Assignment 4

Describe the concept of throttling in APIs. 15 points.

Let us remember that an AP (Application Programming Interface) is the intermediary between the user and the application.

Now, **throttling APIs** is the process of limiting how many API requests a user can make in a certain amount of time; when a webpage has a lot of users at the same time we need to make sure that they all get the same response time. Without this, if a user tries to send the same request several times, it may slow down the system and its performance; it would also make the website weak if it comes across a hacker that tries to send multiple requests to make the site vulnerable.

This technique is used to manage resources at the server or network level, the main goal is to ensure that the API can handle all the traffic it receives.

The steps that a throttling process goes through:

1. A user calls a web service through an API.
2. The API throttling checks if the current request exceeds the allowed number of API calls.
3. If the request is within the specified limits, the API works as usual and does the user's job.
4. If the request exceeds the limit, the API returns an error to the user.
5. The user will have to wait for a pre-agreed time period or pay to make any more API calls.

Advantages:

- It helps prevent system degradation, if a webpage has a lot of users sending a lot of requests, servicing all those API requests will slow down the system and affect its performance.
- An API throttling system acts as a gateway to an API, so it adds a security layer.
- It helps save money, you may have a number of free API requests per hour, but if you have a lot of requests and exceed that number, you would need to pay for that.

Disadvantages:

- One challenge of this is that if the website is worldwide if the user makes consecutive requests this could be forwarded to different servers and overlap if there is not a good synchronization system.
- If the system reaches the limit of requests no more processes will be reached until the time period expires.

Describe the concept of pagination in APIs. 15 points.

Pagination is a process that is used to divide a large dataset/archives into smaller parts (pages). All Square API endpoints that support pagination also support a limit field that the application can use to indicate the page size, in other words, they are the number of items to return in the response.

When we ask a request to an API server it can generate thousands of entries generated from JSON files, this is actually a bad practice since it generates a drain of resources and time lost and doing pagination helps us query databases more easily.

Types of Pagination:

Offset Pagination

It is the most common type of pagination. This uses the "**limit**" (aka size) and "**offset**" (aka page) commands already present in the SQL library as query parameters, for example:

```
GET /<parameter>?offset=0&limit=10
```

Here the **offset** command tells the server the number of items/parameters that should be skipped, while the **limit** tells the number of items /parameters to be returned.

Keyset Pagination

Keyset pagination uses the filter values of the previous page to determine the next set of items, in other words, it uses a key to select by which data it can be sorted. Then, the key is used as a reference to split data into pages, for example:

If data is sorted by user ids, then the key will be `since_user_id`. If the limit is set to 10, then the user will have to specify `since_user_id=10` to get results from 10 to 20.

Seek Pagination

Seek pagination adds the queries `after_id` and `before_id`, it returns consistent ordering even when new items are added to the table.

The only downside to seeking pagination is it can be challenging to create custom sort orders.

Describe the concept of a callback function. 15 points.

A callback function is a function passed as an argument into another function, this allows a function to call another function, this can be separated into two different types of callbacks:

Synchronous callback: it happens right away after its creation; this is not commonly used for synchronization or for delegating work to another thread.

Asynchronous callback: it happens at a later point in time; this is more common in I/O operations or for event handling.

A disadvantage/problem of this A major concern here is the management of privilege and security: while the function is called from the operating system, it should not run with the same privilege as the system.

In the following example, we can see a function **greeting** which then is used until the next function as a parameter:

```
function greeting(name) {
alert(Hello, ${name});
}

function processUserInput(callback) {
const name = prompt("Please enter your name.");
callback(name);
}

processUserInput(greeting);
```

Describe the concept of cold start in AWS Lambda. 15 points.

When the Lambda service receives a request to run a function with Lambda API, the service first prepares an execution environment (downloads the code for the function to be stored in an internal Amazon S3 bucket). It then creates an environment with the memory, runtime, and configuration specified. These two beginning steps are known as a **Cold Start** the good thing is that it won't give you any extra costs the time Lambda to prepares the function to be up and ready to go, but it will add latency to the overall invocation duration.

After the implementation completes, the execution environment is frozen (that is where the name comes from), the Lambda service retains the execution environment for an unknown period of time. During this time, if another request arrives for the same function, the service may reuse the environment.

A solution to this may be using **Provisioned Concurrency,** this lets you build scalable serverless applications with predictable latency. You can avoid cold starts and startup latency issues for your Lambda functions, in other words using serverless functions can help control random uploads of traffic without increasing latency.

Describe each HTTP method. 15 points

An HTTP method is used to indicate the action to be performed for a resource.

`GET` is used to retrieve information from the given server with an URL, they should only retrieve data.

`HEAD` asks for a response identical to a GET request, but without the response body (it transfers the status line and header section only).

`POST` is used to send data to the server using HTML forms.

`PUT` is used to replace all current representations of the target resource with the request payload.

`DELETE` is used to delete the specified resource.

`CONNECT` is used to establish a tunnel to the server identified by the target resource.

`OPTIONS` is used to describe the communication options for the target resource.

`TRACE` is used to perform a message loop-back test along the path to the target resource.

`PATCH` is used to apply partial modifications to a resource.

Describe how you can automate the deployment of a static website to S3. 15 points.

Here we can use **AWS CodePipeline**, this is a continuous delivery service. It is used to automate the process of building, testing, and deploying software into a testing/production environment, this creates a CI/CD pipeline. Keep in mind that this process is recommended for displaying read-only content. It isn't recommended for collecting or transferring sensitive information, because Amazon S3 uses the HTTP protocol.

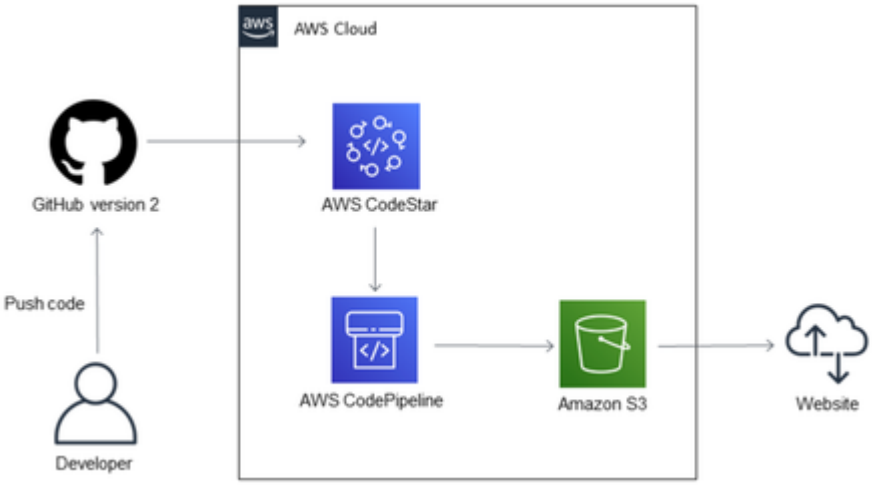
Once we have a static website with a bucket policy we can create a GitHub repository, with that we can the pipeline by writing up the pipeline name, a service role (a role that an AWS service assumes to perform actions on your behalf, it performs backup operations on your behalf), and an artifact store (choose the default location option); after that, we add the GitHub repository location.

To do the pipeline we use the command:

```
aws codepipeline create-pipeline --pipeline
```

As soon as the connection is successful we can select the Deploy provider (Amazon S3) and the bucket we are working on and extract files before deploying (this is because the pipeline compresses the artifact).

With this when a **Git Push** is made into the repo, CodePipeline should pick it up and deploy those additions/updates and deploy that on the S3 bucket, automatically. Keep in mind that this pipeline will cost only $1 per month and charges only if a deployment happened.



Read the Real-world Engineering Challenges #8: Breaking up a Monolith article and write a summary and opinions about it. 10 points.

Khan academy decided to move to a new language when Python 2 (2019) ended his job; they had to migrate 100% of Python 2 code to **Go** language and **GraphQL gateways**, and a query plan is generated that includes data from multiple backend services. Now, this is now something that you can do from one day to the other, we are talking of millions of coding lines, that is why they had to split the code into two phases:

Phase 1: Minimum Viable Experience (MVE): since there was an already existing product they had to decide what elements were crucial so that Khan Academy did not lose its essence, they had to consider:

- Content publishing
- Content delivery
- Progress tracking
- User management

This process took 2 years (2021), and made that 95% of traffic flowed through the new code where 32 services were built.

Phase 2: endgame: This part took 1.5 years to be completed, fewer developers were working on the project and more new features were built, as well as migrating the existing ones.

A downside of this migration was especially with the engineers, it can be a hard job to keep migrating something already existing instead of working on creating something new, something positive about this is the learning experience they get from working with new technologies.

I think that in this case, it was a migration that MUST have been done, but as engineers, the way to avoid this last-minute implementation is by doing research day to day into what new technologies are being made and how to make a product better.

It is always important to keep in mind that deadlines are important, this helps the team to organize and keep track of what work is done.

Tests are important, they help us see our vulnerabilities, In the article, they said that when they only had Python2 “some areas were well-covered, while others were not”, this is not a good practice, and had to do "side-by-side" testing when the migration started.