**PhoneBook Application by Munyaradzi Mandava**

**Technology Used:**

Angular 7, Angular Material, Nginx, .Net Standard 2.0, .Net Core 2.2, Entity Framework Core 2.2, and Docker

**Important docker commands:**

* List all images: docker images
* List all containers: docker ps
* Delete every Docker containers: docker-compose down then run the command docker rm -f $(docker ps -a -q)
* Delete a Docker container with CONTAINER ID 10e62ea29d83: docker rm –f 10e62ea29d83
* Delete every Docker image: docker rmi -f $(docker images -q)
* Delete a Docker image with IMAGE ID 870fda08c907: docker rmi –f 870fda08c907
* Build your application: docker-compose build
* Run your application: docker-compose up

**Environment:**

* Make sure you have the following installed

1. .Net Core 2.2 SDK
2. Node v10.12.1
3. Visual Studio 2017
4. Visual Studio Code

* Remove the global Angular CLI (if installed): npm uninstall -g @angular/cli
* Install the latest Angular CLI globally: npm install -g @angular/cli
* To upgrade or update npm run the following command: npm i -g npm
* Install angular material by navigating to ClientApp and run the command: npm install --save @angular/material @angular/cdk @angular/animations
* Make sure you add an angular material theme of your choice in styles.css: e.g. @import "~@angular/material/prebuilt-themes/indigo-pink.css";

**Architecture and other Software Design Principles used:**

**Clean Architecture / Onion Ring Architecture**

**Asynchronous Programming**

**Unit Tests**

**Functional Tests**

**Integration Tests**

**Docker Containerization**

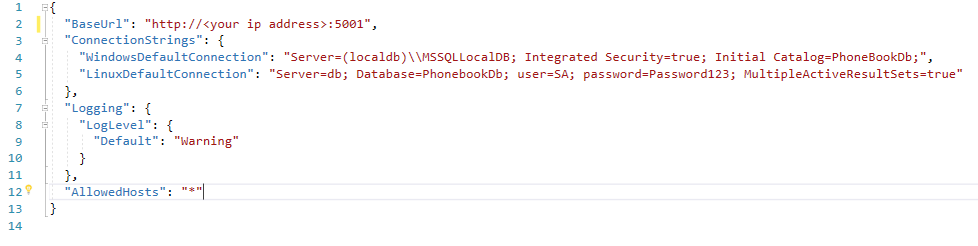
**Container Orchestrator Support**

**Single Page Applications**

**Database, Entity framework**

**Database Configuration:**

Make sure you add your machine’s ip address in the baseUrl



**The application will handle the connection strings depending on the Operation System the application. In case you want to modify the connection strings see the following:**

**In (docker) Linux use the following connection string:**

"WindowsDefaultConnection":"Server=(localdb)\\MSSQLLocalDB; Integrated Security=true; Initial Catalog=PhoneBookDb;"

**In (local) windows use the following connection string:**

"LinuxDefaultConnection":"Server=db; Database=PhonebookDb; user=SA; password=Password123; MultipleActiveResultSets=true"

**Deploying the Frontend:**

Navigate to the *ClientApp* directory and run the following command: npm install and then run ng build --prod

This command will compile the angular application code from typescript to minified javascript and output application files into a *dist* directory.

When deploying the frontend application we only deploy the contents in the *dist* directory.

**Deploying the Backend:**

Make sure the *ASPNETCORE\_ENVIRONMENT* variable is set to *Production.* This included the variables in the launchSettings.json file. See the following figure:



Rename ASPNETCORE\_ENVIRONMENT from *Development* to *Production*.

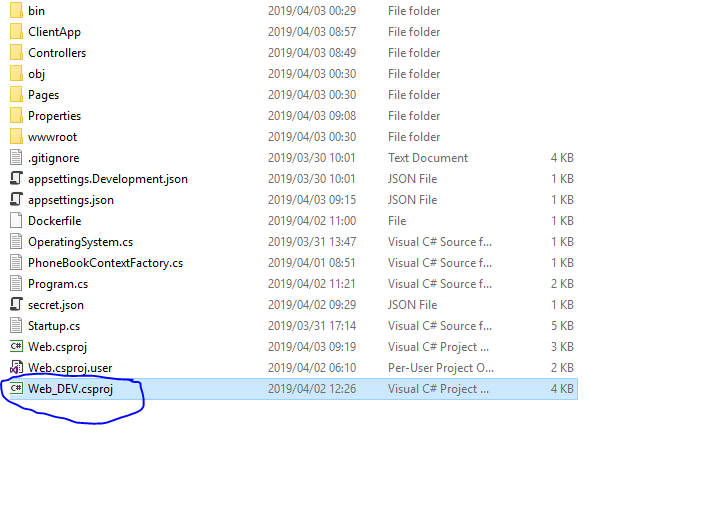
To run the solution run your local PC in visual studo rename to *Development*.

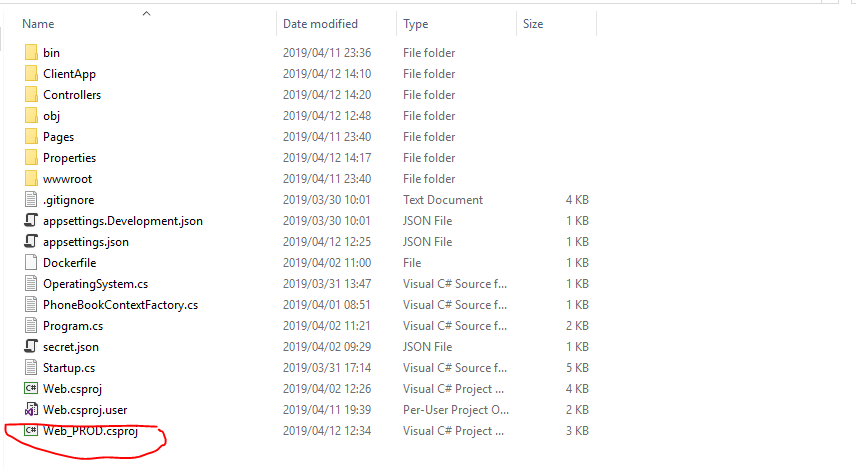


In your Web.csproj file make sure you comment out or remove the following block before running the command: docker-compose build



The dev Web.csproj file is provided see file name Web\_DEV.csproj. Rename it to Web.csproj if you are running your solution in visual studio see the following image:





To run your solution on your local PC in visual studio add the Target BLOCK (see Web\_DEV.csproj). In our case we are running docker and the block has been removed for you (Check Web\_PROD.csproj).

**Running the Application on your Local machine**

The provided solution has been configured to run on the local machine in visual studio 2017 or using dotnet run command from the root directory.

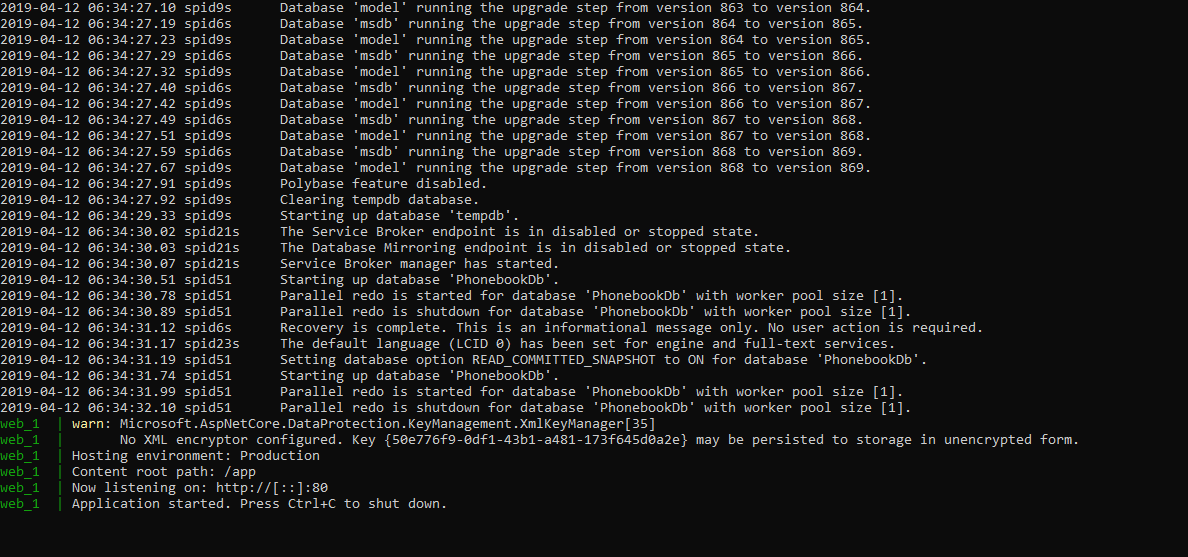
**Running the Application in docker containers**

Just to make sure everything build well, from the Web directory run dotnet restore Web.csproj or from the root directory run dotnet restore src/UI/Web/Web.csproj

Run the command: docker-compose build

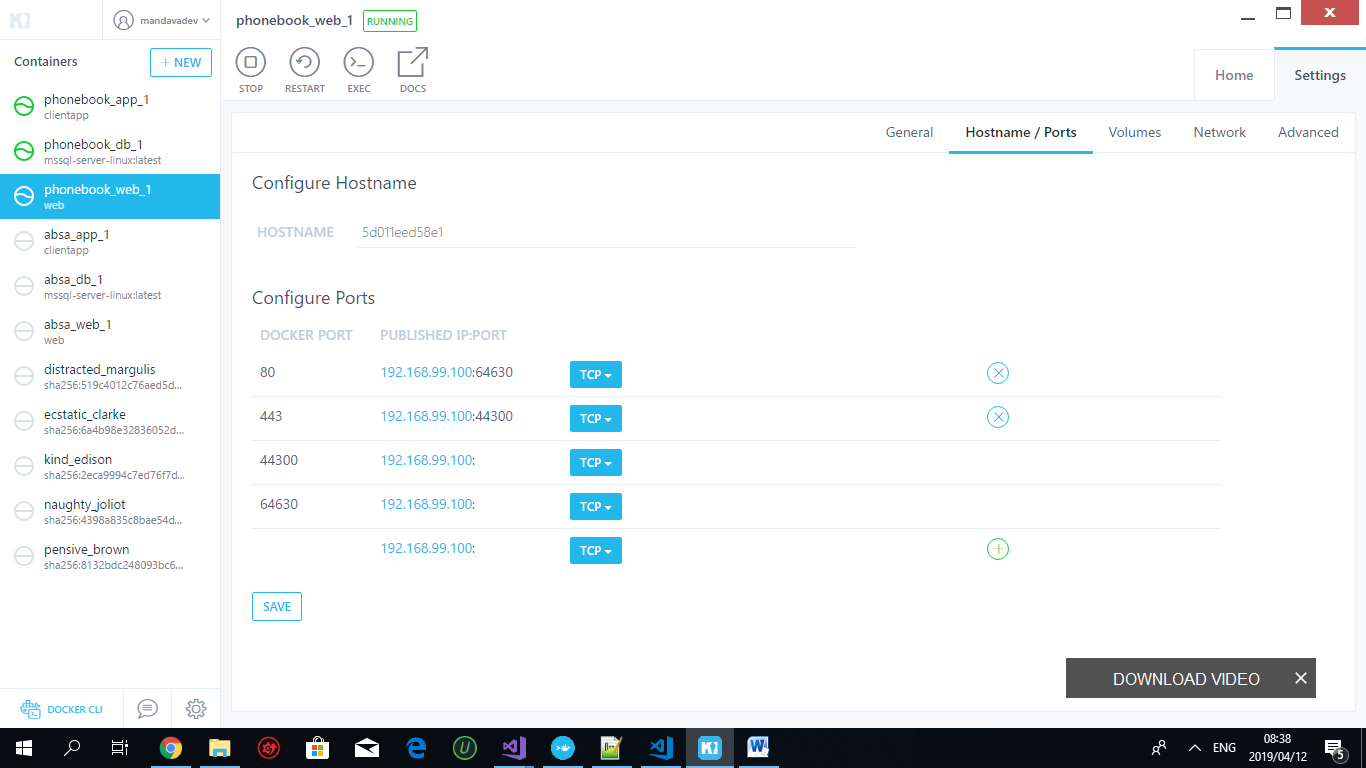
And then lastly: docker-compose up

In your docker terminal if successful you should see something like the following:



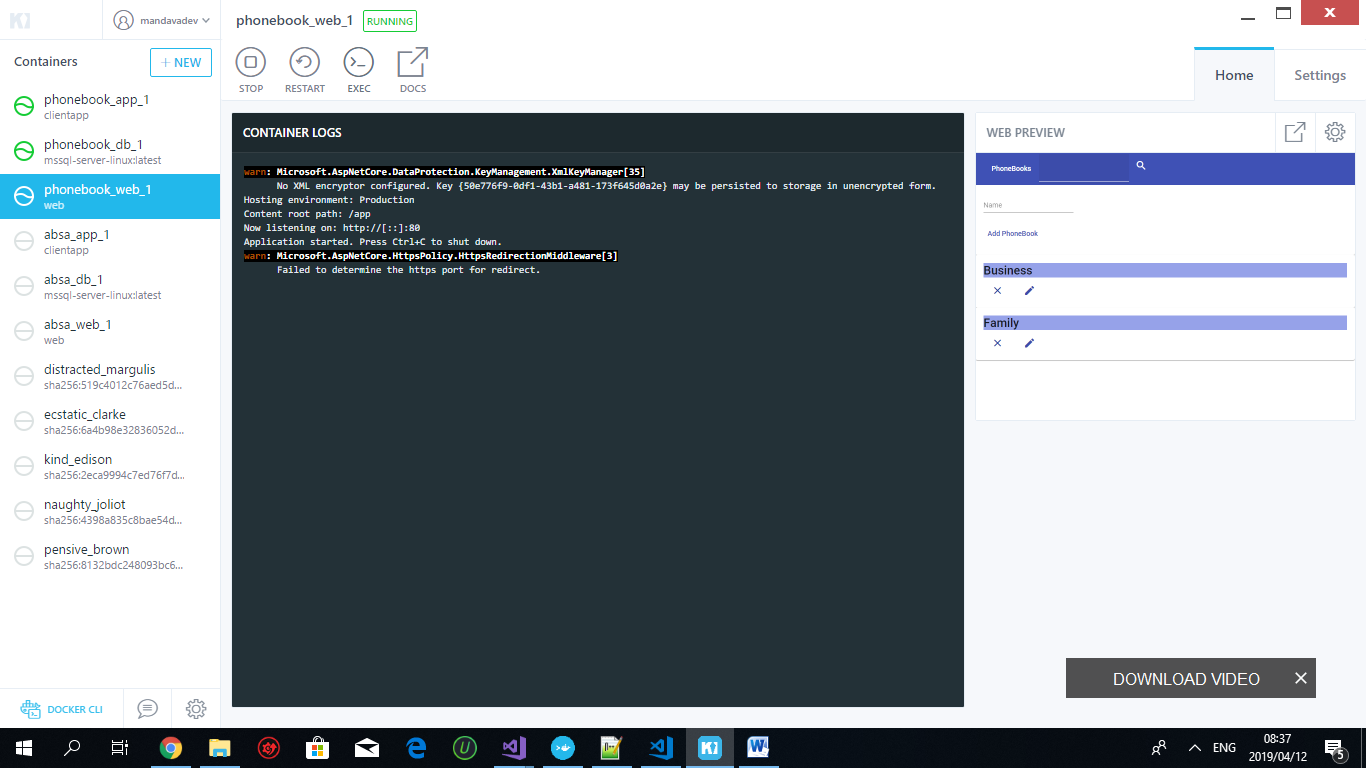
**Open Docker Kitematic**

The BaseUrl should also match your PUBLISH IP:PORT for your docker container you can inspect the values in docker kitematic; see the image below:



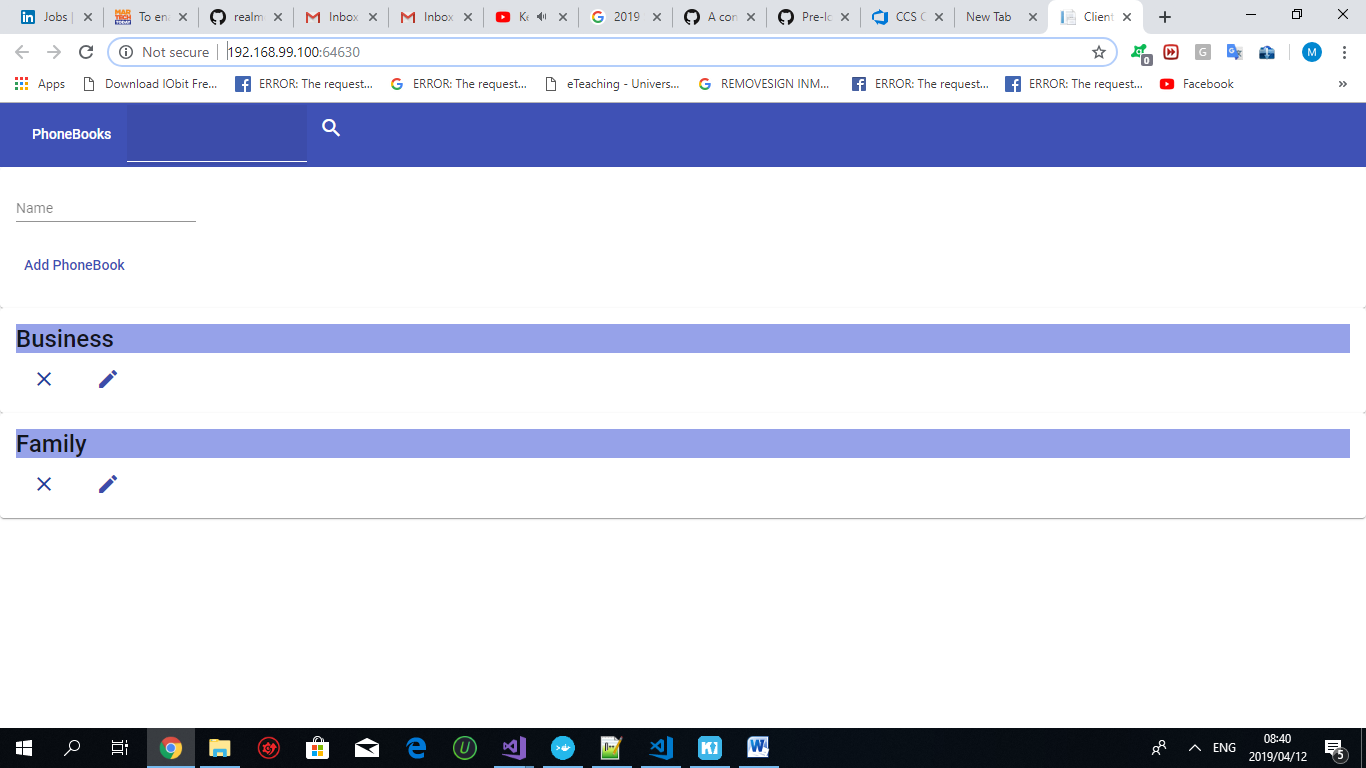
The diagram shows the available ports and application can be access through: for http:// <http://192.168.99.100:64630/> and

**View Your App in Docker**

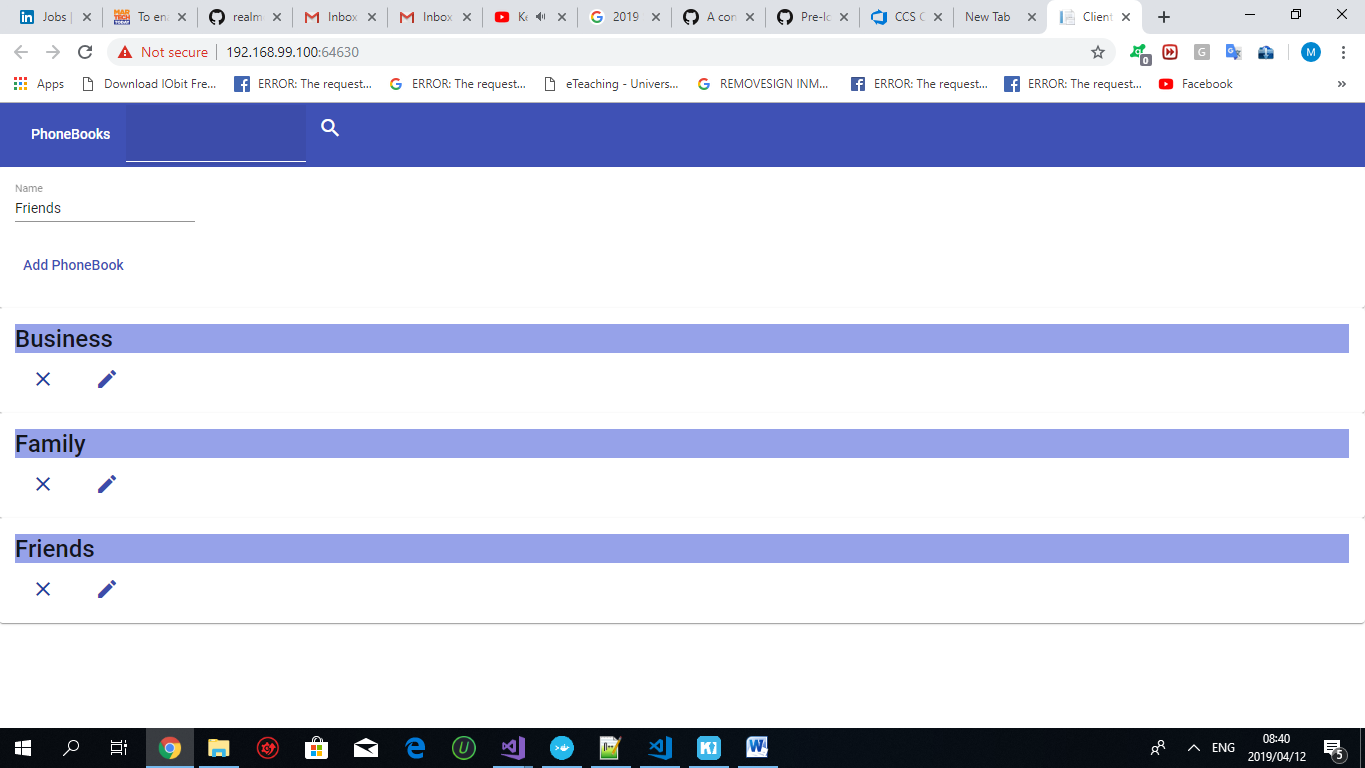


**Demo**

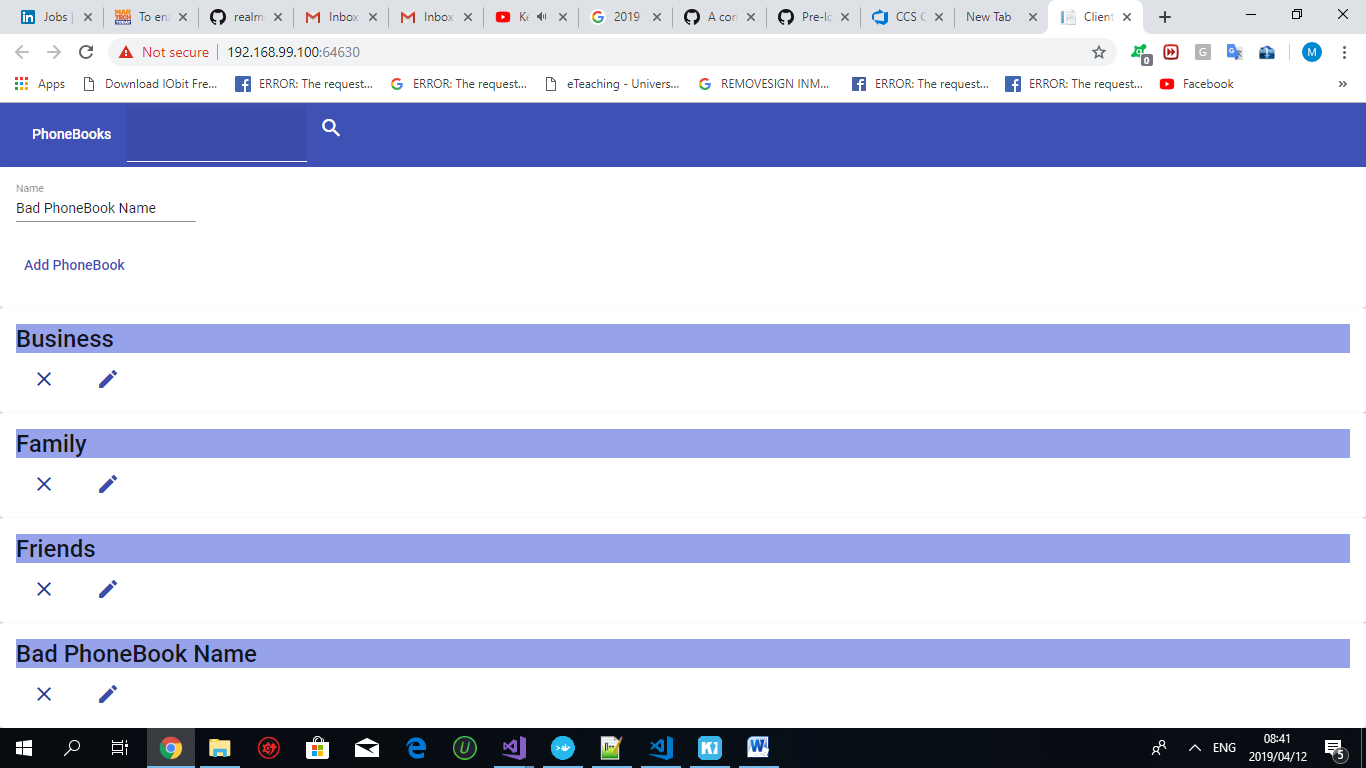
* 1. **PhoneBooks Home**



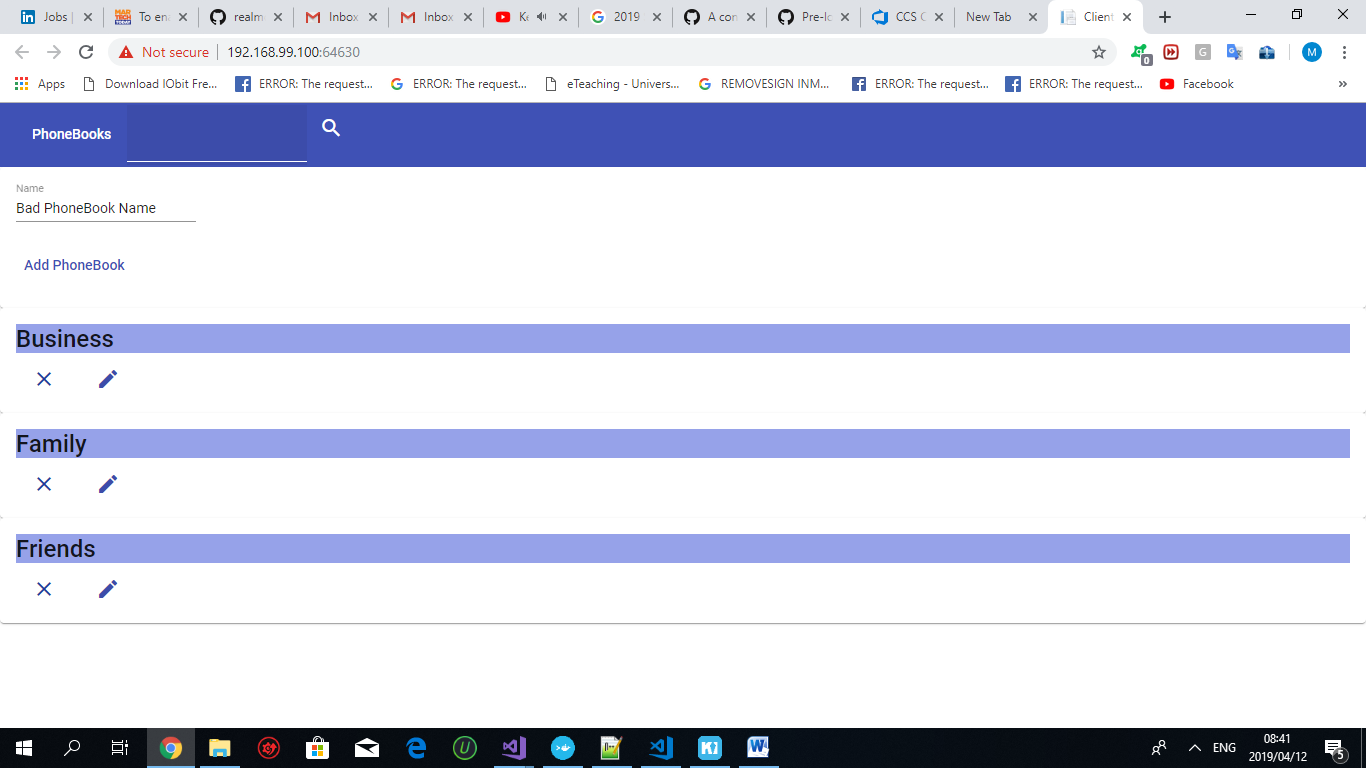
* 1. **Add New PhoneBook**



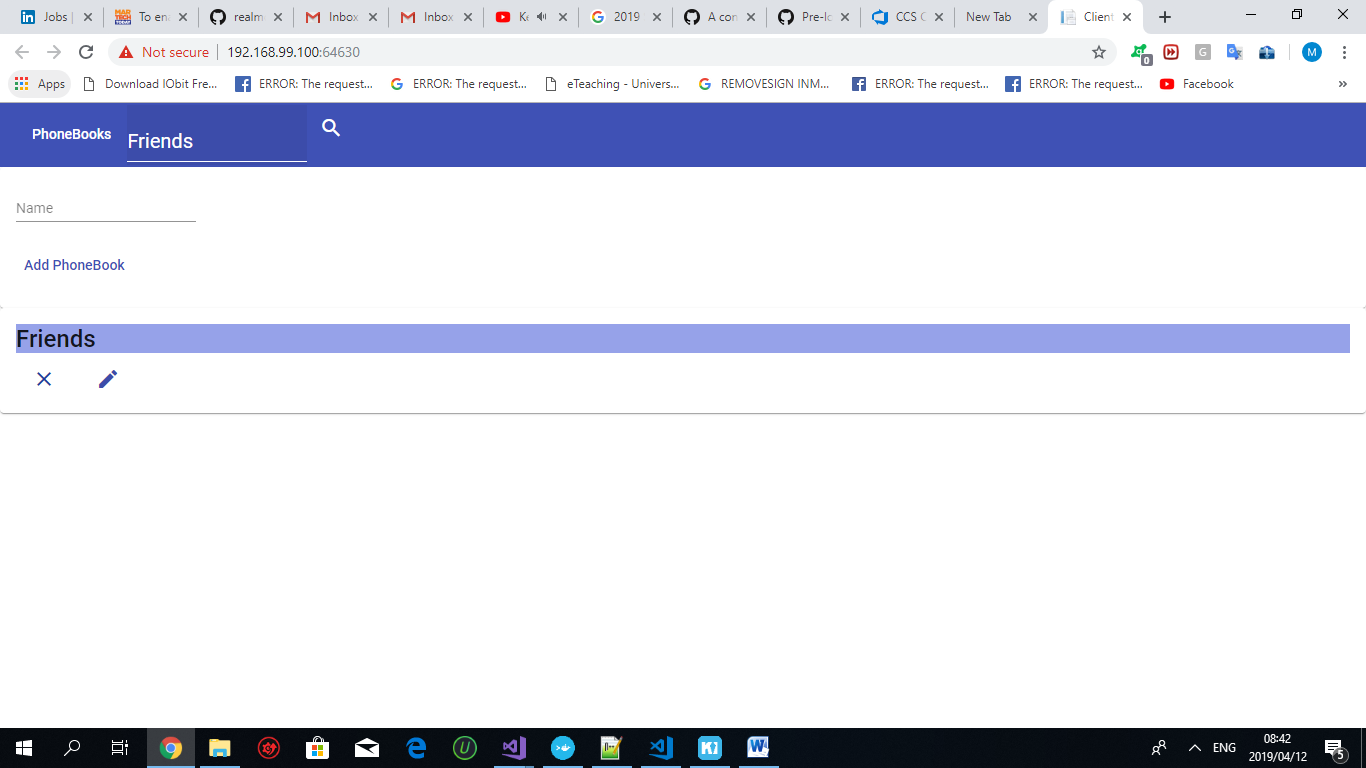
* 1. **Delete** 
     1. **Delete PhoneBook (1)**



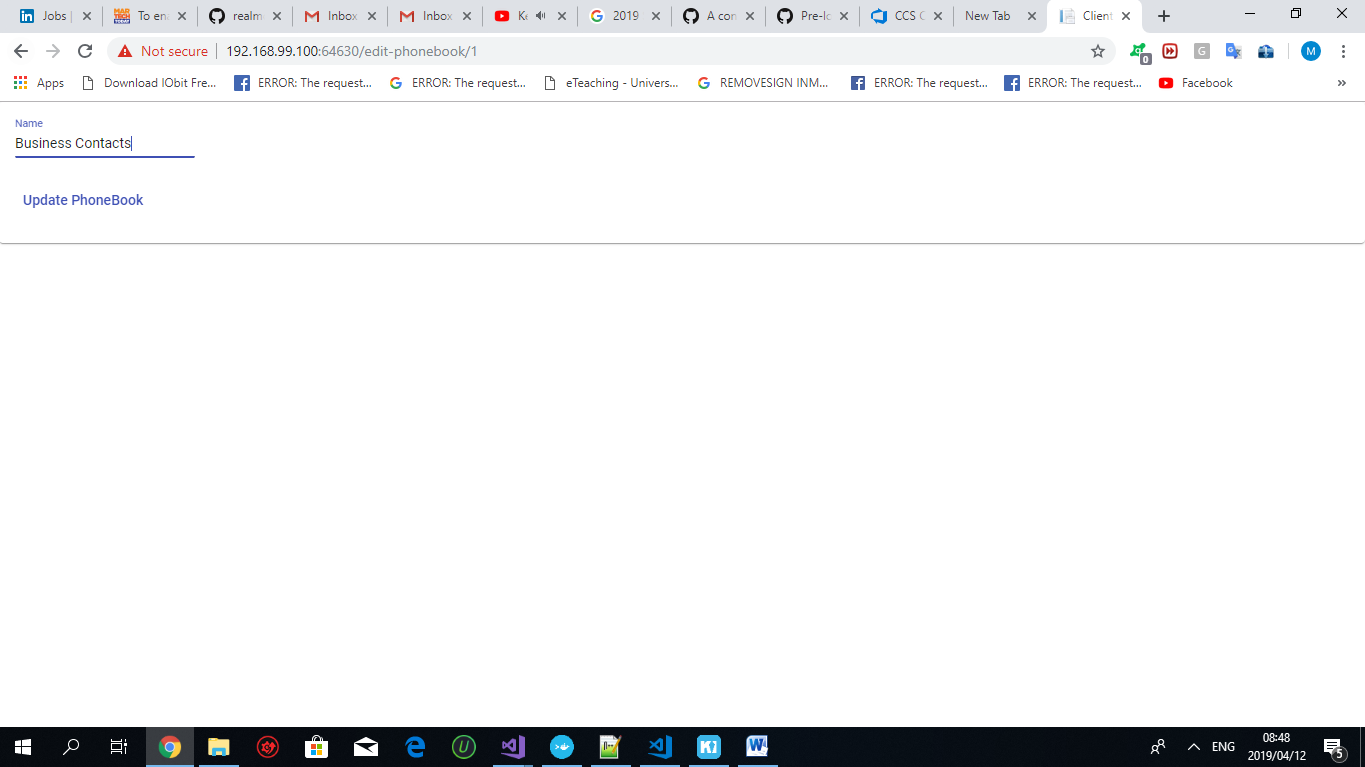
* + 1. **Delete PhoneBook(2)**



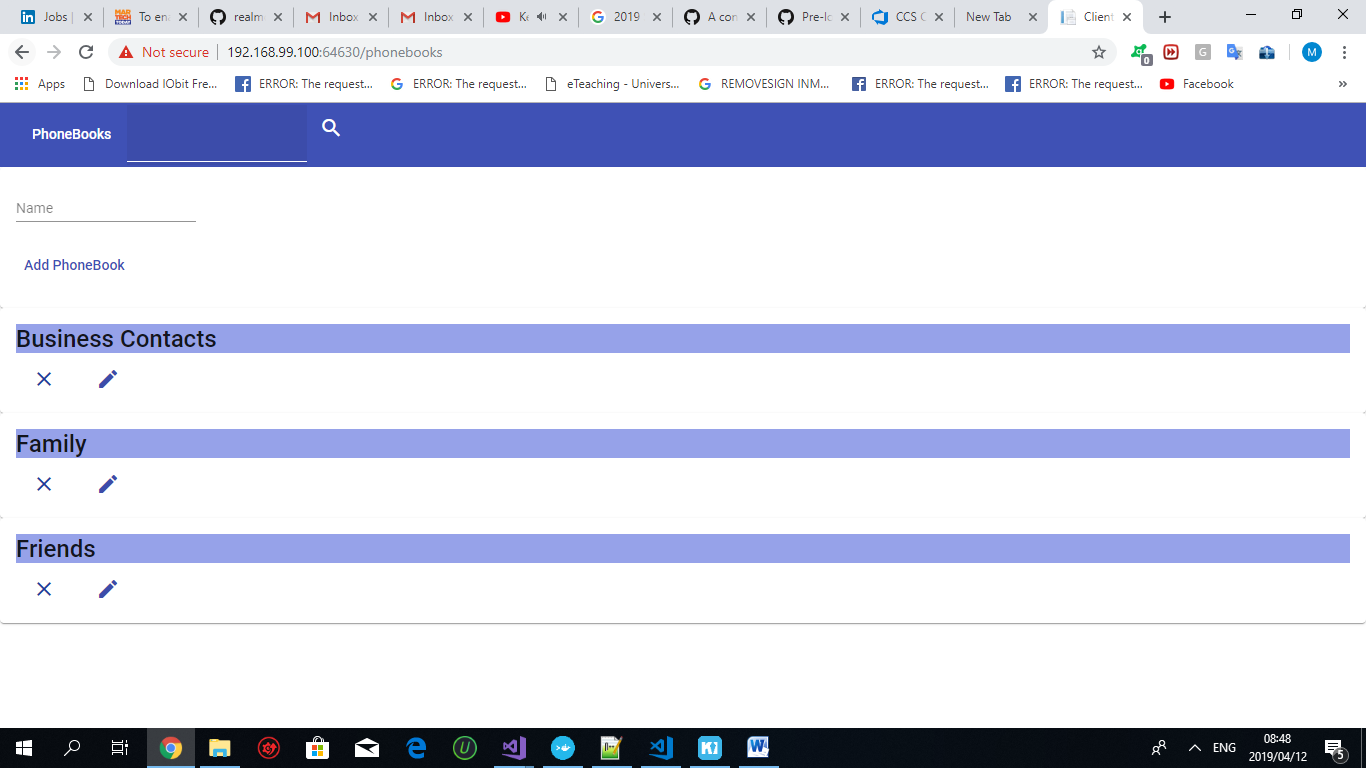
* 1. **Search for a PhoneBook**



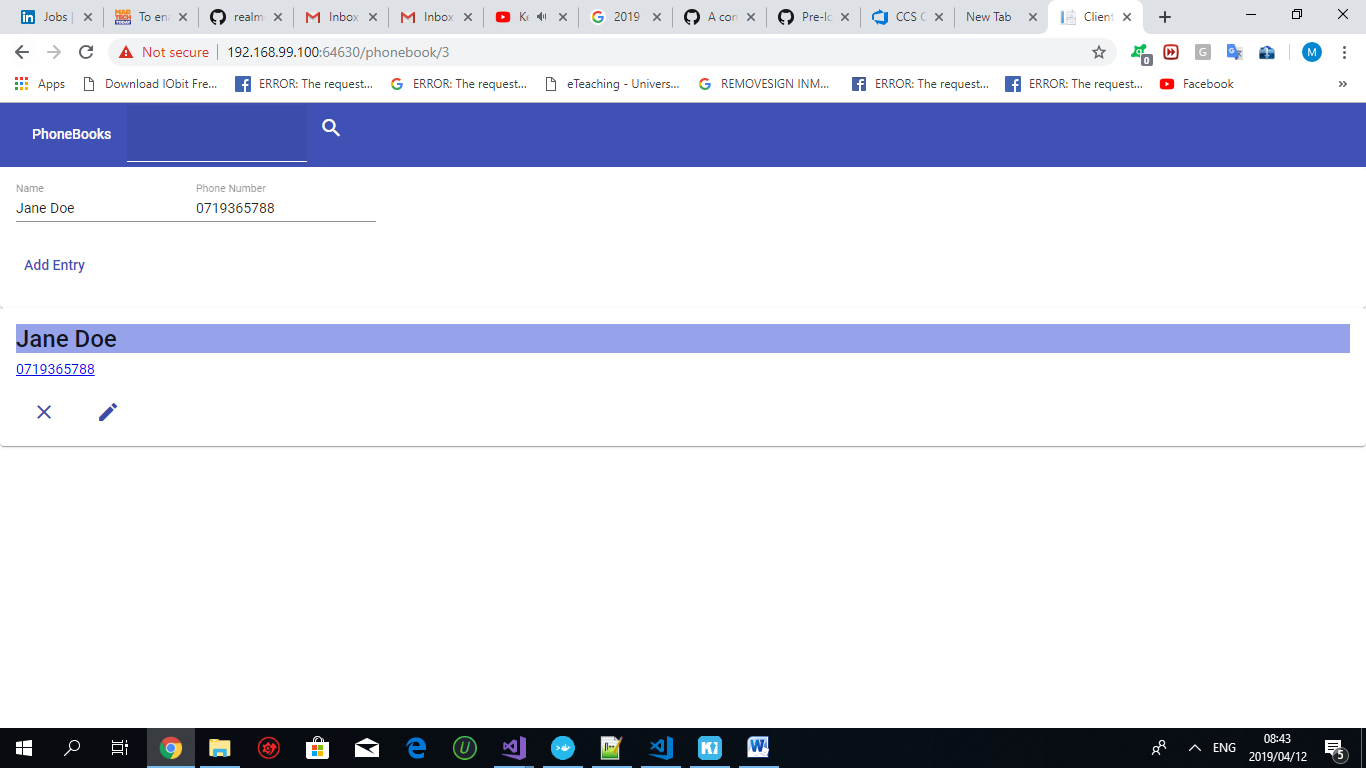
* 1. **Edit** 
     1. **Edit PhoneBook (1)**

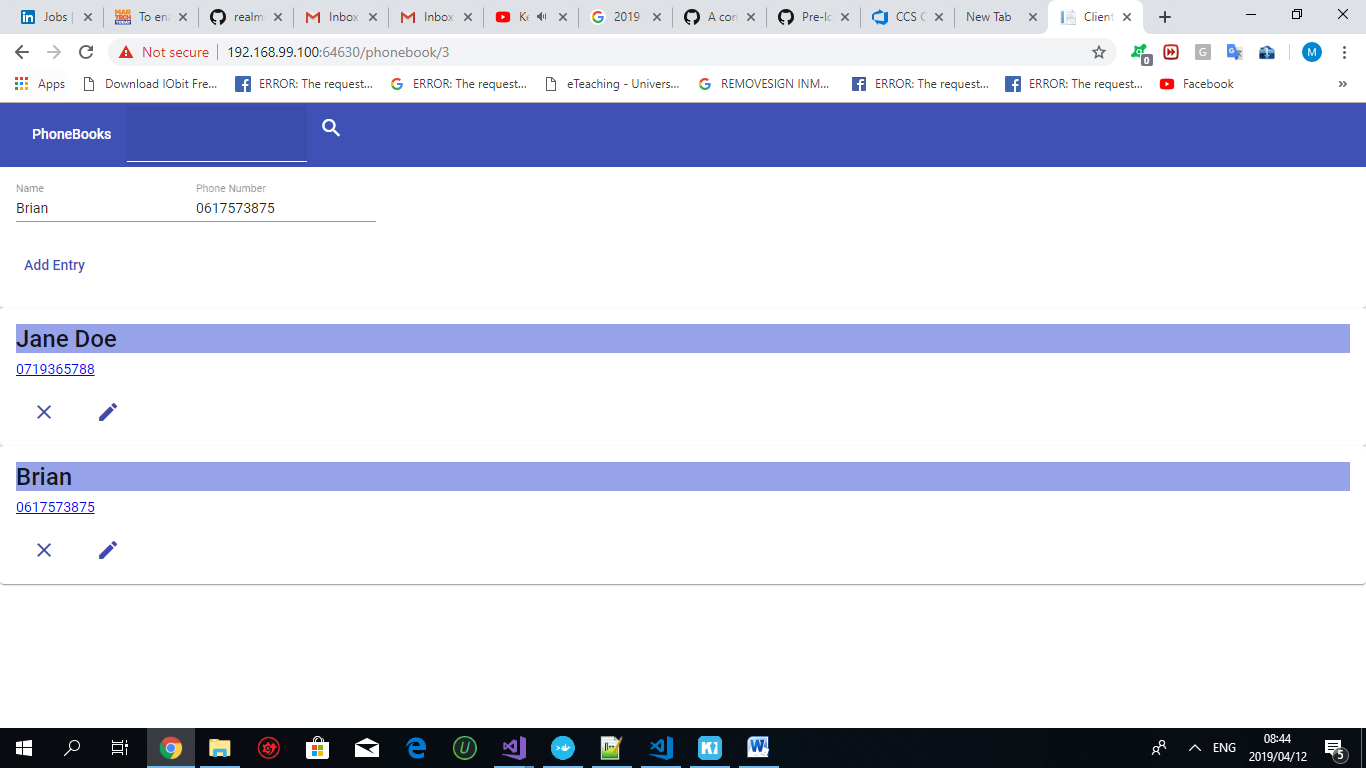


* + 1. **Edit PhoneBook (2)**

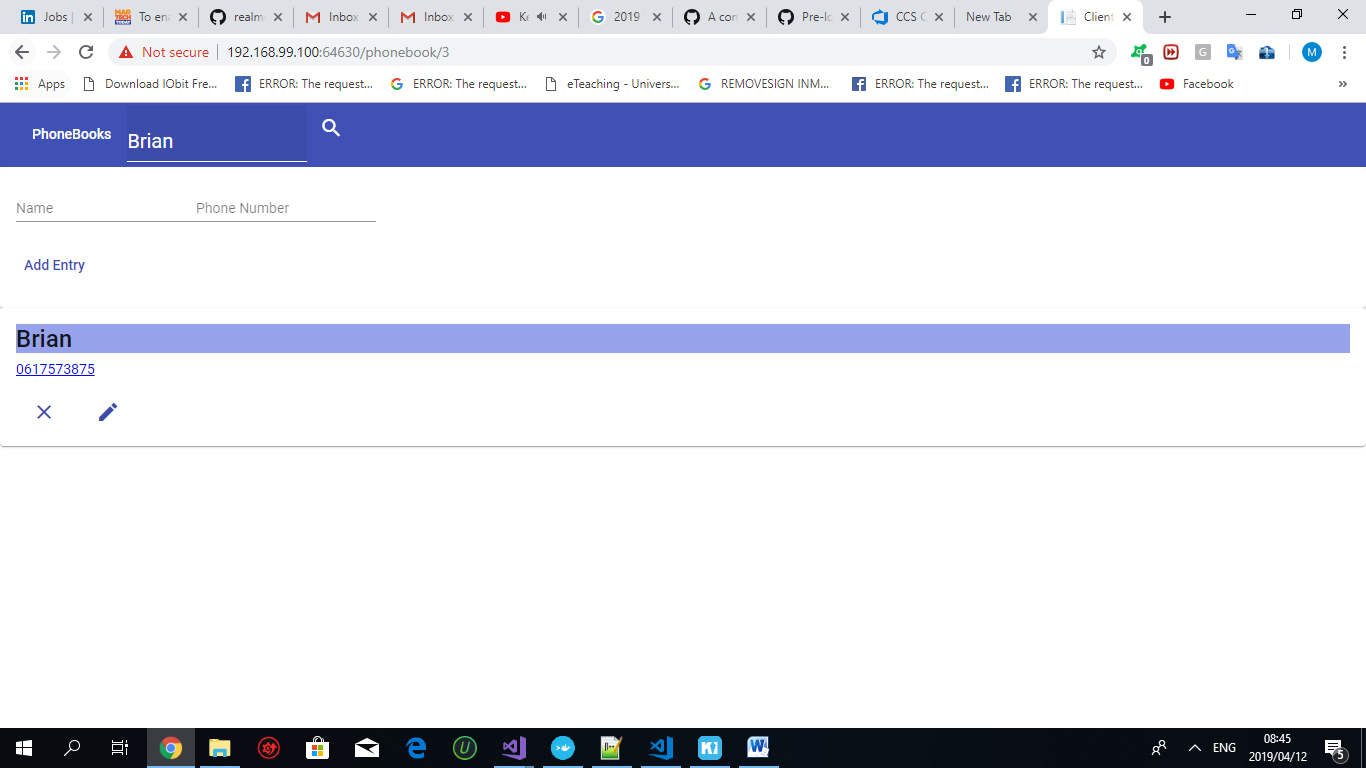


**Add New Entries to (Friends)PhoneBook**

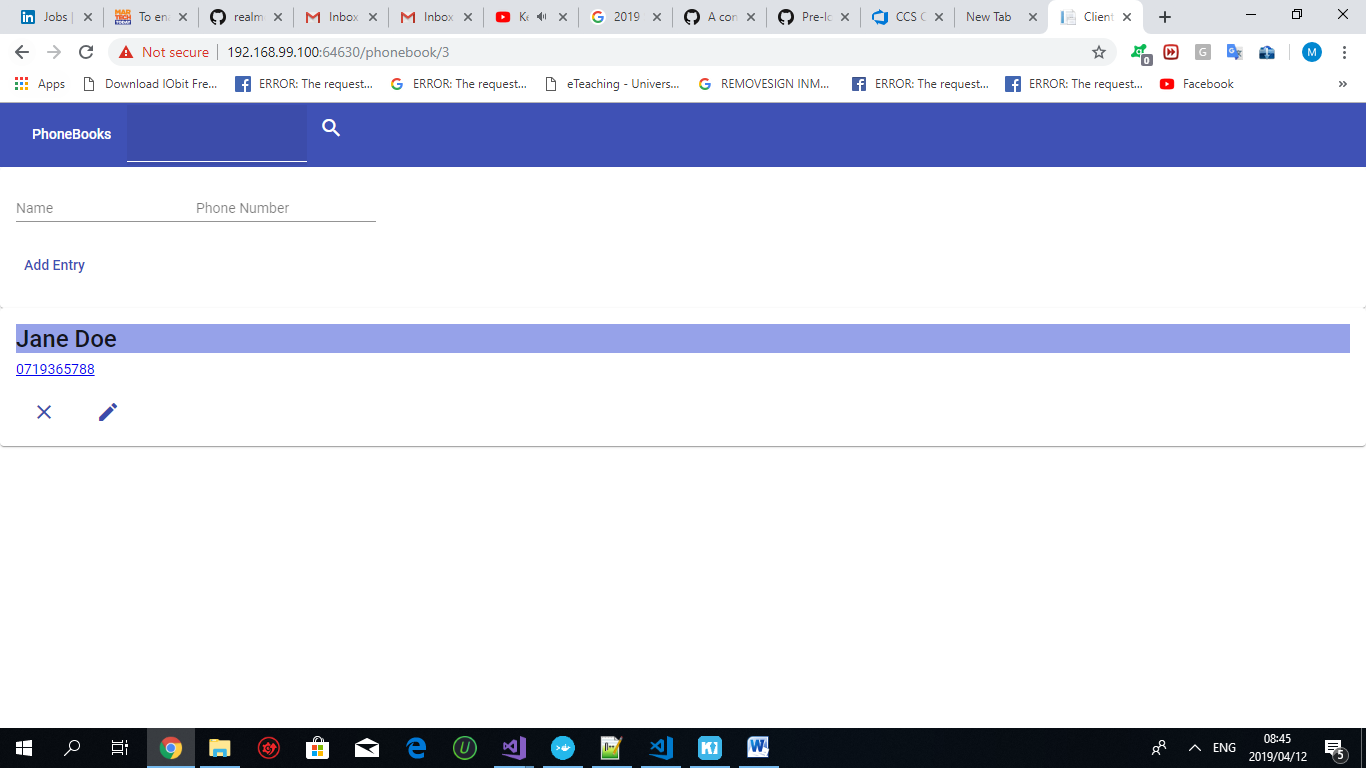




**Search Entry**



**Delete Entry**



**Edit Entry**

