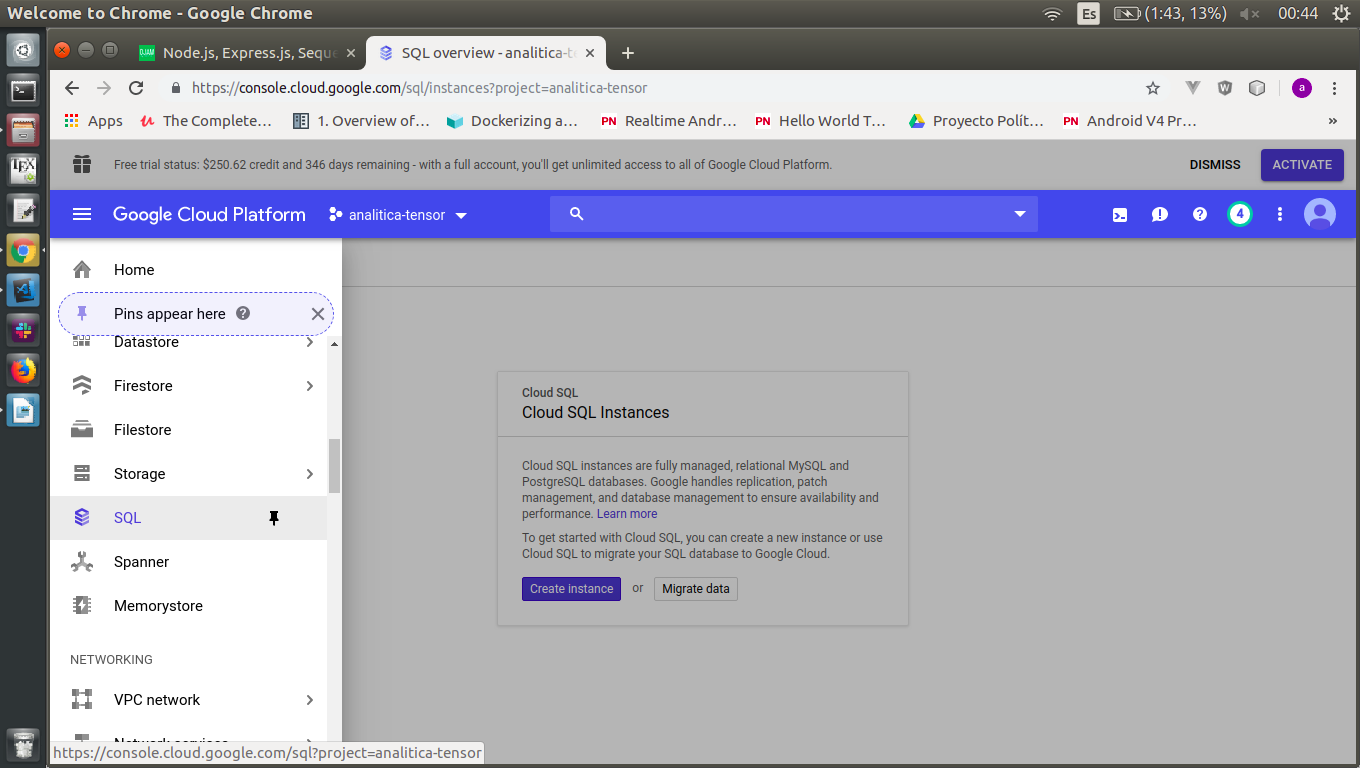
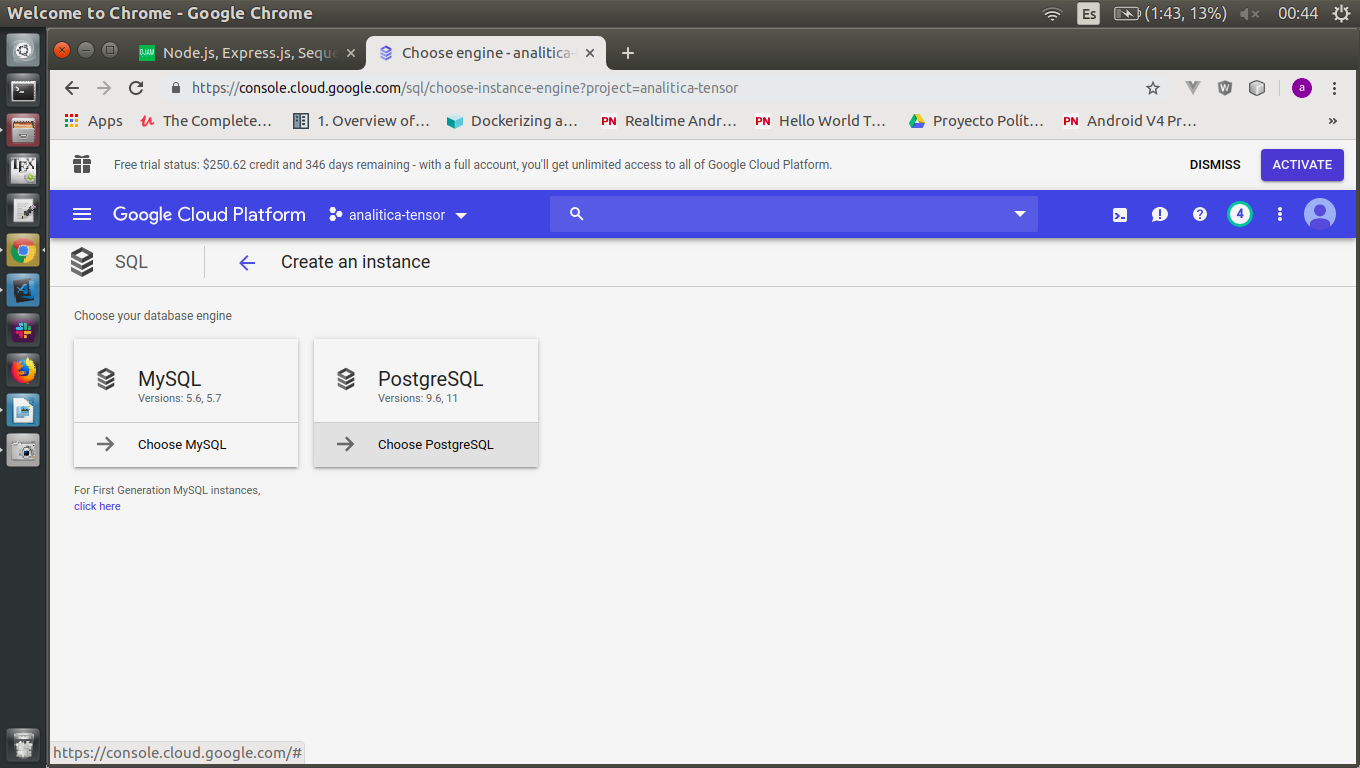
**Habilitar proyecto en swagger para lograr conectividad con base de datos en PostgreSQL**

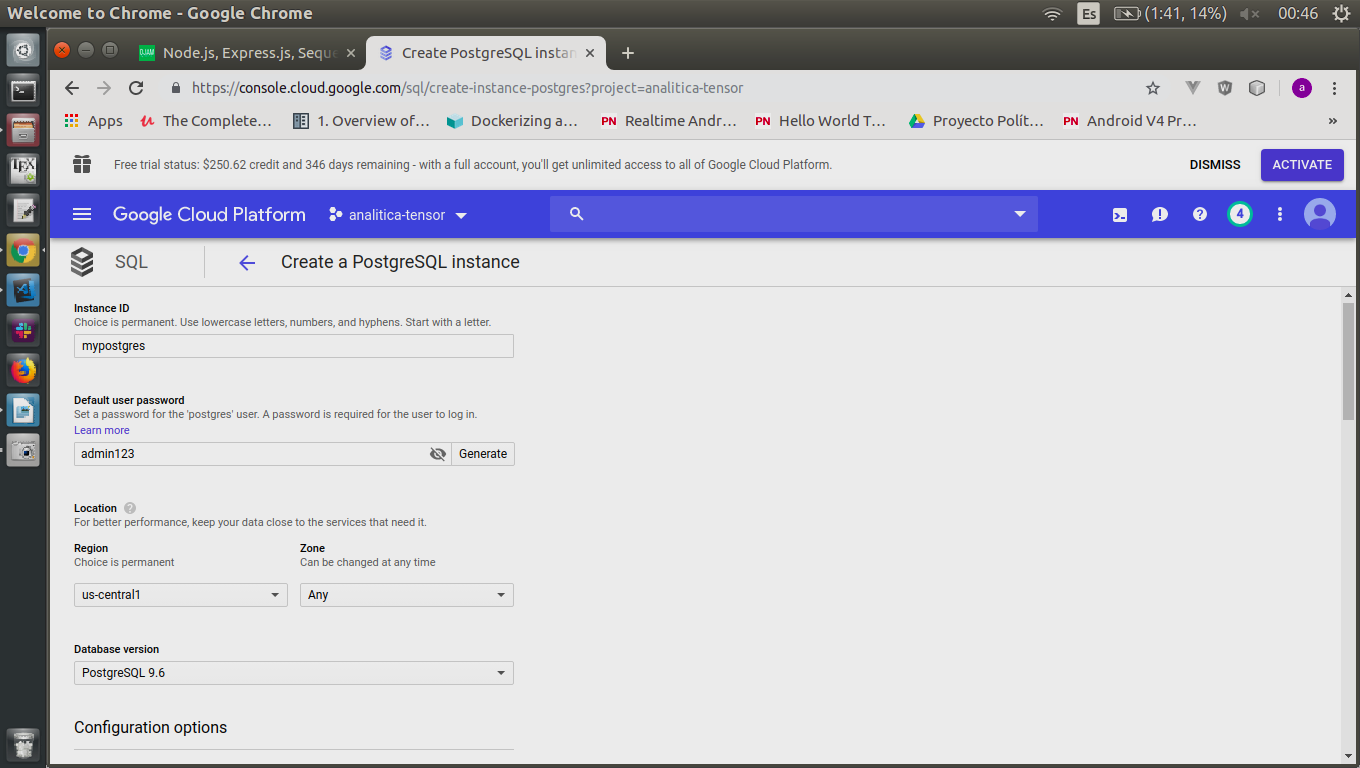
8.- Creación de instancia de PostgreSQL en la nube, ingresar a [https://console.cloud.google.com](https://cloud.console.google.com/)



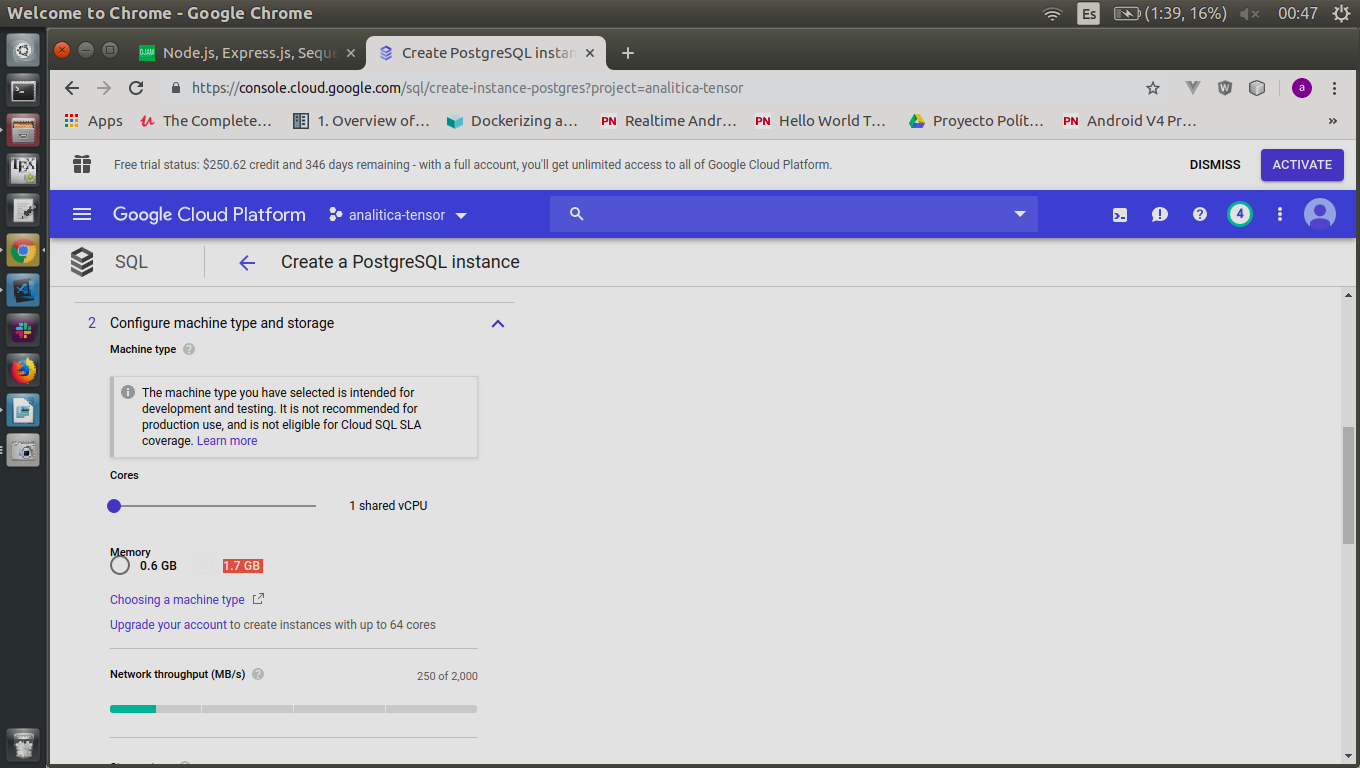
- Seleccionar instancia de PostgreSQL



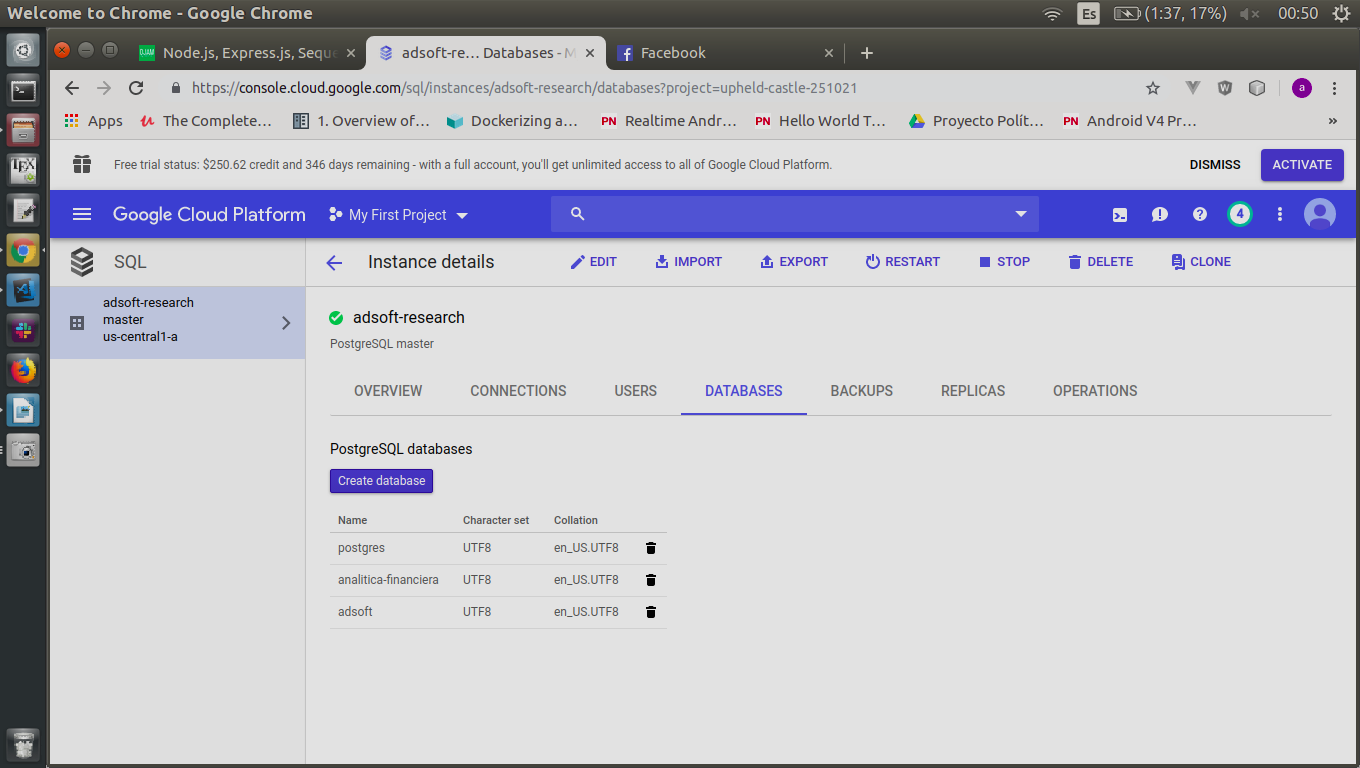
- Capturar nombre de instancia, password de usuario *postgres*



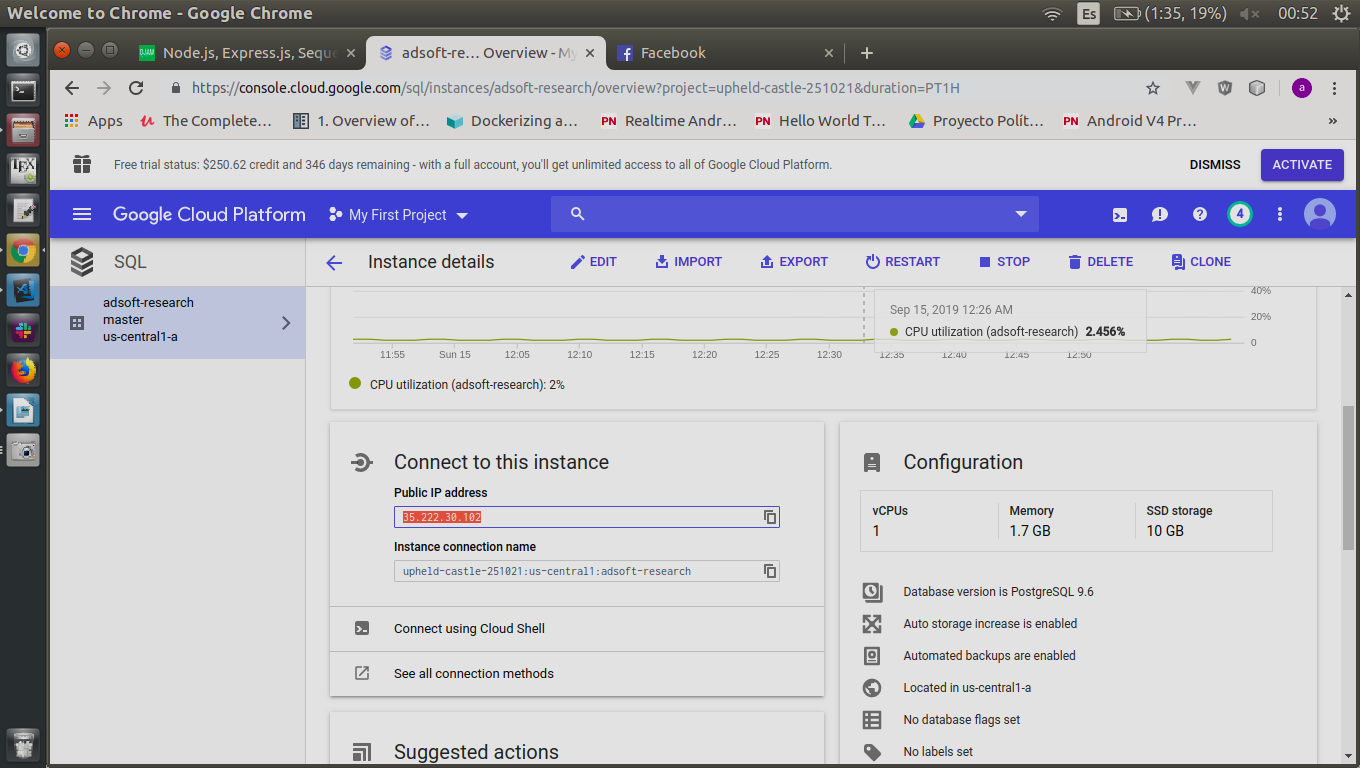
- Seleccionar tipo de máquina, para etapa de desarrollo 1 shared vCPU, 1.7 gb RAM, 10gb HD



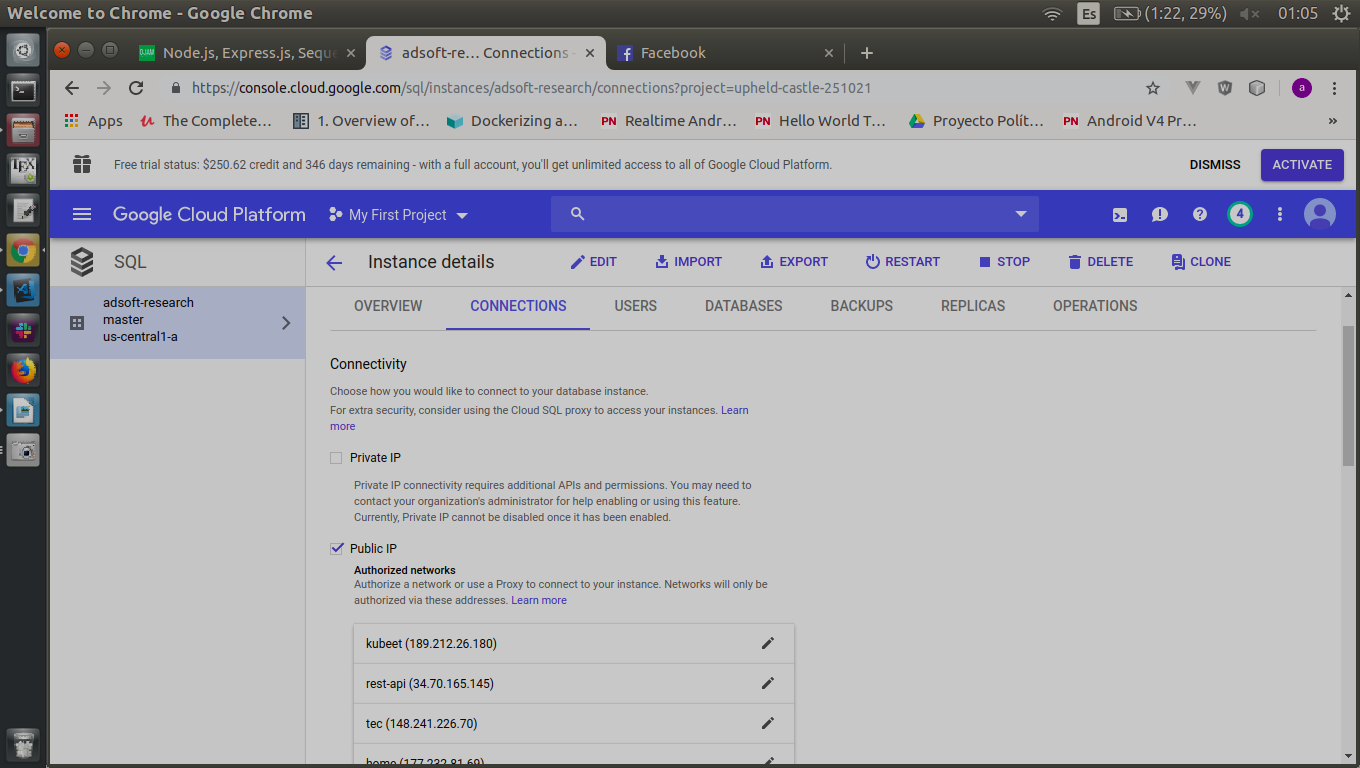
- Crear base de datos data-science-<user>



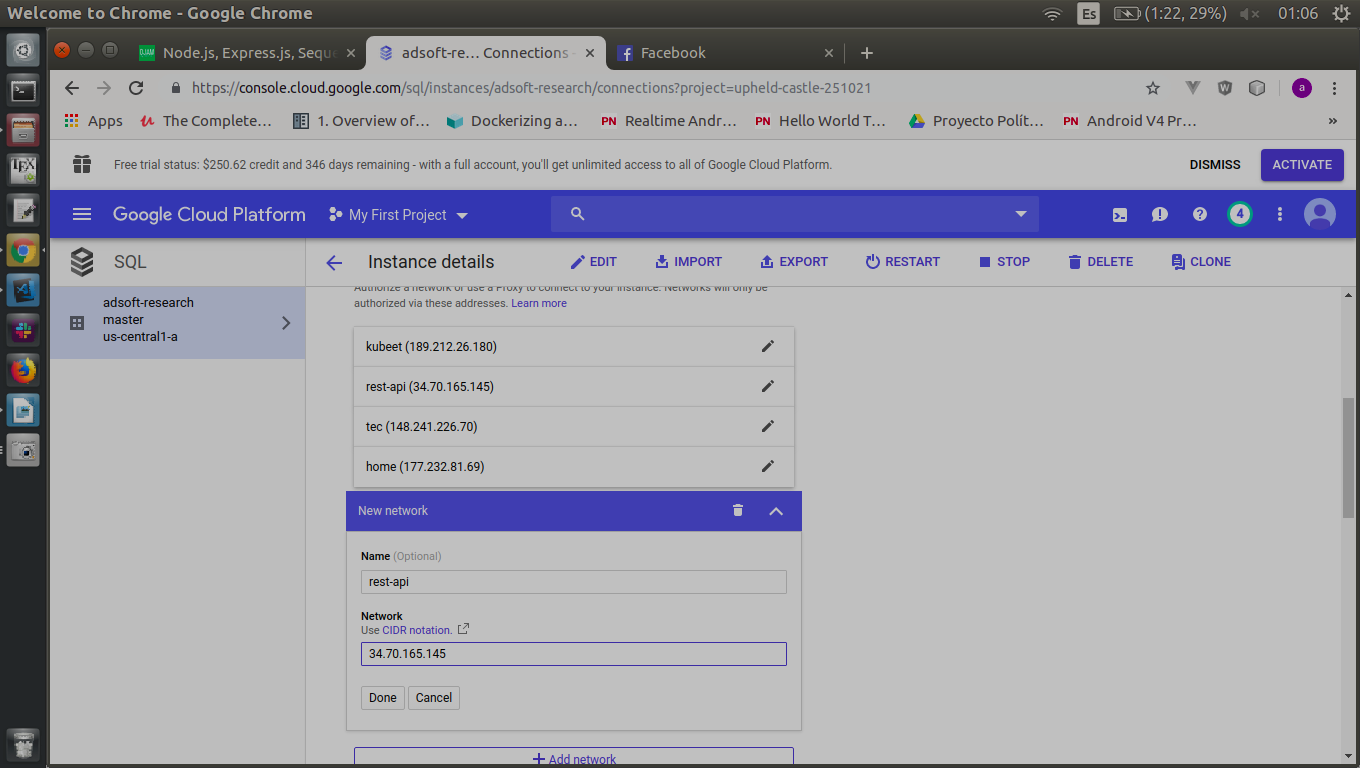
- Copiar direccion IP del servidor postgres, usuario, contraseña y nombre de base de datos.



- Abrir firewall de postgres a tu IP actual, obtener de <https://www.whatismyip.com/>



Click en Add Network



Click en Save

8.- Deploy

**- Agregar cabeceras a cada metodo, como el siguiente ejemplo**

function getEstados(req, res) {  
res.setHeader('Access-Control-Allow-Origin', '\*');  
res.setHeader('Access-Control-Allow-Methods', 'GET, POST, OPTIONS, PUT, PATCH, DELETE'); // If needed  
res.setHeader('Access-Control-Allow-Headers', 'X-Requested-With,contenttype'); // If needed  
res.header('Access-Control-Allow-Headers', 'Content-Type');try {  
console.log("Estados...");  
console.log(Estados);  
Estados.findAll({  
})  
.then((estados) => {  
console.log(estados);  
res.status(200).send(estados);  
}, (error) => {  
res.status(500).send(error);  
});  
} catch (error) {  
controllerHelper.handleErrorResponse(MODULE\_NAME, getEstados.name, error, res);  
}  
}

En raiz del proyecto crear archivo app.yaml con el siguiente código

*runtime: nodejs*

*env: flex*

*manual\_scaling:*

*instances: 1*

*resources:*

*cpu: 1*

*memory\_gb: 1.7*

*disk\_size\_gb: 10*

*# [START configuration]*

*endpoints\_api\_service:*

*# The following values are to be replaced by information from the output of*

*# 'gcloud endpoints services deploy openapi-appengine.yaml' command.*

*name: tuproyecto.appspot.com*

*#rollout\_strategy: managed*

*# Copyright 2015-2016, Google,*

*config\_id:* ***2019-09-11r0 # este config\_id se cambia de acuerdo a la fecha actual***

*# [END configuration]*

**- Actualizar archivo swagger.yaml**

*….*

*host: tuproyecto.appspot.com*

x-google-endpoints:- name: '***tuproyecto***[**.appspot.com**](http://valid-decoder-258800.appspot.com/)'allowCors: 'true'

*#host: localhost:10010*

*….*

9.- crear repositorio para tu proyecto rest api en github.com y subirlo.

*git init*

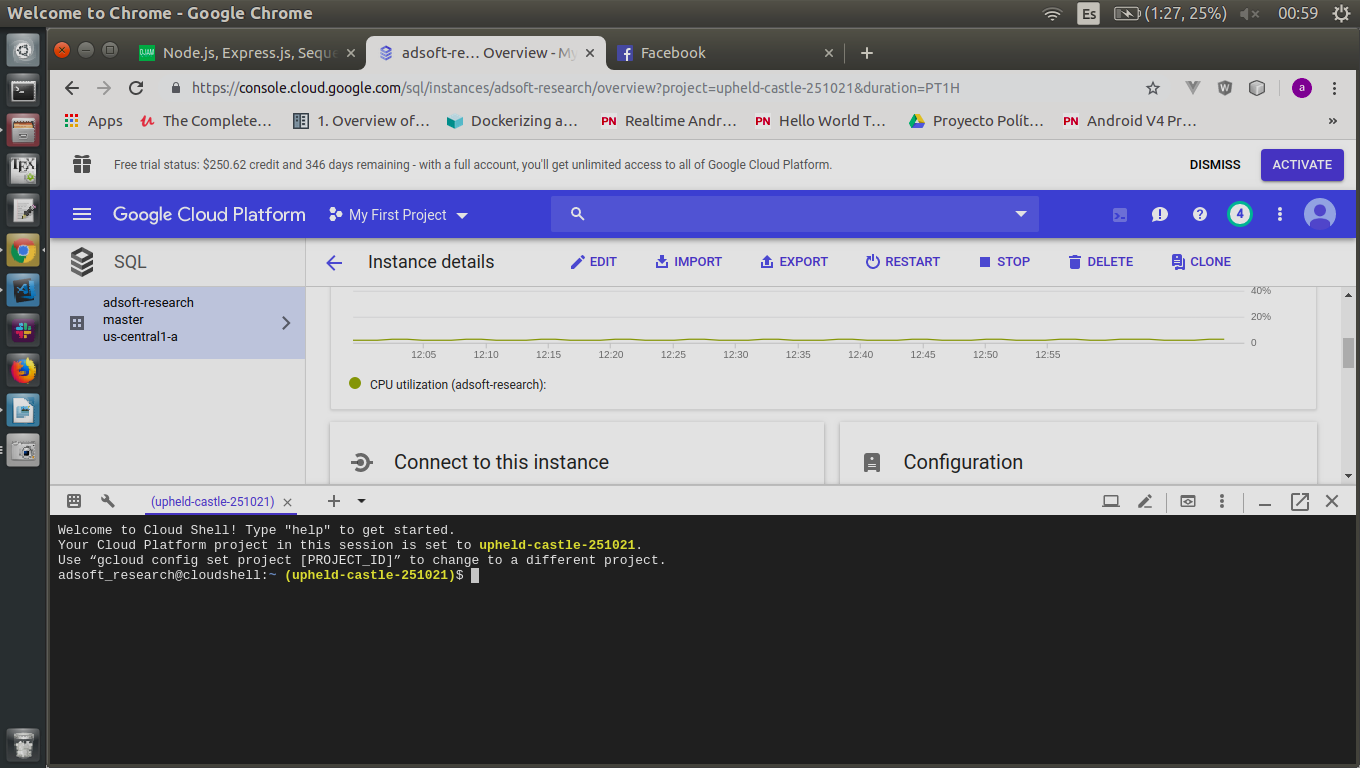
*git remote add origin https://github,com/tu-usuario/tu.api,git*

*git add .*

*git commit -m “init rest api”*

*git push -u origin master*

10.- Deploy en google app engine, abrir cloud shell



**- En cloud shell, clonar tu repositorio de rest api.**

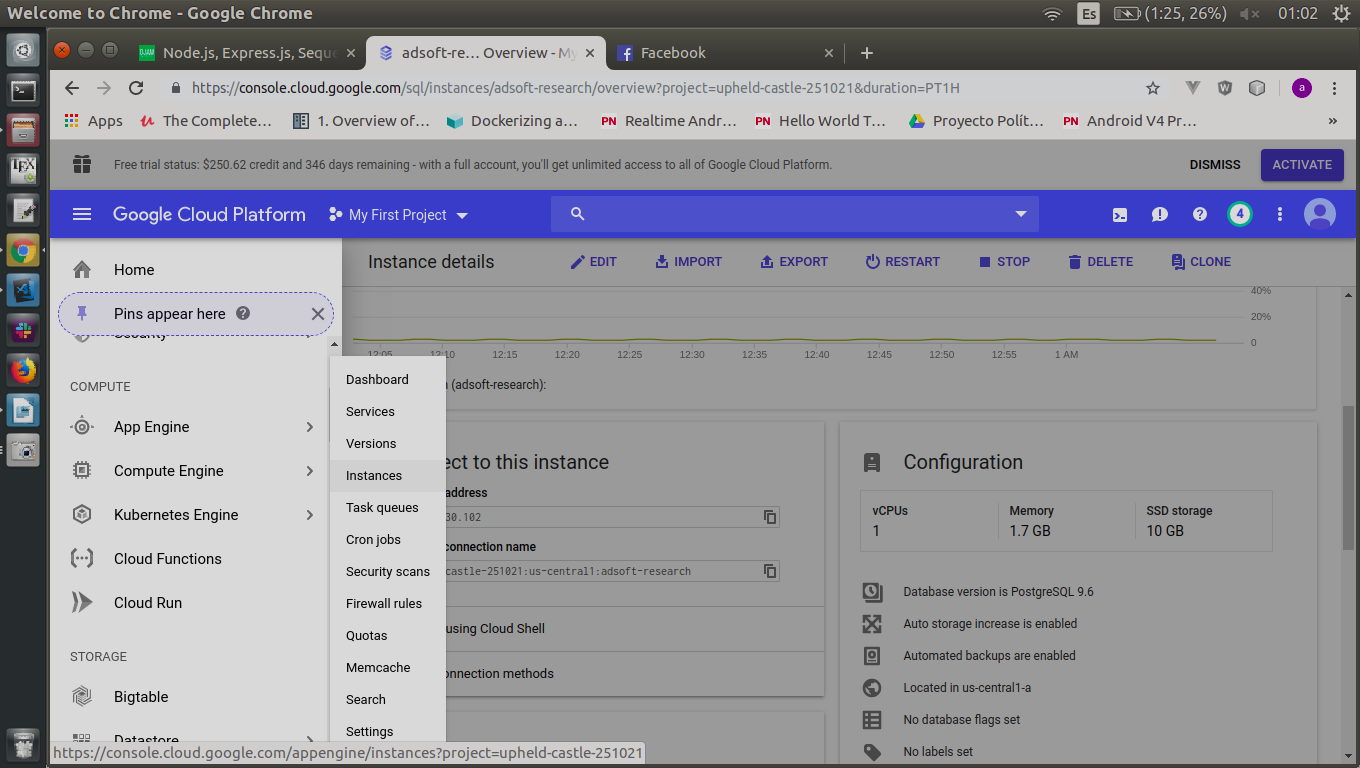
$ git clone [https://github.com/tu-usuario/tu-api.git](https://github.com/usuario/api.git)

*$ cd tu-api*

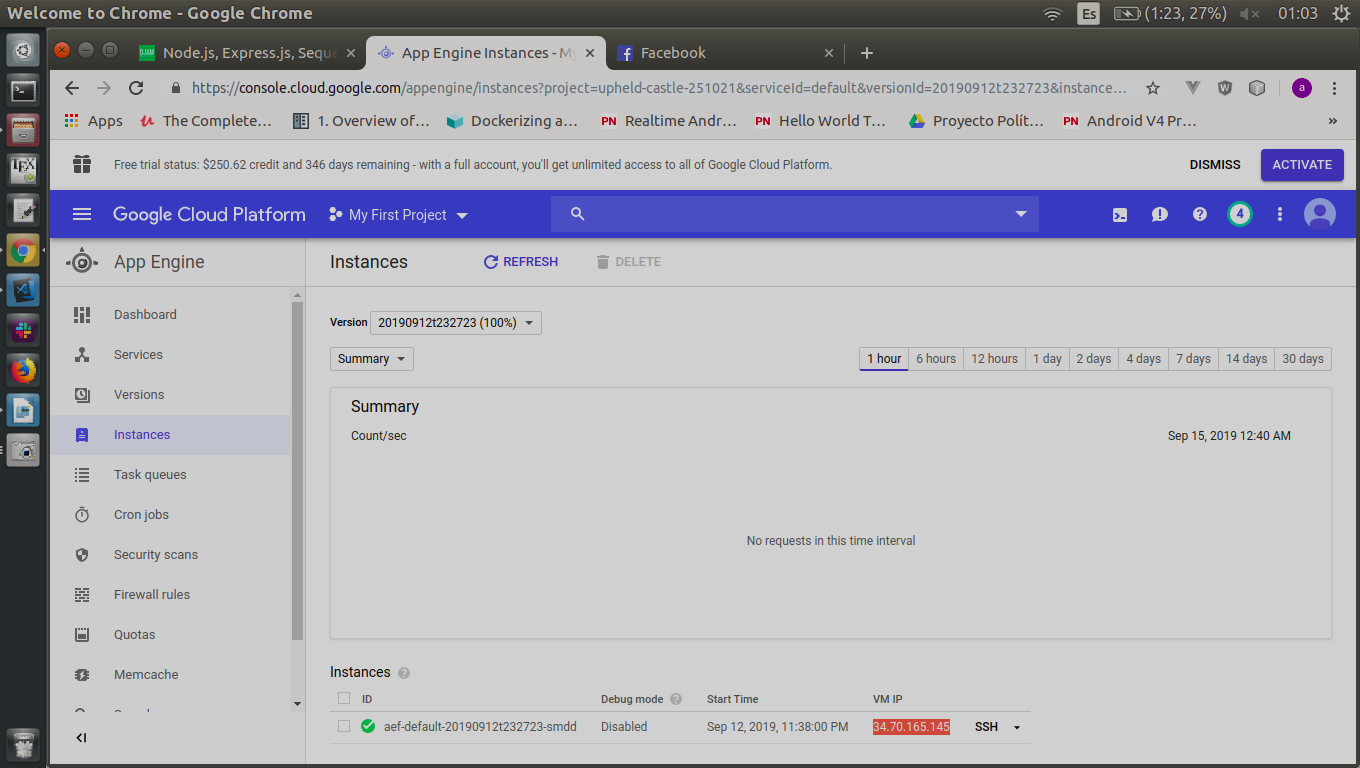
*$ gcloud endpoints services deploy api/swagger/swagger.yaml*

*$ gcloud app deploy*

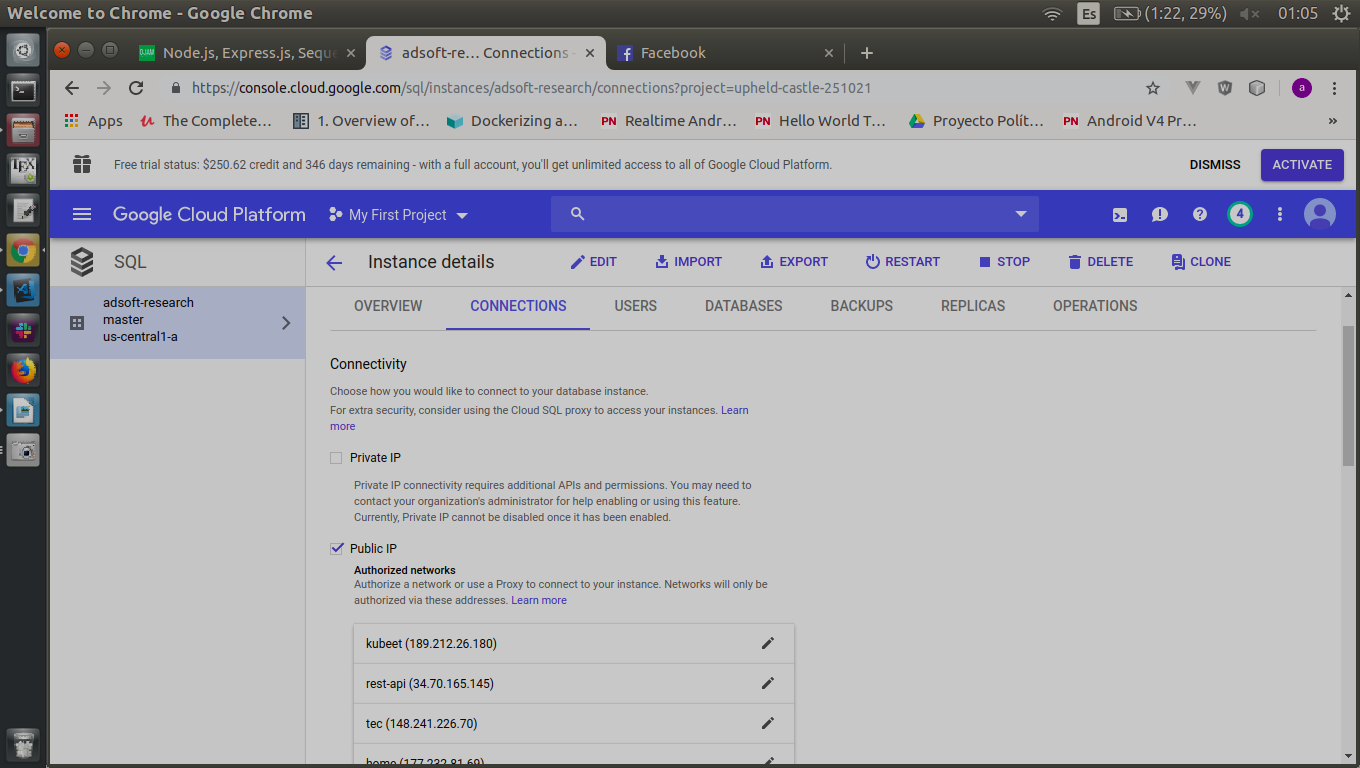
11.- Copiar la ip de tu instancia de rest api en Google App Engine y agregarla al firewall de PostgreSQL



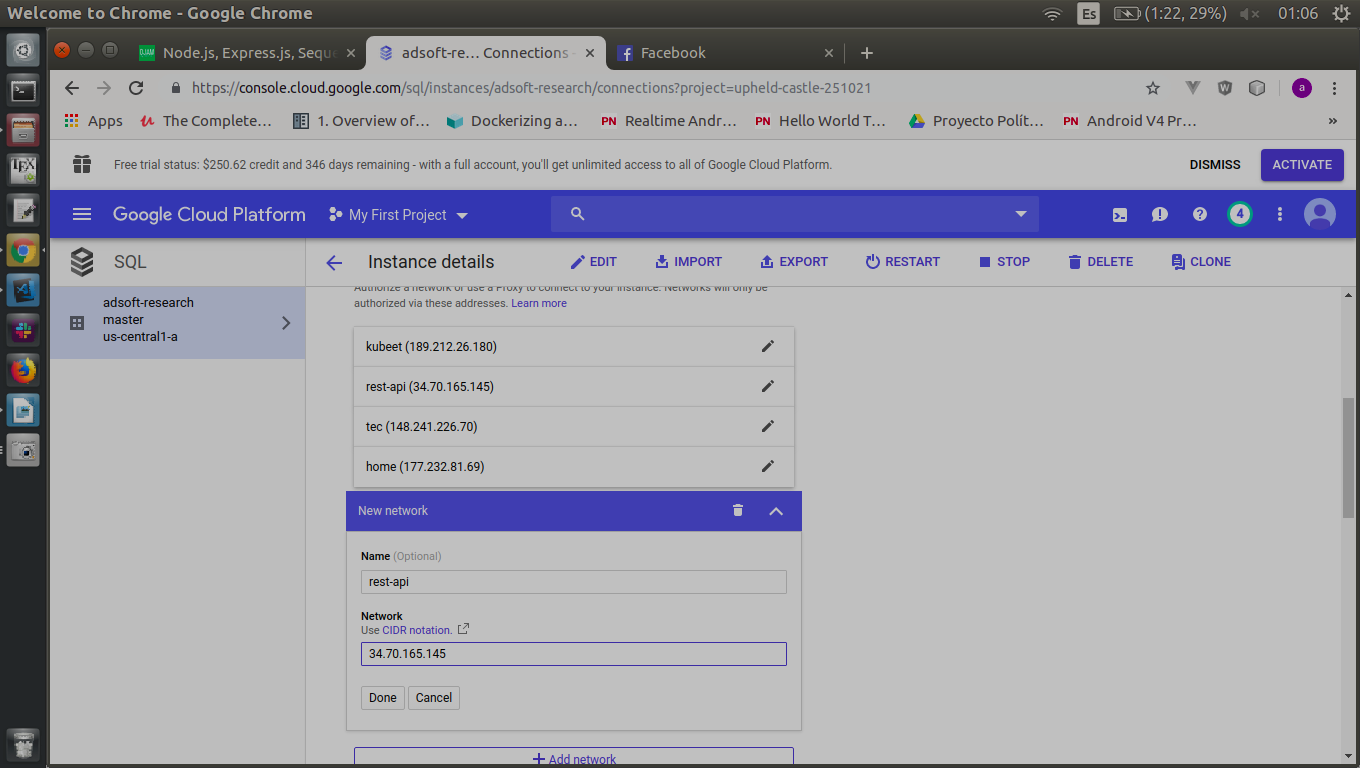
- Copiar ip de instancia



- Agregarla a ips permitidas en instancia de PostgreSQL



- Click en Add network



**- Click en Save**

**Deploy de fronted**

**- Crear un proyecto en firebase**

**- Ingresar a console.firebase.google.com**

***$ ng build***

***$ npm install -f firebase—tools***

***$ firebase login***

***$ firebase init***

***$ firebase deploy***