 million daily active users, there should be image hosting, analytics and metrics should be tracked on usage, user can build several decks, up to 10 total, each with up to 5 cards, user needs to be able to save the deck for future use beyond the current session, user needs a way to toggle between decks

Architecture

Considerations
The main considerations taken into account when designing the architecture of this app is:

- Web/Mobile Web UI Only (no mobile app).
- Users should be able to login and create and update profiles.
- Pokemon cards should be read, updated and deleted.
- Reading and creation of user owned Pokemon decks.
    - We should expect the most reads/writes and updating of data to happen here.
- We must be able to access what current user sessions and decks are being used.
- Analytics must be tracked.
- We need the ability to easily scale as the application use grows.

Authentication
Users will login in with a username/password and create an account. Internally we will use Oauth to authenticate and authorize user access to different services and functionality. Authentication sessions will be tokenized and saved as browser cookies with a session logout TTL.

Architectural Layers
There will be four main layers for this application:

- Presentation layer: manages the web and mobile browser UIs. Session data will be used at this layer to manage a user's login status and current status of open decks.
- Logic layer: This backend layer will power the business logic for the three main services of the app. These services include Profiles, Cards and UserDecks.
- Data Logic layer: This layer will manage the CRUD of the three main services (Users, Cards, Decks) and also will manage processing analytics events and any updates to media.
- Data Storage: This is the database layer for the app.

External Gateway
Above the logic layer there will be an external API Gateway. This gateway will manage the following for requests coming from the presentation layer:

- Authentication (Who is making the request)
- Authorization (Do they have access to the service and method requested
- Rate limiting

It will then route requests to the appropriate service in the logic layer and will include identification information on the user who is making the request. Services within the logic layer will then do additional processing and filtering based on the user and their access to the requested data or actions requested to be taken.

Scalability Considerations
There will be a few solutions implemented to help support scalability in regards to user growth.

- Load Balancer: The load balancer acts as the starting point of the system, directing incoming traffic to the presentation layer.
- Caching at the Data Storage layer: This reduces the load on the database and improves performance.
- Horizontal scaling: Support of independently scaling the microservices within the data and logic layers to meet demand.
- Auto scaling of the presentation layer: This enables the system to handle peak loads efficiently by allocating more resources when necessary.
- May consider in the future to scaling to multiple data centers if usage increases to 100 million daily active users

Analytics
Analytics will be stored in a separate data warehouse. Actions within each service will trigger their own analytics events that will be pushed into a queue that will be read and processed from the data logic layer. This layer will then push events to the data warehouse where further data analysis can be done. The database will also be ETLed to the data warehouse when possible with considerations for GDPR and PII data protection concerns.

Media Storage
The system also integrates with a Media CDN, which stores and distributes media content.

Cloud Storage
A cloud storage solution will serve as a backup for both the data warehouse and the Media CDN.

High Level Architecture Overview Diagram