Code Assignment Unity

Adrianna Śliwak
13.11.2021

General Questions:

1.) Your team has just deployed a new build. The application starts crashing because it's running out of memory after several hours. The team suspects a memory leak has been introduced in one of the apps dependencies.

a.) How would you help the web team diagnose and fix the problem?

b.) How would you stop issues like this making its way into production?

1a):
Speaking about diagnosing any kind of error is always very vague. Afterall, each app is different, works differently and behaves differently.
Diagnosing should be farely simple, assuming we already have a suspect in this specific dependency. We can either inspect the logs, manually debug code or in drastic cases analyze inputs and outputs of the api.
There is usually a way to put restrictions on the ways dependencies are being called, which can solve our issues. Making sure that memory is being properly allocated and that unused parts are being disposed is a key to avoid memory leaks.
2b):
Every piece of software that is about to be shipped to production should "live" for sometime in some WIP environment. Example could be:
Develop branch is ready to be shipped (tested by devs)
Ship it to Staging branch, allow QA department test it
After having greenlight from QA, ship to production and monitor using monitoring software and a good logging system.

2.) Describe some of the advantages and disadvantages of SQL and NoSQL data stores, and the general use cases you think they each fit.

Sql:
Good for transaction and highly coupled data
ACID compliance

NoSql:
Better scalability
Better for big amounts of unstructured

Example use case can be using Sql for analytics database, and non sql for storing documents.

3.) Describe the workflows, ecosystems, and technology you would use/create to ensure the dual goals of GDPR/privacy compliance and data accessibility for business/application needs.

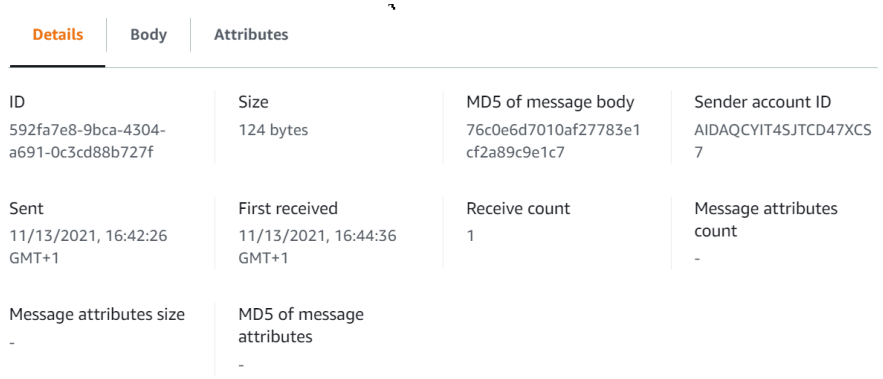
That widely depends on the type of data and the way the data is being used.
There can be db scheduled operations, for periodical deletion of so to say expired data.
There can be software that manages data using access keys with different permission, to ensure nobody pulls data that shouldn't be seen.

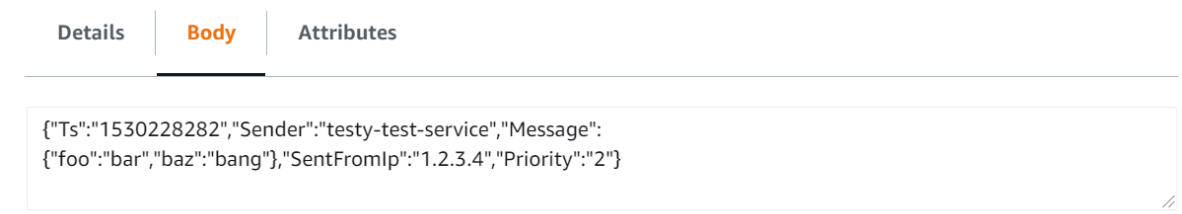
4.) What's your pre and post-deployment checklist for a new application, inclusive of the entire stack (assume the infra/cloud provider of your choice)?

Assuming using k8s and eg github workflows:
Predeployment:
- automatic build and dotnet test (or other command for invoking test, depending on used language/framework)
- validate yaml files
- check if containers can be created
- create and deploy
Postdeployment:
- healthchecks (this one cannot be stretched enough)
- monitoring (datadog can be an example)
- pods monitoring/ CD tool like ArgoCd

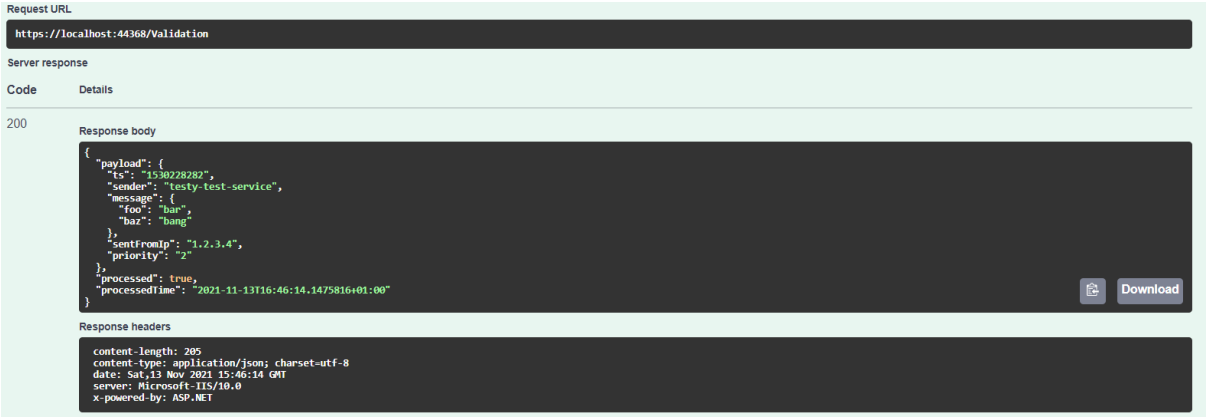
Engineering Excersize:

Presented solution is written using .NET Core (web api using swagger UI). The Message Queue used for this assignment is Amazon SQS.
After testing messages are appearing properly in Amazon SQS:





The application contains hardcoded access info for the mentioned queue (yes, it is a bad practice, normaly I would store them as secrets in k8s, I decided to make it visible, in case you would want to check the queue).
Swagger UI has been properly generated, passable payload results in status code 200:



K8s files are present in the kube folder. Because Amazon SQS is running on Amazon server, deployment files consist only of application deployment.

Scalability

> *Now that you have your service, describe/implement your approach to making it ready for production at scale (100k RPS).*

K8s is very convenient in terms of scalability. In the deployment.yaml file there are two, very important parts for that manner. Number of pods (together with built in k8s load balancer allows to create additional pods that would handle requests) and resources.

```yaml
resources:
  requests:
    memory: "512Mi"
    cpu: "250m"
  limits:
    memory: "2Gi"
    cpu: "1" #Example resources
```

For managing 100k RPS the most important part would be CPU.