DevOps Project

You need to create an application which should consist of microservices (3 at least) and they communicate with each other and some (at least 1) or all of them should connect to any DB(MySQL, Postgres, Mongo, etc). If you are uncomfortable with coding, I have bootstrapped a sample app
https://github.com/kahootali/doctor-appointment-system
You can clone it and then create a private Github repo of your own, and copy and add files there.

Make sure the repo should be private and you add my user `kahootali` as a collaborator to your repo, if it's a public repo you will have 50% mark deduction.

The above repository has a base skeleton for a simple doctors & appointments application, it has

- Frontend
- Appointments: hardcoded appointments
- Doctors: hardcoded doctors

You need to remove hardcoding and use db for it, and have some sort of initial bootstrapping to add some initial values.

You need to submit

- Google Doc link containing all the items below and screenshots of implementation of below steps and actual running of them
- Repo link: **make sure my github user (kahootali) is added as collaborator, else I won't be able to access it and won't mark your project**

Phase 1 (30):

- Need to dockerize the microservices, add dockerfiles for each microservice in their folder
- Build & push image to Dockerhub, share url of dockerhub repos
- Add a single docker-compose file at root path which will start all microservices, networks and db on a single deployment

Phase 2 (30):

Add CI/CD to the repo, can use Jenkins or Github Actions, which will

- Pipeline should run on **Pull Request** & **on main branch**
- **For versioning, you can use the commit hash or semantic versioning or any other way (but dont use latest tag)**

- Build & push docker images with respective version of the microservices to dockerhub
- Use path based filtering so that if there is a change in the **appointments** folder, only the pipeline will build appointments microservice.
- **Which means there should be separate pipelines for all microservices and if a change is in one microservice, only that pipeline should be triggered**
- Update docker-compose file to update the image tag to the one that is built in this pipeline and push back to repo

Phase 3 (30):

- Add Kubernetes Deployment & Service for all the microservices in respective folder in a single file e.g.
  - appointments/k8s/app.yaml
  - doctors/k8s/app.yaml
  - frontend/k8s/app.yaml
  - db/k8s/mysql.yaml
- Make sure you are following best practices of using environment variables for inter-communication, and how to pass those env vars to your pod in best way.

Phase 4 (20):

- All of the microservice's Deployment should have K8s Resources for request and limits on memory & CPU (need to read it yourself), add K8s Probes(readiness & liveness) to your container

Bonus 1

- **Appointments microservice should have environment variables NAME and PASSWORD that should be set from a Secret named appointment**
- **Doctors microservice should mount a file details.txt on path /user/details.txt that should be set from Configmap named doctors where you can have an intro about yourself.**
- **The Configmap & Secret manifests should be added to respective file as mentioned in Phase 3**

Bonus 2

- **Pod should run as non-root user and follow hardened container best practices**
- **Add initContainer(read yourself) based on ubuntu image, which will print your name and sleep for 5 seconds and exit and then main container will start**