Draft

How to Develop a Website

Although there are many ways to build a website using a web builder, a professional page requires coding and development. Currently, a website builder should not be used if the goal is to make a website suitable for professional businesses. Therefore, the objective of this article is to show how to develop a website from scratch, understanding how to deploy a frontend and a backend using different technologies.

In this article, we select the technologies that we believe are the easiest to build a setup for our website. We work on three environments: frontend, backend and database. First, we recommend implementing the frontend using Angular, which is a framework that a non-expert, but with some web-developing knowledge, can understand easily. Then, for the backend, we suggest using Golang, which is a programming language that has been very popular for software companies these past years. Finally, we propose to deploy our database using MariaDB, which is a free and open-source software that is great for non-experts in the web-development field.

But how is this end-to-end setup going to be unified as a single web environment? We suggest using virtual machines (VMs) to emulate each of the environments. In this case, we should create 3 VMs, one for each environment (see fig.1), using VirtualBox. Then, when the environments are set up, we configure the database's VM. As we suggested, we use MariaDB, which simulates a database server. To start generating it we create a table using SQL language and we assign default values to each cell.

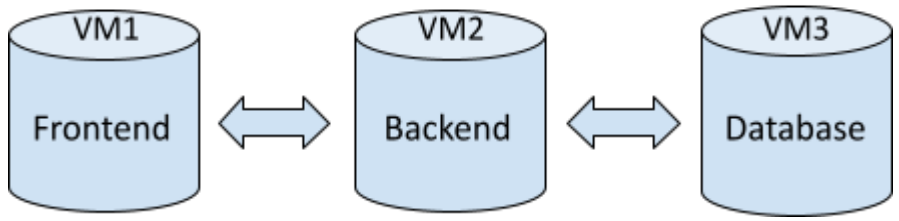

fig.1. Virtual Machines distribution

The next step is to assemble the backend in the second VM, which has an API Rest structure. We start with the configuration file to establish a connection between the database VM and the backend VM. In order to reach the database, we use the libraries "database/sql" and "github.com/go-sql-driver/mysql". Also, it is necessary to know the IP address and port where the database VM is listening. Once the connection is established, it is time to arrange a router that will listen to the incoming http calls. For this file, the libraries needed are "github.com/gorilla/mux" and "net/http". This router will be listening to the indicated port and will redirect the calls at the specified path to the specified method. Then, we create the method that will serve the http requests. This function uses the "net/http" library and the connection established to the database. To get the content of the database we use the *Query()* method with the query we want to execute as an argument. Finally, we just need to manage all these files and functions. This is achieved in the main file. After completing these

steps and by compiling and executing this project we have an API Rest listening on the specified port that can return a value from the database.

Lastly, our frontend with Angular is launched on the third VM. First, we create a new project with the command: *npm new web-page*. If we execute this project, a default page should be able to be seen on "http://localhost:4200/". Then, we create a new empty component with the command *ng g component newPage* and create a global variable "result". The return value of the call *this.http.get<any>('url')* is assigned to the variable. This value is referenced in the html file of the corresponding angular component. Finally, by executing the project our variable stored in the database should be seen on screen.

To conclude, having seen how to create an end-to-end setup for a website, we now have an idea on how to deploy a frontend, backend and web database. We used Angular, Golang and MariaDB, respectively, as our technologies to build each environment and we connected them using virtual machines that simulated the three servers. It is now the turn of the web designer and web manager to start adding content and personalized style to the page.