Exercise

You are asked to design a system that connects to users' mailboxes (emails). Your goal is to expose an API, with endpoints for a client (for example an iOS app) to:

- Endpoint 1:
  - Authenticate with the user's email provider. This endpoint should accept an `email` as one of the inputs. For this exercise, assume all emails are formatted either as `*@gmail.com` or `*@microsoft.com`
  - For example /connect/email?email=pierre@gmail.com
  - This endpoint should redirect to the OAuth flow of the email provider.
- Endpoint 2:
  - Given a timestamp as an input, and for a given authenticated user, return simply a boolean (true, false) depending on if the authenticated user has received at least one email since timestamp.
  - For example /emails/search?since=timestamp&token=current-user

Information

- The only two email providers we are interested in are: Microsoft 365, and Google.
  - You can implement either one of the two email providers (Microsoft, Google), or both. If you implement only one, your code should take into account that both providers will be implemented in the future.
- You are responsible for setting the authentication system you want between the API and the client.
- You decide what libraries to use, or not..
- You can use the programming language of your choice. Provide an explanation for why you selected this language.
- If you don't want to expose a REST API, feel free to use another communication framework (graphQL, gRPC) between the client and the app, but please keep the same business objectives.

Deliverables

- [ ] The API is deployed on a publicly accessible URL.
- [ ] You send us your code source on Github.
- [ ] Documentation that you would give to the client team to access your API. You decide how and what documentation you want to pass along to the client team.
  - [ ] We also document quite a few things internally for our own benefit. A readme.md with documentation of the architecture, decisions made, and challenges is quite helpful for us to understand your thought process.
- [ ] Tests. We will pay attention to how you are using tests in your code. Write as many tests as possible.

Bonus

If you have more time, you can expose another (3rd) endpoint that simply returns an integer. This integer is either the approximate or exact cost (in dollars) of how much it will cost to put all the emails received by the authenticated users in the last 24 hours into a Large Language Model, specifically the model `gpt-4` from OpenAI.

- For example /cost?token=current-user

Timeline

Take as long as you like. And do it when you have the time. We expect that this challenge can be done in as little as 4-5 hours, but you could also spend a week on it.

Find a balance: Perfection comes at a high cost, so make reasonable decisions where to invest your time and how much time to invest.

Not choosing to spend time on something can be a positive indicator of good decision making. Find the details that are worth paying attention to.

If you're in doubt about anything, **ask**.

Thanks!