Tech Test - Platform Engineer - Credit Tracking and Billing

Test Purposes

This should test:

- AWS platform knowledge
- C# or Golang
- General code quality
- Logical thinking
- DDD

Test

✏️ This scenario is realistic but not real - details have been fabricated just for the purpose of creating a useful exercise

The scenario is that Xapien want to get on top of their billing - we want our customers to pay us what they should, and not to use our system more than they've paid for.

Our customers use Xapien to run reports (also called "enquiries"), and our pricing generally works around how many reports we let them run every month.

We generally have customers on one of four plans:

- Ultimate: 1000 reports per month
- Enterprise: 500 reports per month
- Basic: 100 reports per month
- Lite: 20 reports per month (we're probably going to phase this one out soon!)

We also let people trial Xapien for a limited time: they can have 10 reports over the course of a couple weeks but then they aren't allowed any more until they've signed up for a full plan.

We want to build a system that tracks the usage of each customer and:

1. Stops them running more than their allotted reports
2. Sends us a notification when a customer is on track to hit their monthly limit - so that our customer success team can reach out and try and sell an "uplift". This warning notifications is probably more of a "nice-to-have", but at the very least the system needs to send us a notification when the limit is reached.
3. Allows them to configure per-user limits - say if I'm an organisation with 500 reports a month and 10 users, I might reasonably set a 50 user/month limit to make sure one employee doesn't use them all.
4. Allows us to top up a particular customer's monthly report credits at our discretion. These will expire at the end of the month. We're definately going to need this before releasing because otherwise customers will run out of credits but we won't be able to help them.

You should assume that we already have organisation and user identifiers, and enquiry (report) ids.

So we'd like you to implement 4 REST API endpoints:

1. **Set Plan** - this will accept an org identifier and the Ultimate/Enterprise/Basic/Lite "plan type" and store it on the backend against that org
2. **Set Per-User Limit** - this will accept an org identifier and a limit number, and should store that as the per user limit for that org on the backend
3. **Use Report Credit** - this will accept the org identifier and the enquiry identifier being billed. It should check that this enquiry is allowed to be run (by checking credits remaining on the org and the user) and update backend state accordingly. A HTTP success code should be returned.
   If the enquiry is not allowed to be run, a HTTP error code should be returned.
4. **Top Up Customer Account** - this will accept the org identifier and the number of report credits to add

Please use AWS stack and either C# or Golang as this aligns with our current platform architecture.

For simplicity:

- “per month” should be taken to mean “per calendar month”
- authenication and access management should be considered out of scope
- “send us a notification” can be simulated - it can even be a method call that does nothing

We would like to see how far you get on this in 4 hours, but this is not a hard limit. If you cannot complete your solution that’s fine, but make some notes on what else you’d do. In the manner of Agile development, we recommend you start with the core functionality first.

The solution will be evaluated for:

- Code quality, and efforts taken to ensure code equality
- Use of DDD in backend code
- Appropriate use of various cloud technologies
- Correct implementation of logic

We might ask you to improve your solution in your follow-up interview