# Goldbach DevOps Challenge

This challenge should demonstrate that you understand the world of containers and microservices. We want to build a simple but resilient Hello World app, deploy it to Kubernetes as a minimal Container and expose it via a load balancer that automatically retrieves the certificates from Let's Encrypt. There are 4 main Challenges:

- writing a Docker container
- automating the build of the container
- setting up the TLS Offloading load balancer
- defining the whole infrastructure as code

# Components:

## Simple http hello world:

Find some example of hello world http app on the internet, in any programming language.

- Modify it to return message **"Hello Goldbach from @YourName"**!!!
- Also modify it to work on port **11000**!!!

## build the webapp container.

Create a Dockerfile that wraps the app. It is important that the resulting container is as small and as secure as possible!

Automate the build and upload of the container. There are various free CI services which you can use for that. Those services can then push the app to any container registry that is publicly usable from Kubernetes.

## TLS Offloading and Load Balancer

This part will require some research. The goal is to have a Kubernetes http://kubernetes.io/docs/user-guide/load-balancer/ infrastructure that is automatically able to get and update Let's Encrypt certificates for any service you deploy. Find a suitable solution that you can easily spin up within your cluster. Document what made you chose that solution over other ones. You should never need to touch a Certificate and all Certificates should be updated automatically. Adding a new TLS service behind the load balancer should be just a few lines of Kubernetes states to be applied. In case you don't own a spare subdomain name to test you can use freenom.com to create one.

- Your app url MUST contain **goldbach** in some part of it!!!
- You app must be externally exposed on port 443 (https)!!!

## infrastructure code & scripts.

Create the necessary Kubernetes configuration to tear up everything. Like everything else put them in a git repository.

## Suggested Approach:

1. research the technologies you plan to use
2. create a simple hello world service. Automate the build and upload of the container.
3. tear up a kubernetes cluster (google cloud offers a free trial, use minikube or choose something else that suits)
4. create kubernetes yaml files and a find a way to tear everything up/down in the correct order.
5. Document your setup, explain eventual drawbacks
6. send back links to all git repos (please use github or gitlab).

Evaluation Criteria:

- is everything automated as much as possible
- is the app container small and secure
- is the infrastructure resilient
- documentation
- customization points marked with **!!!** are a must, missing any of them is an automatic fail!!!

Bonus:

- Diagram of the setup
- everything can be built up and deleted with one command