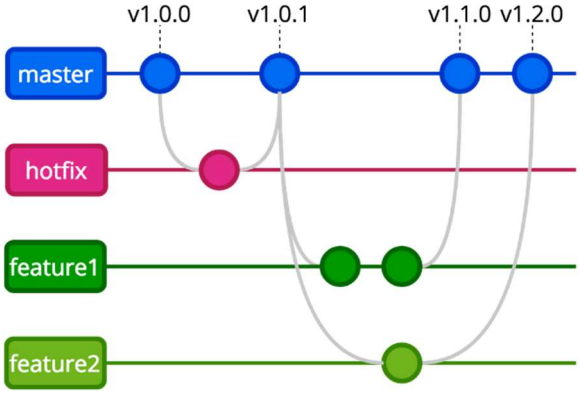
Fullstack Developer candidates tech exam

Created by: Jorge Useche
Reviewer: Gabriel Morris
Version: 1.0

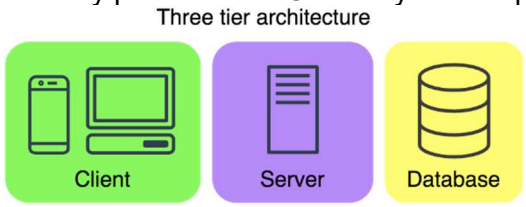
Evite usar el nombre **Banco De Bogotá** en el nombre de los recursos.

Las respuestas deben quedar sobre este mismo archivo *(el que se comparte)*, se pueden poner enlaces a repositorios de código remotos de github u otros servicios, capturas de pantalla o cualquier material adicional que mejore la comprensión de la respuesta.

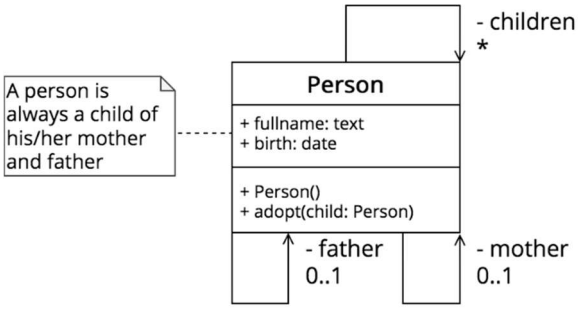
1. Collaborative software build skills
    1. Build a simple code repository with the following workflow.



2. Architecting skills
    1. Given a simple 3 tier web project diagram, write or diagram a plan to deploy it in any public cloud. Choose your components.



3. Programming skills
    1. Given the following model, create a Restful API with CR actions (create and retrieve/read of CRUD) using Java or NodeJS. You could use MySQL, PostgreSQL (desirable), REDIS (desirable) or ElasticSearch (desirable) as persistence. You need to use Docker Swarm or Docker Compose to provision that database. Please add a README.md with all the steps to run the application in localhost.



        1. Use any test framework to add unit tests to your software.
    2. Use Angular to implement the following:
        1. A form to create a **person**, you can add or delete any new fields to the model (support your decision).

        1. Validate the creation if it already exists using any criteria (support your decision).
        2. List all the **persons** in a table.
4. Automation skills
    1. Use CIRCLE CI https://circleci.com/ or any free CI system to execute the previous unit test in all push to the repository, execute once in each existing branch.

Full Stack developer Challenge Solution

1. Repository in Github where you can find the 4 required branches, tags and releases. <https://github.com/chpatinos/Fullstack-developer-challenge.git>

2. To deploy the 3 tier web project there are multiple options for choice, each one depends on the requirements and purpose.

- For a local environment, I would use a group of 3 containers with docker compose for each one of the services in the app.

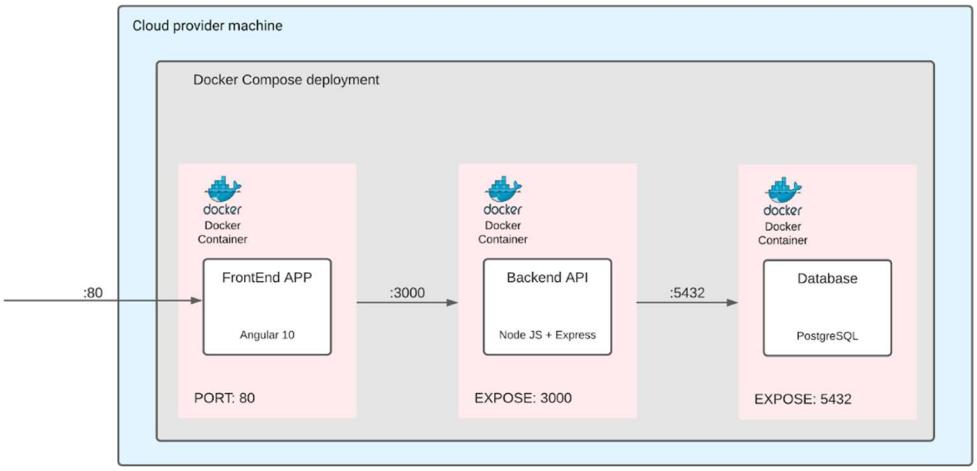


Image 1: Local environment deployment architecture.

- For a testing or staging environment, I would implement a cloud service solution for databases and for the apps client and server would use a serverless compute engine for containers like Fargate by AWS or any other provider alternative.

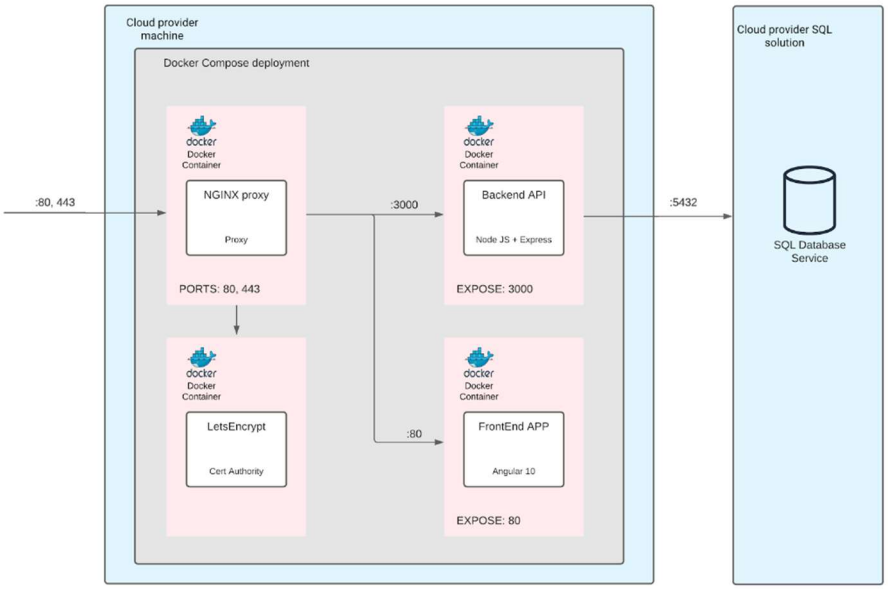


Image 2: Staging environment deployment architecture.

- Finally for the production environment, I would implement a Kubernetes solucion in order to improve the quality of the service and maximizing the availability with Horizontal Pod Autoscaling, Cluster Autoscaling, Ingress Routing for SSL configuration and many other techniques to get that.

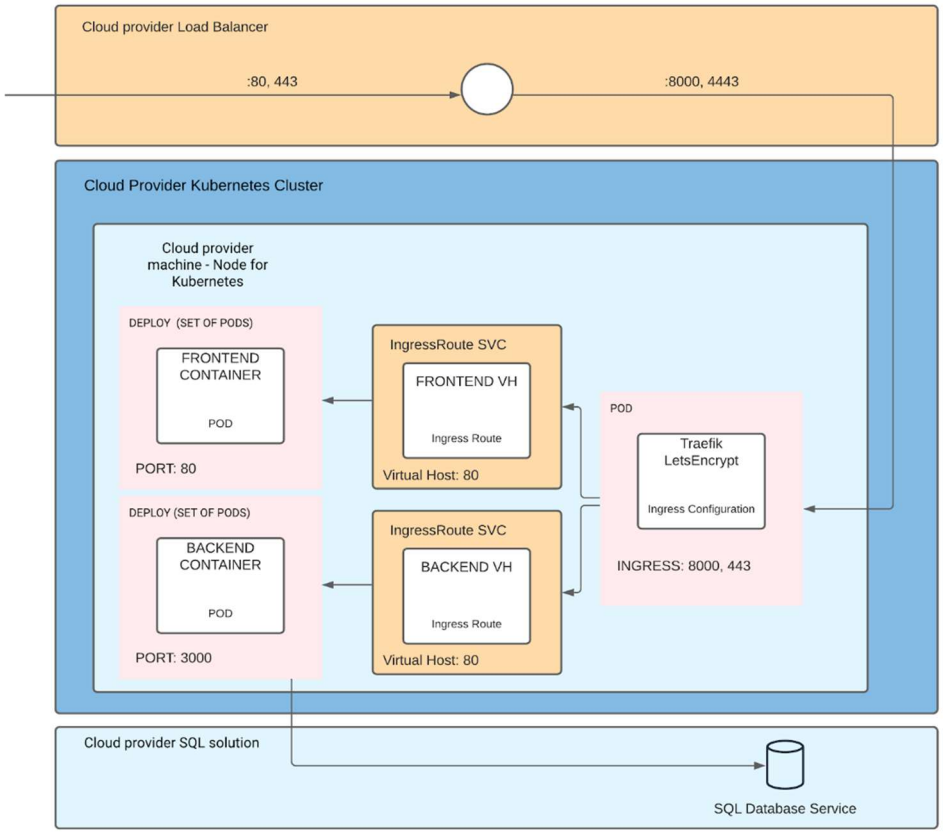


Image 3: Production environment deployment architecture.

Note: This is in a case when information is limited. For any other case is required to generate a deep analysis to take any architectural decision.

3. For this point I used Nodejs + Express to build the API quickly. To the connection with the database I used the ORM sequelize that maps each class and model in the project with a table in the database. I create 5 endpoints (Get all, Get by ID, Create, Father adopt and Mother Adopt). Finally, to test the API I used Jest with the SuperTest package to test each endpoint in the API.

I decided to add a field called CC for an extra identification for a Person and not to use the field “childrens” cause it represents redundant information after applying 3NF (3th normal form). Adding you can access to all the childs of a person with a simple query and the performance writing and reading keeps low.

The criteria chosen for validation if a user already exists was the identification number in Colombia called CC for database.

On the Frontend side, I used Angular 10 with Bootstrap Library to design and build the interface, making use of the best practices for components, responsiveness and accessibility.

4. I used circle CI like CI/CD system, running the testing in the backend and building the frontend.

The repo in: <https://github.com/chpatinos/Fullstack-developer-challenge.git>.