Deploying a Serverless Function with OpenFaaS

OpenFaaS is an open-source serverless platform that runs on top of Kubernetes or Docker Swarm. It hides much of the complexity of infrastructure management, so developers can focus on writing small, event-driven functions.

Function Implementation

I started by generating the function scaffold with:

```
faas-cli new hello-openfaas --lang python
```

This created a folder with a `handler.py`, where I added my own logic:

```
def handle(req):
    name = req.strip() or "World"
    return f"Hello, {name}! Greetings from OpenFaaS 🚀"
```

The deployment was done with a single command:

```
faas-cli up -f hello-openfaas.yml
```

At first it felt odd that one command both builds the Docker image and deploys it, but after checking the logs I realised how convenient that is. Once deployed, I could test the function by invoking it:

```
echo "Gesine" | faas-cli invoke hello-openfaas
```

This returned:

```
Hello, Gesine! Greetings from OpenFaaS 🚀
```

Technical Discussion

Even with such a simple example, I got a good impression of how serverless changes the workflow. Instead of worrying about Kubernetes pods or services, I only had to write a few lines of Python. OpenFaaS handled packaging, scaling, and routing automatically.

One interesting part is that functions can **scale down to zero** when not in use. That means fewer wasted resources compared to keeping a web service running 24/7. When traffic increases, Kubernetes automatically scales the replicas back up. This makes the model both cost-efficient and responsive.

Conclusion

Overall, this small demo helped me understand the main value of serverless: faster development, efficient resource usage, and simpler scaling. It felt much lighter than building and deploying a traditional application.