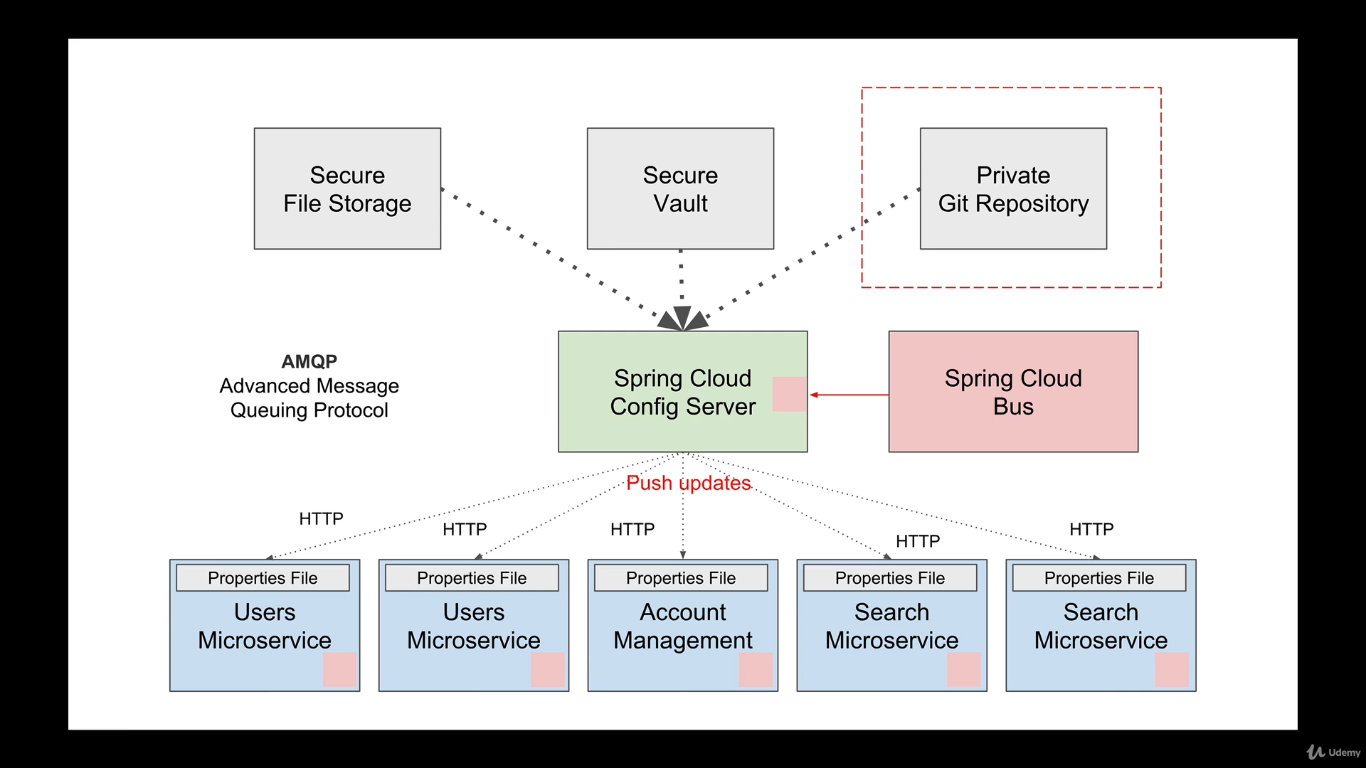
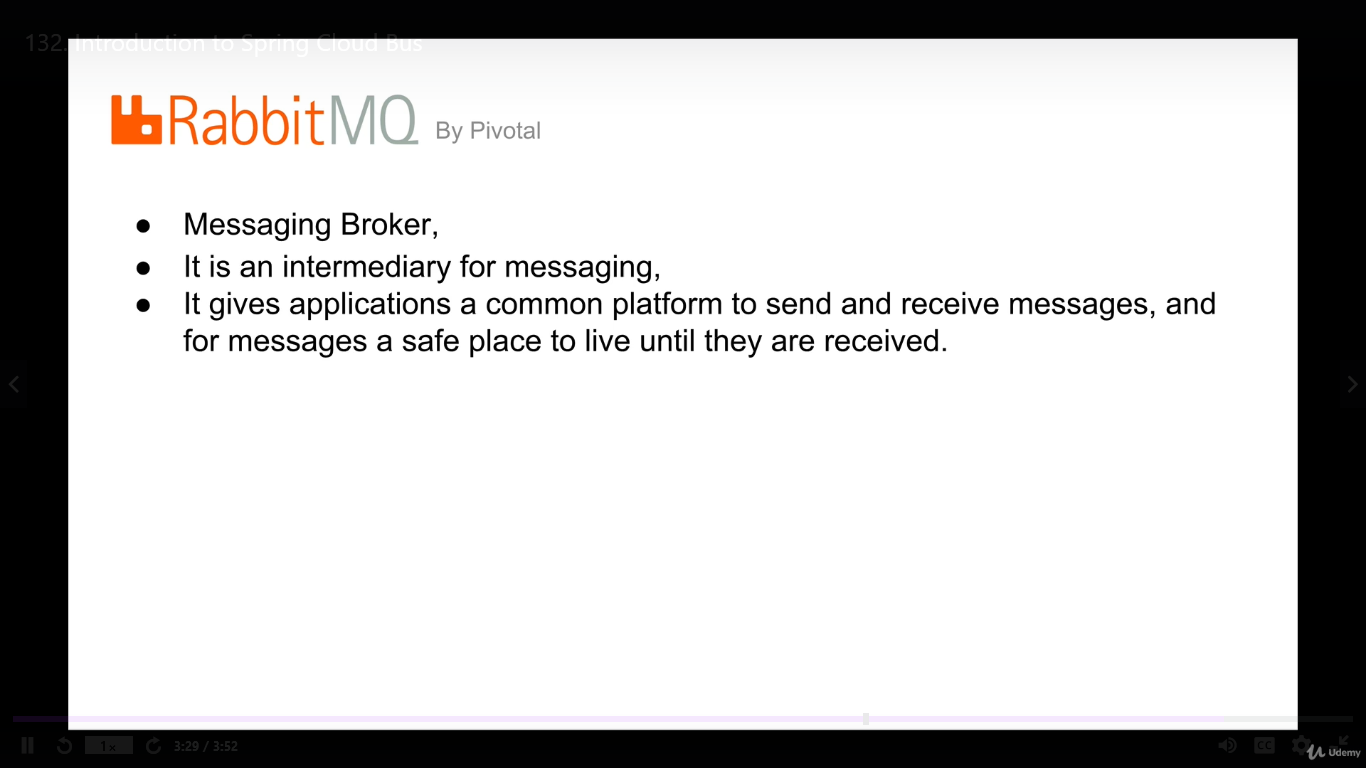
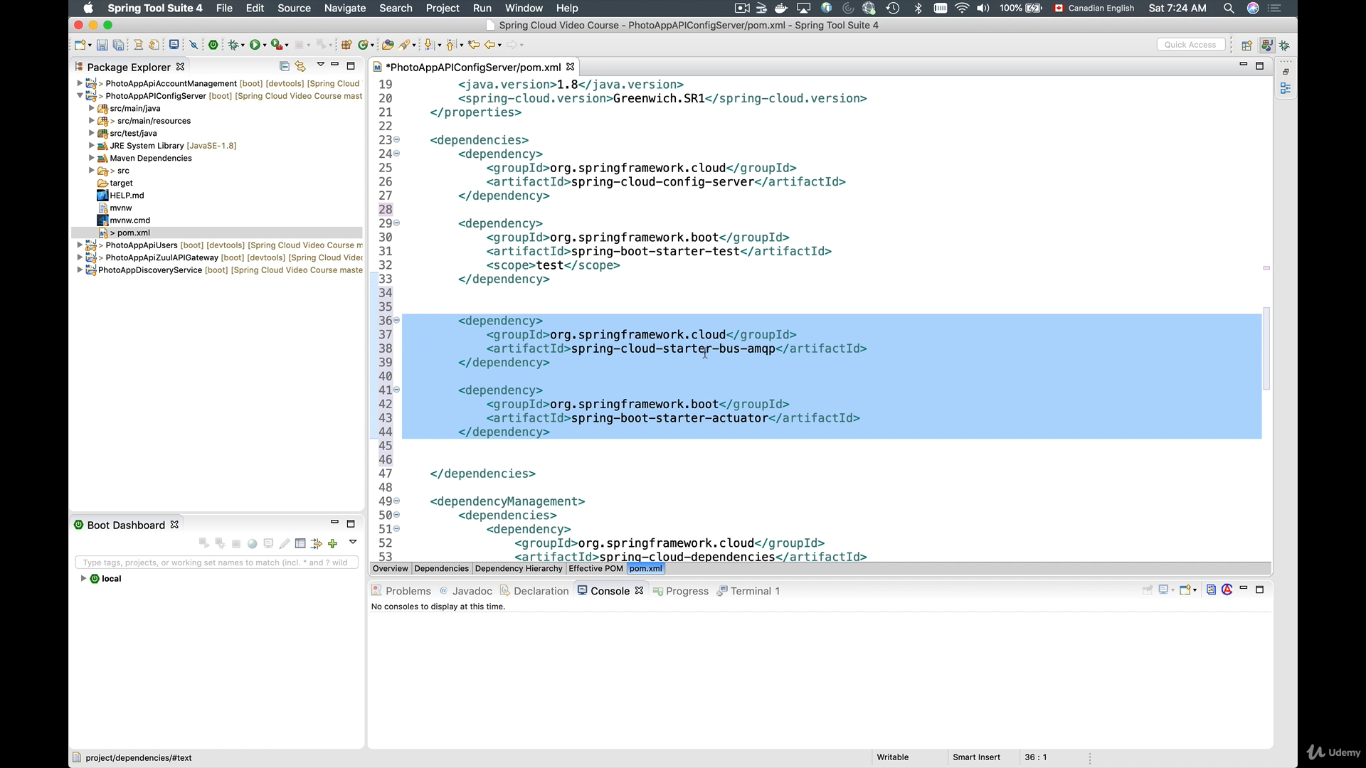
When the microservices use spring cloud config server to read the configurations it loads all the configuration only one time that is at the time of startup of the application. If the configurations are updated then it is not pushed to the microservices. To solve this issue we have the concept of spring cloud bus.



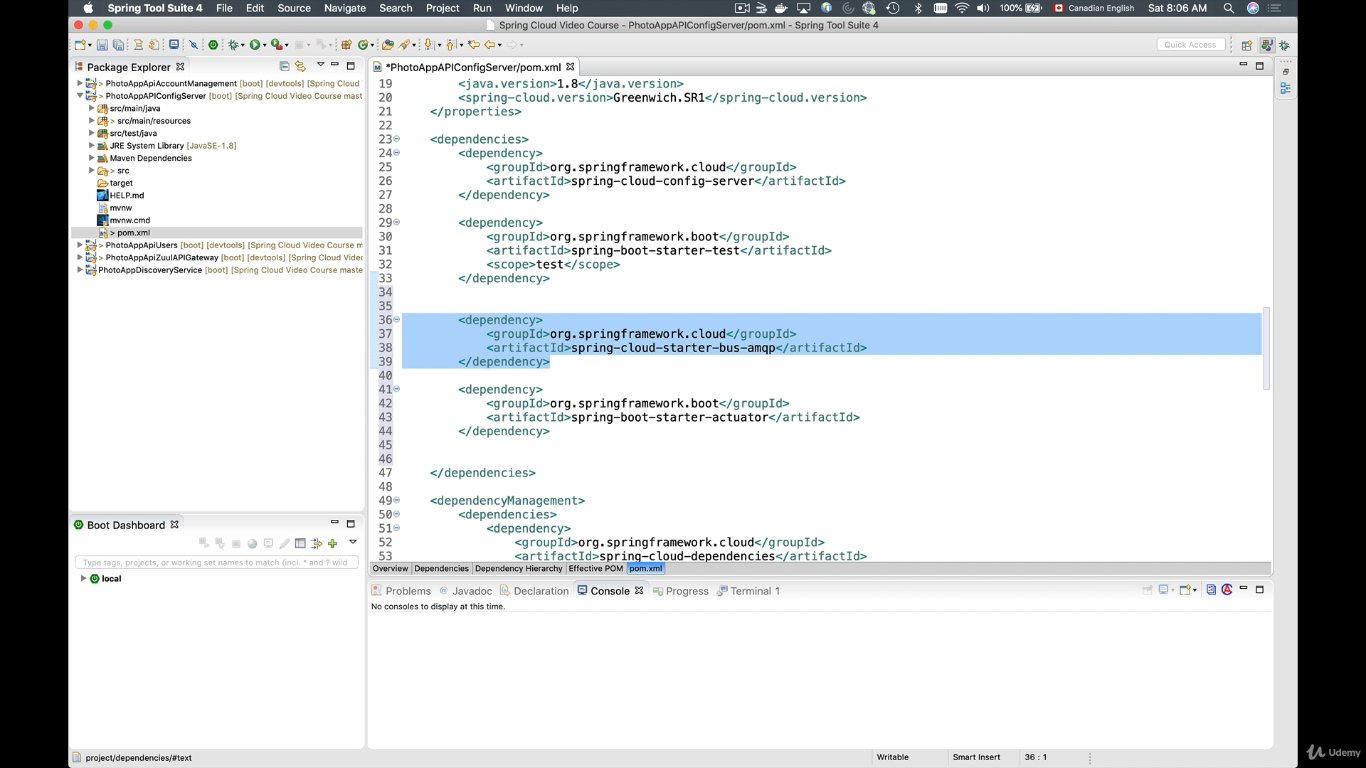
We don’t have to create a stand alone spring boot application to configure spring cloud bus but we can achieve it by making some changes in spring cloud config server and microservices which wants to read get updated when config file is updated.

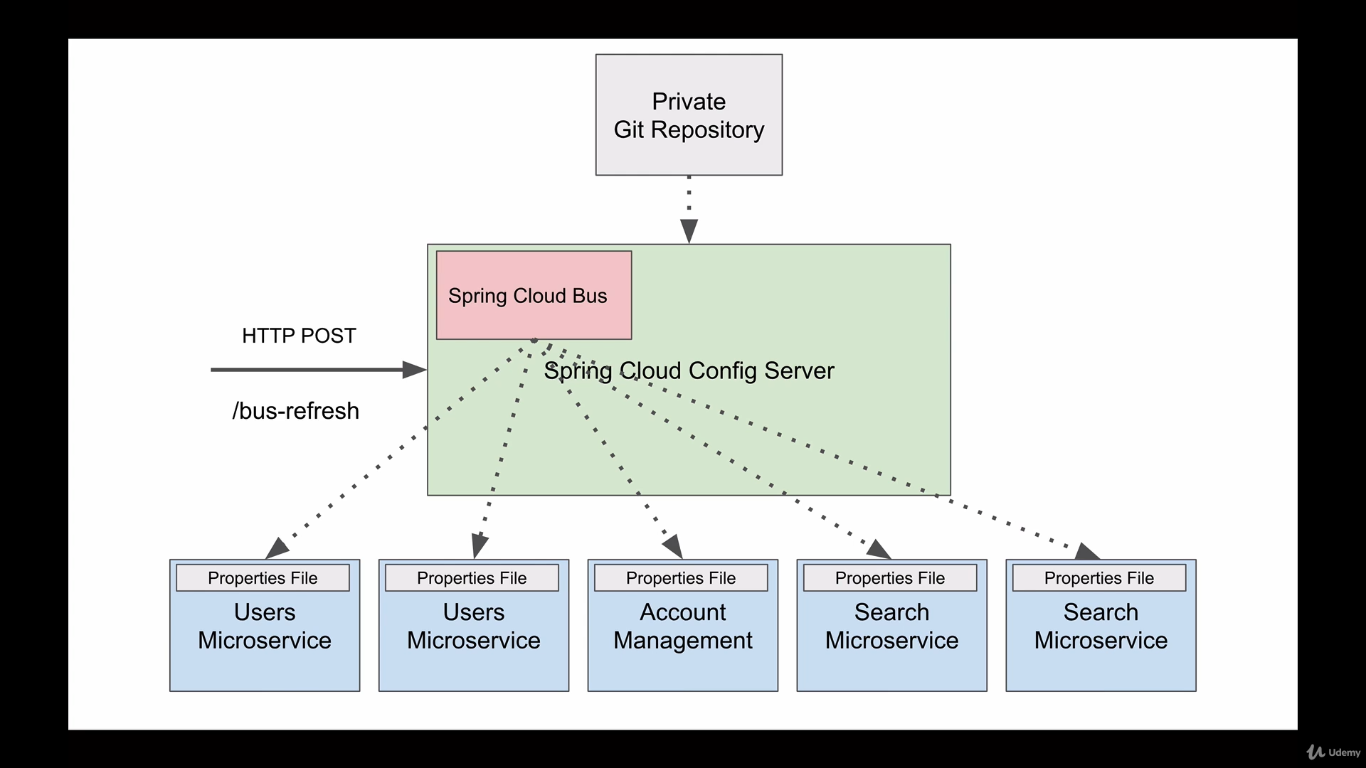


Add the following 2 dependencies in the config server.



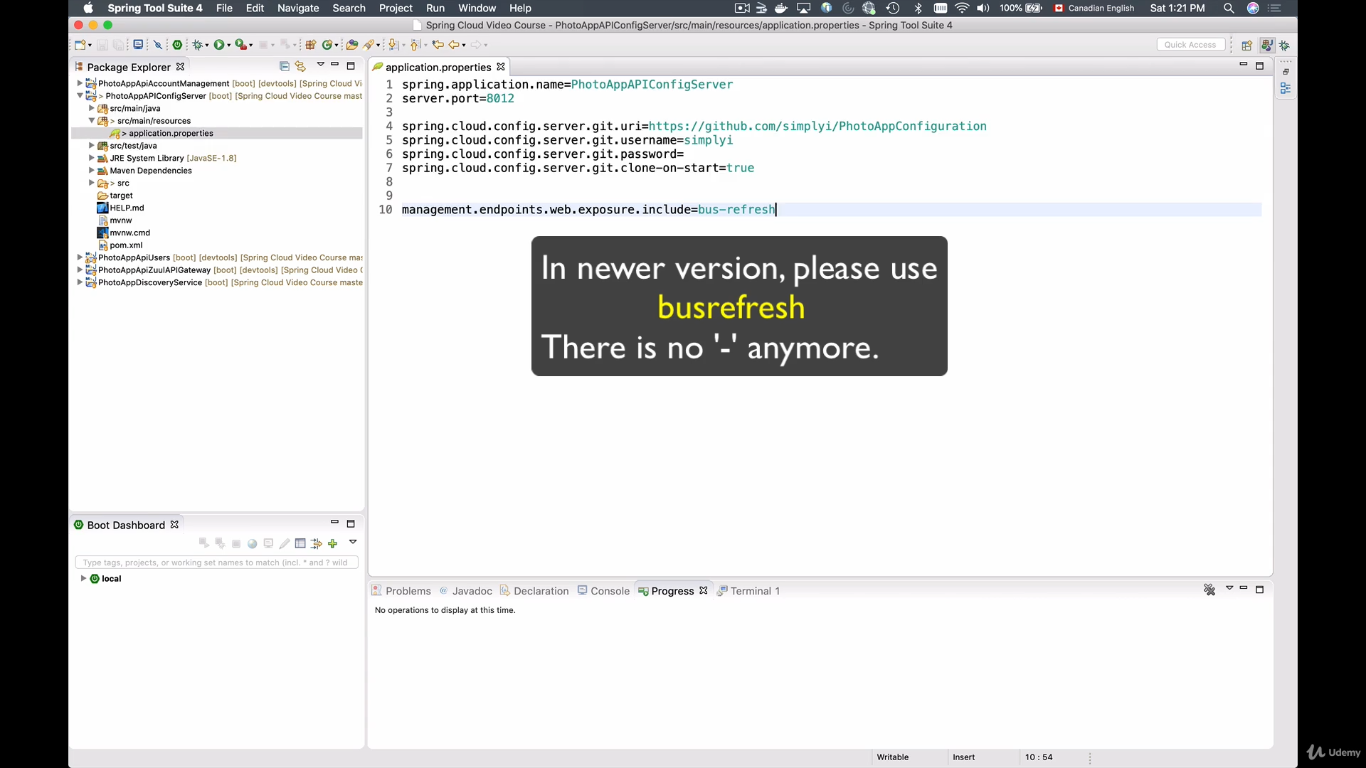
Add the following dependency in each of the microservices which want to receive the configuration dynamically.



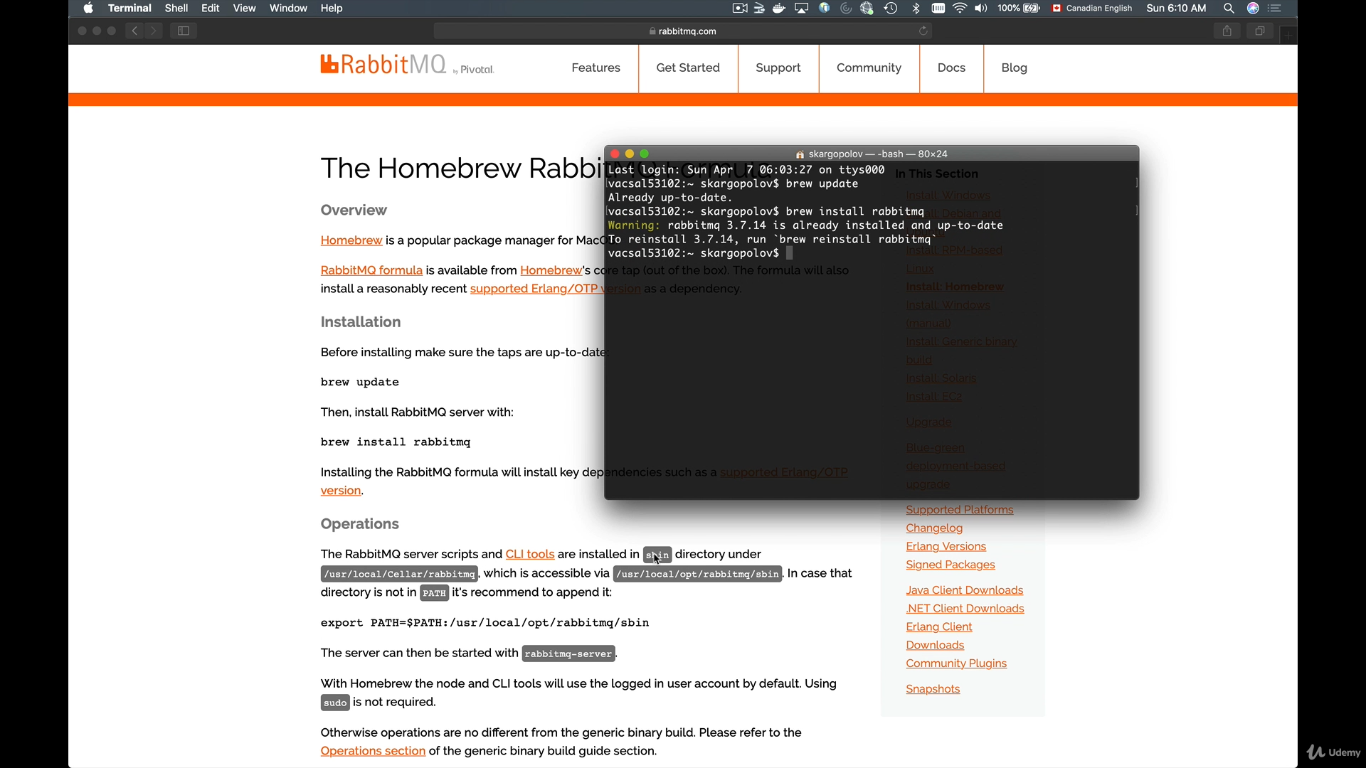


**Enable “/bus-refresh” endpoint.**

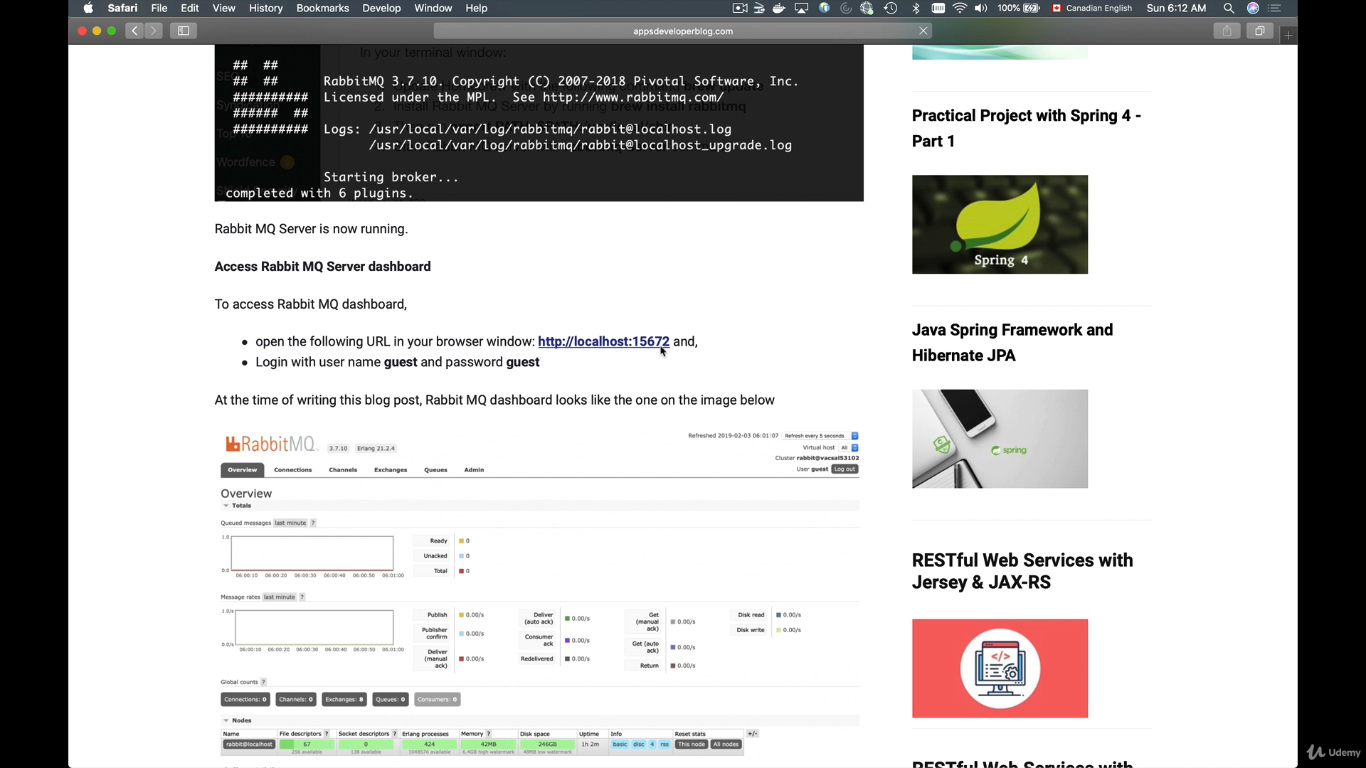
The moment we update the configuration in centralised repository we have to inform the spring cloud bus to broadcast the information to all its client microservices by sending a post request to /bus-refresh.



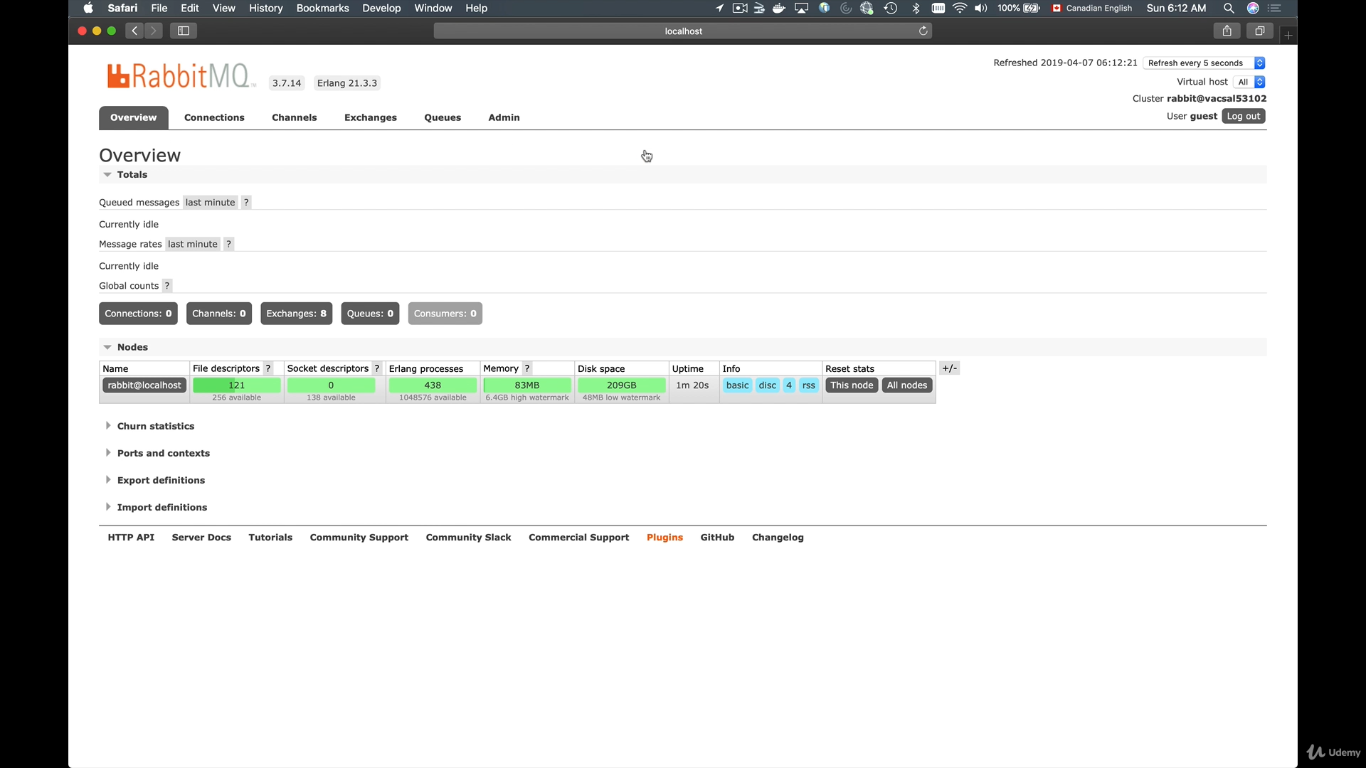
By adding this configuration in properties file we refresh the config server with latest changes.



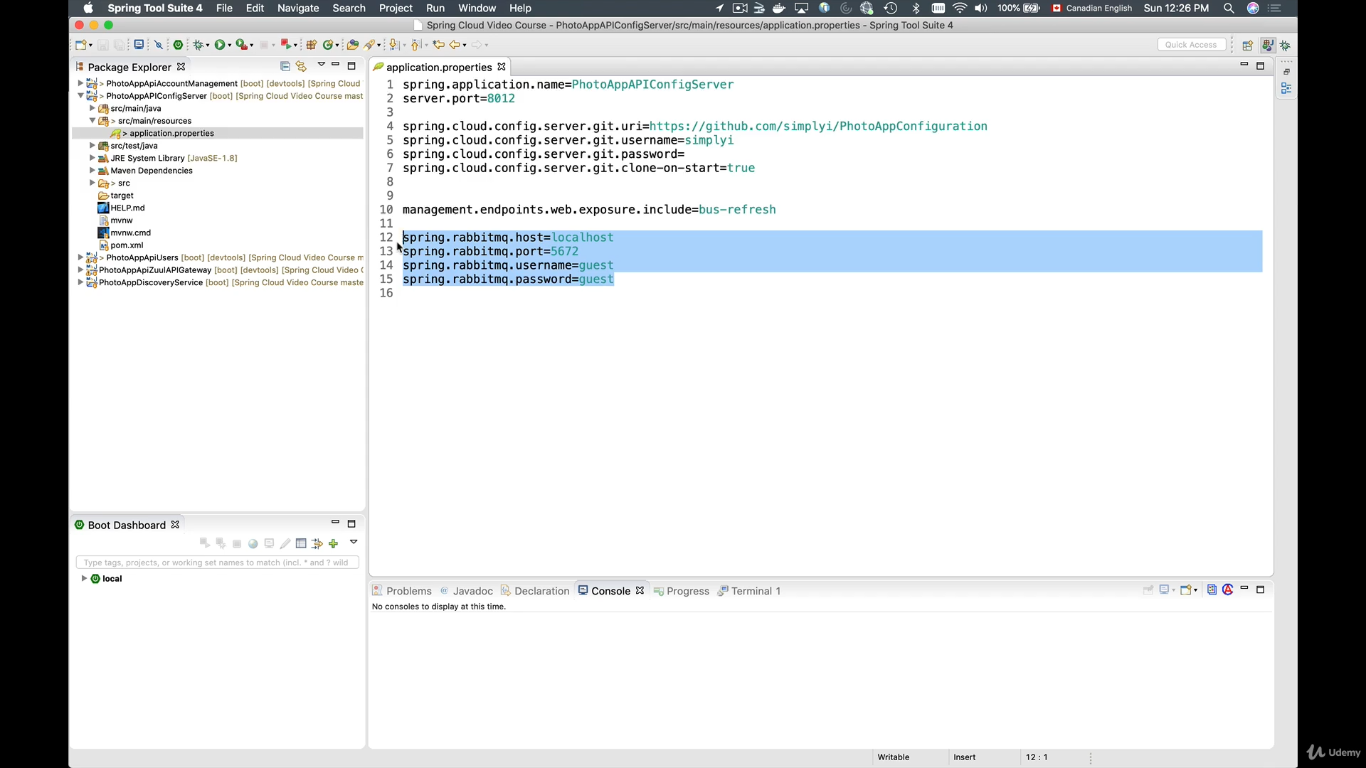
Download and run rabbit MQ.



Default username and password is “guest”.

rabbit MQ server is running.

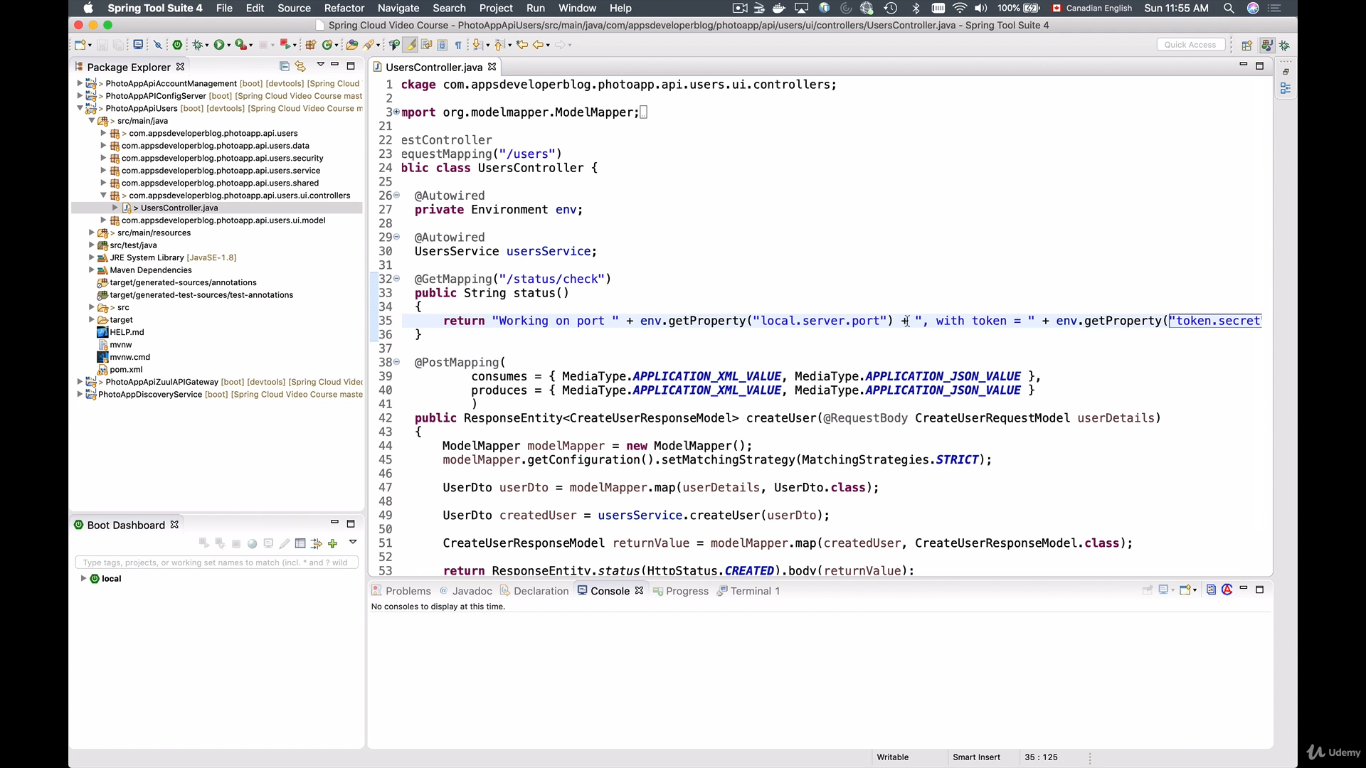
Add the following configurations to communicate with RabbitMq.



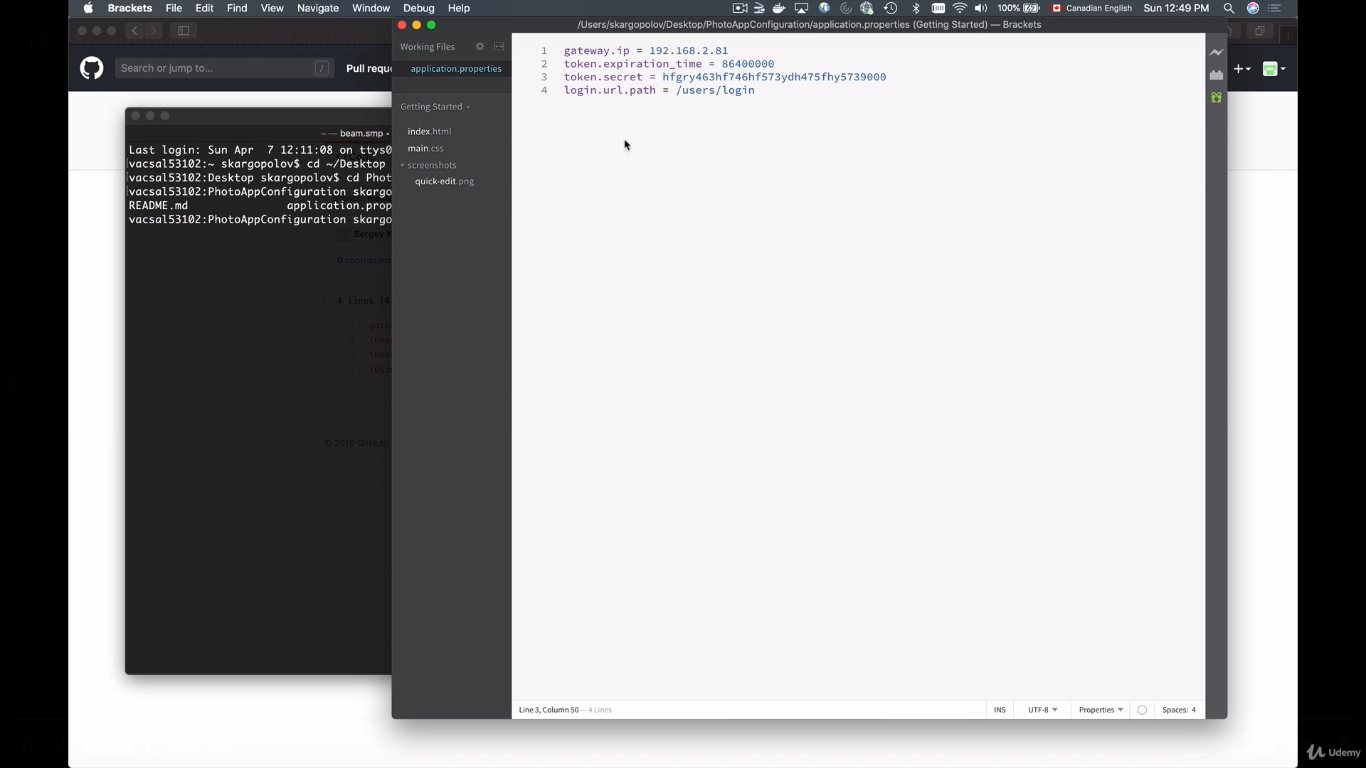
Add in config server and all the microservices which want to communicate with RabbitMQ.

Now run the application and test how spring cloud bus is working.

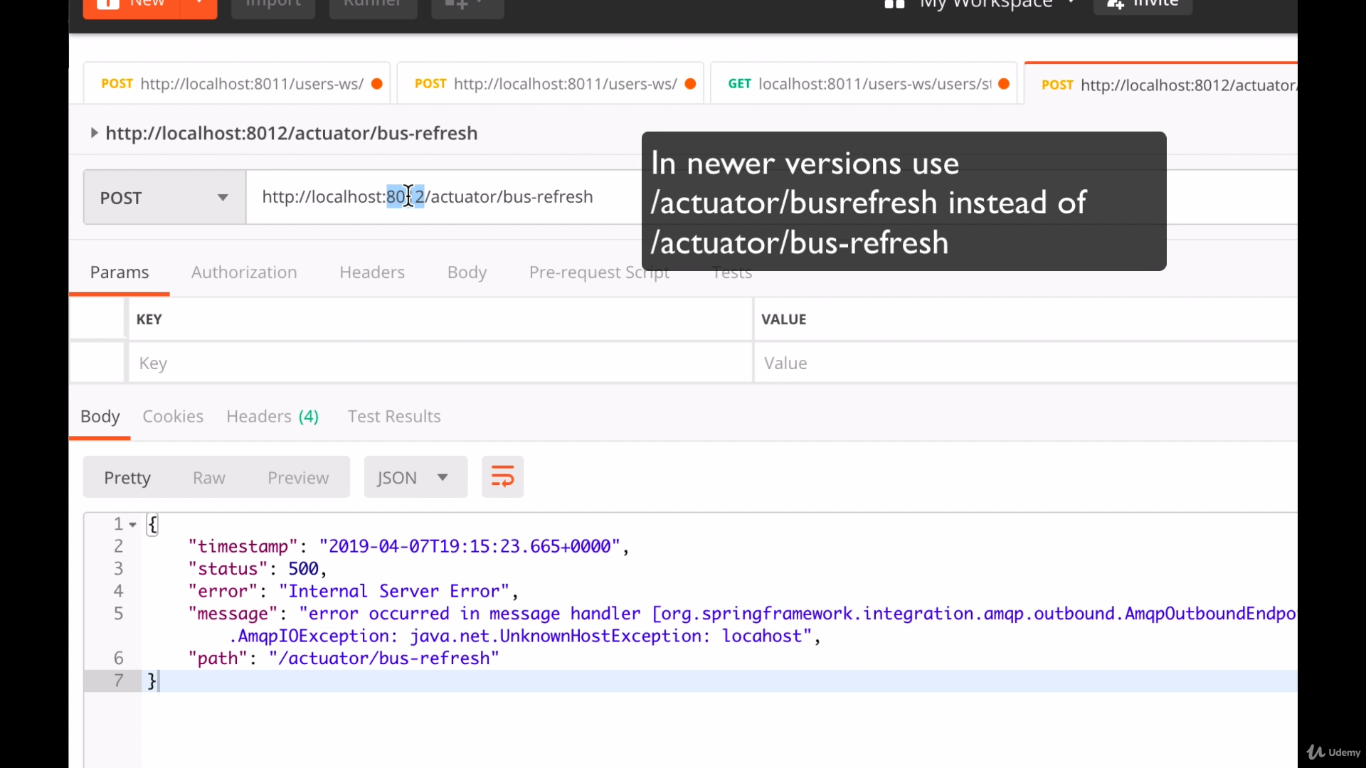
1. Register a user
2. Login with the registered user
3. And check the jwt signingkey
4. Update the signing key and then check whether the RabbitMQ is broadcasting the message to the other microservices.



The above screen we will print the secret key written in local properties file.

update the value and test

Add, commit, push.



To make the new settings effective.