Development Report of Professor&Course Rating Application

> For better format, please visit the **online page** of this report: https://ntu-21fall-internet-programming-6206.github.io/Professor-Course-Rating-Applicaion/docs/Report

> Our application is deployed to a cloud server and can be visit publicly, please visit: http://34.126.85.190/. Please remember to **disable the cross origin restriction** (a browser's security strategy) of your browser, see the end of 4.3 Deployment & Run in Cloud Server in this report.

1. Project Scenario

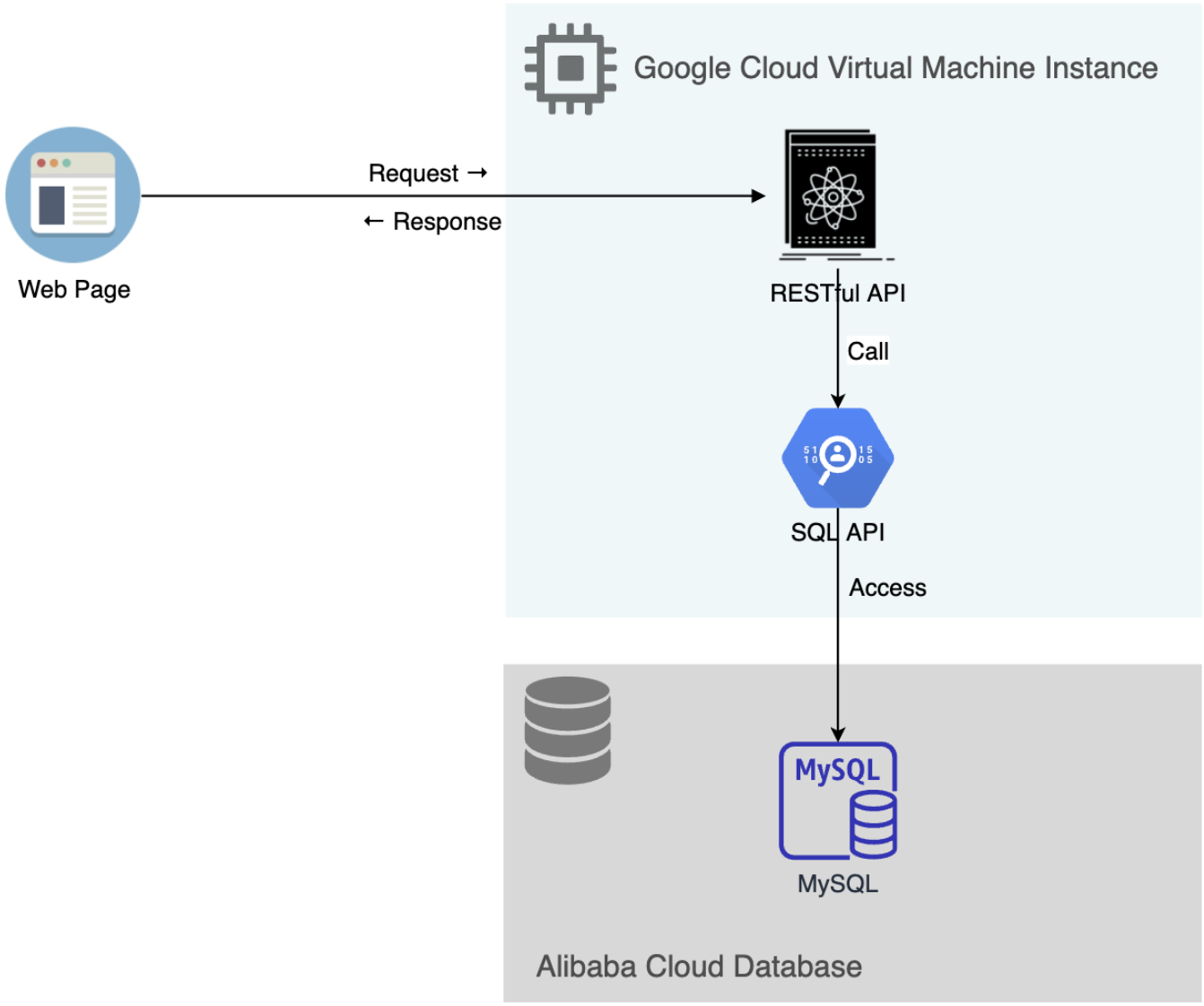
The Professor & Course Rating System (PCRA) provide a platform where students can publish their comments and rates (C&R) on a specific professor and his/her course anonymously. Students can also browse the comments and rates on the professor or the course.

Features Completed:

1. User Register & Login.
2. User Creating Course Entry.
3. Professors List & Courses List, where users could click an entry to access the C&R of that entry.
4. Users can give their C&R for a professor or a course. Their C&R will be shown website publicly.

2. Architecture Design

2.1 Entire Architecture



The above diagram show our project whole architecture. Our project **does not** integrate the frontend (web page code) into the servlet project. Instead, we separate the frontend and backend clearly, and use RESTful API to connect the frontend and the backend. This mechanism is popular in current internet companies.

We confirm the RESTful API URL in our API document. Then we use servlet to implement the RESTful APIs, strictly based on API document. And we write the API URLs into our frontend project to make the frontend able to use such APIs.

For Security and Authentication:

1. We use salted mechanism to protect password.
2. The backend will return a **token** to the frontend when the frontend sends login request. Then the token is added to the header of every request sent by frontend. The backend will check the token's validity each time.

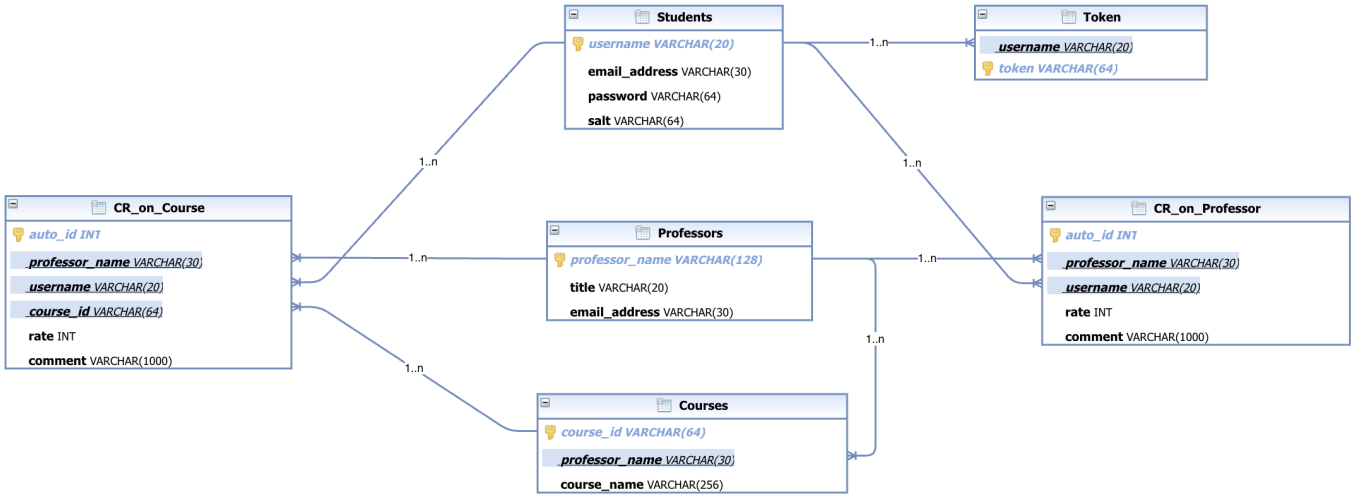
Frontend Codes: codes/Frontend in the repository.

The **API document** of our project is codes/API_doc.md (which is in Chinese for convenience inside our team) in the repository. All backend RESTful APIs are developed based on the API document.

Backend Codes codes/Backend in the repository.

2.2 DataBase Design

As this ER diagram show. Please note that the **foreign key constraint** is specifically shown (field to field) in this diagram. Data Defined Language script of Database, which can also be found in codes/Database/DDL.sql



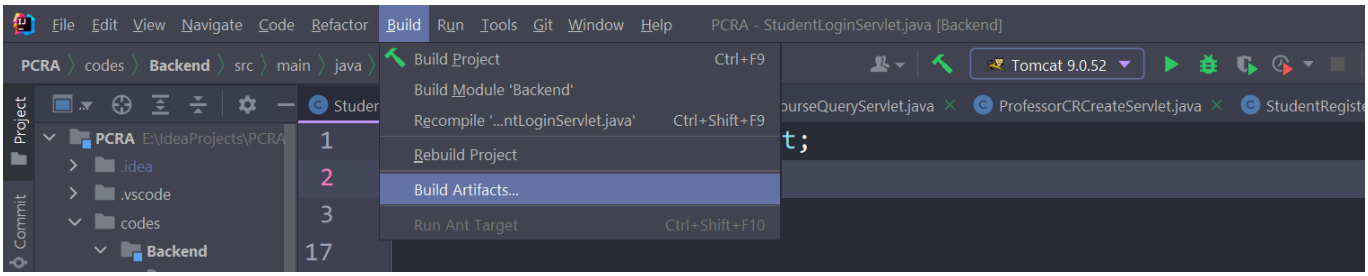
3. Setup Guide

3.1 Frontend

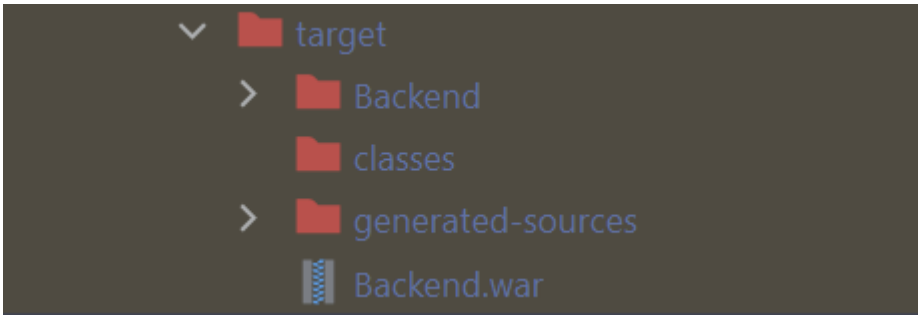
Run the command `npm run build` to obtain the frontend's build directory for deploying the website on server.

3.2 Backend

In the IDE IntelliJ IDEA, The build can be done by building artifacts:



Thus, the war file is generated in the target directory.



3.3 Deployment & Run in Cloud Server

Open Terminal on Mac, use `ssh` command `ssh root@34.126.85.190` to login the google cloud server.

On server, Use `wget` to download Tomcat 9.0.54 and decompress the tar file.

```
wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.54/bin/apache-tomcat-9.0.54.tar.gz
tar -xzvf apache-tomcat-9.0.54.tar.gz
```

Use `scp` command or use FTP tool such as Filezilla on Mac to upload backend's war file to Tomcat's webapps directory (apache-tomcat-9.0.54/webapps). And rename the war file to "ROOT.war", because Tomcat will automatically set "ROOT.war" as the root web app and users can access such a root web app without enter any suffix after server's IP & port.

Then enter apache-tomcat-9.0.54/bin. Run the script startup.sh `bash startup.sh` to start the Tomcat. Now, the backend is running. Our project's backend is running at http://34.126.85.190:8080.

Then deploy the frontend project to the server.

First, install nginx `sudo apt install npm` on the server. Then upload frontend's build folder to server's /var/www/html directory.

Use `vim` command `vim /etc/nginx/sites-enabled/default` to modify the config of nginx, as codes/niginx_config.conf in the repository.

Then restart nginx service `service nginx restart` to make the new config work. Now our website run at server's 80 port. User can directly use the URL http://34.126.85.190 to visit our website.

To normally use the website, user should disable the **cross origin restriction** (a browser's security strategy) of the browser.

Take Safari as an example. First open Safari-Preferences-Advanced. Enable the "Show Develop menu in menu bar" option. Then open develop menu and click "Disable Cross-Origin Restriction".

4. Description of Roles

- Chen Haoyu: Frontend Engineer, UI/UX designer.
- Lin Jingkun: Backend Engineer. Cloud Server and Cloud Database Applier.
- Wang Mingye: Team Leader, Architect (Design the whole architect), Backend Engineer (Write SQL APIs and RESTful APIs), Database Administrator (Define and create data tables, check the data in database), Site Reliability Engineer (Deploy both frontend project and backend project to the cloud server).