**Microservice GCP Case study requirements:**

**Dev**

Create microservices as below for building an internet movies database.

1. **Movie Service:** This service will have a list of movies details like name, lead actor/actress, director, year of release etc, it will be an authenticated service. It will be an independent private service not exposed on api gateway.
2. **Review and Ratings Service:** Using this service, a user can post reviews and ratings against a movie, at a time he can post 1 review/rating for 1 movie and there will be maximum 1 review a user can post against a movie, though he can post reviews for multiple movies. User should be able to edit his reviews and ratings for a movie. It will be linked with User service and movie service. It will be an independent private service not exposed on api gateway and it will be an authenticated service.
3. **Full Movie Details Service:** It will be dependent on all of the above services and return all the details for a searched movie.
4. **Authentication Service:** It will provide spring authentication using oAuth2 token, A user existing in user-service can be authenticated with authentication service.
5. **API Gateway Service:** It will provide the api gateway mechanism for calling all the public api.
6. **Config Service:** This service will provide a mechanism to put all the configuration at single place, and this will be parent of all services as the config details for all services will be fetched from this service.
7. **Registry Service:** All the above services excluding config service will be registered by this service, it is required for spring boot microservices to locate each other service via a common registry.
8. **Monitoring Service:** To monitor the health of all services.

Services from 1-5 are application microservices and 6-10 are infrastructure microservices, each of these service should be testable independently.

**QA**

QA should prepare the test steps/strategy/automation for all the services properly integrated with api gateway and authentication service.

**Dev Ops**

1. Dockerize all services from 1-9
2. Create CI/CD pipeline for all services Using Jenkins, they should be independently built.
3. Single click deployment on gcp using a deployment manager gcp supports.
4. Autoscaling support.

**Technology**

Java 8, Spring Boot Microservice, Hibernate JPA, Postgres DB, Gradle, Spring security, ZUUL gateway, Docker, Jenkins, GCP, Postman

**Design**:

