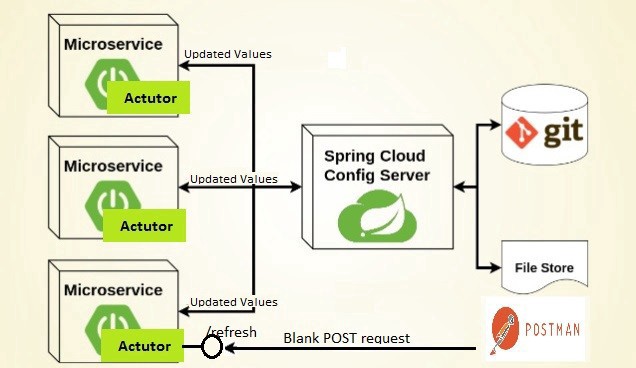
**Spring Boot Cloud Config Server**

**Benefits:-**

1. Centralized repository.
2. No redeployment
3. Fail fast configuration.



we will create a configuration server and a client that reads your configuration of that server. The configurations server uses a GIT repository on GitHub, where we store the configuration files.

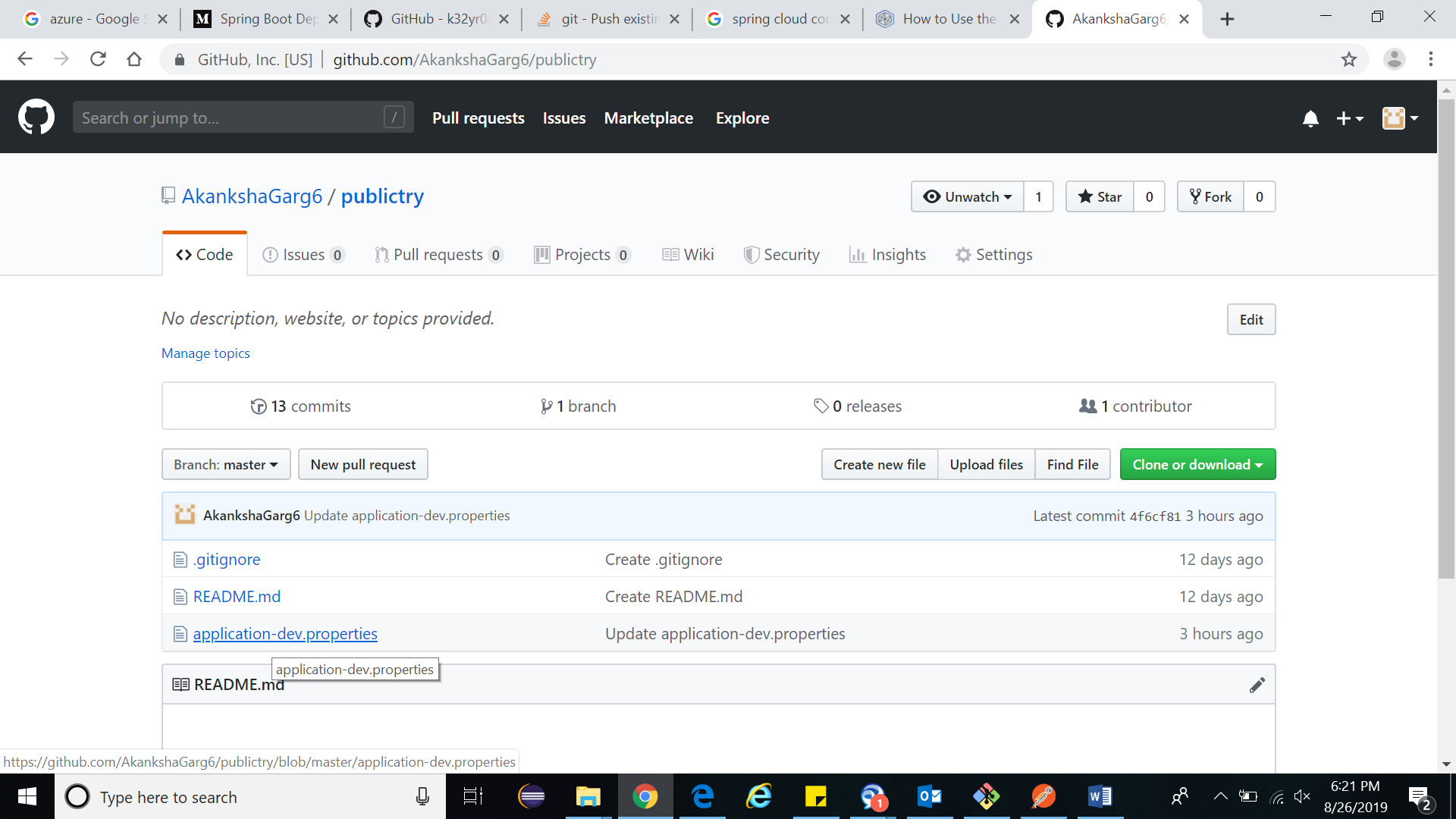
**Configuration Server**

* **Project**: config-server
* **Port**: 8888 // Any port number
* **Spring Name**: config-server

**Configuration Client**

* **Project**: config-client
* **Port**: 8080 // Any port number available
* **Spring Name**: config-client
* **Configuration server**

1. First, we need to create a remote git repository where we will create a file i.e. application.properties and will put all the configuration here.



1. Create a maven project that will be config server and include this dependency in it.

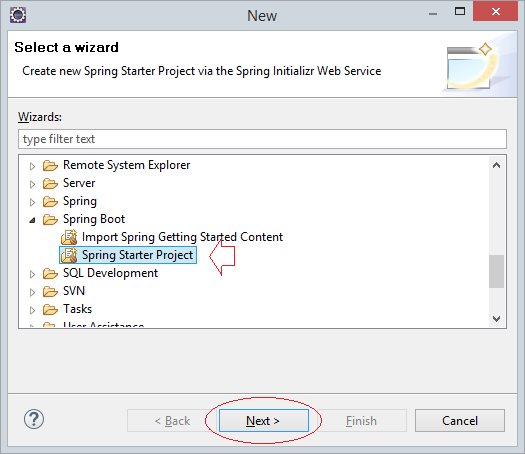
<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-config-server</artifactId>

</dependency>

The name of the Spring Starter is the**Config Server:**



1. The server configuration consists of a single class, which is detailed below:

package com.profesorp.configserver;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.cloud.config.server.EnableConfigServer;

@EnableConfigServer

@SpringBootApplication

public class ConfigServerApplication {

public static void main(String[] args) {

SpringApplication.run(ConfigServerApplication.class, args);

}

}

The only thing to highlight here is the label @EnableConfigServer.

1. In the file bootstrap.properties , these parameters should be provided :-

spring.application.name=config-server

server.port=8888

#If Private Repo then can give the credential :-

spring.cloud.config.server.git.username=AkankshaGarg6

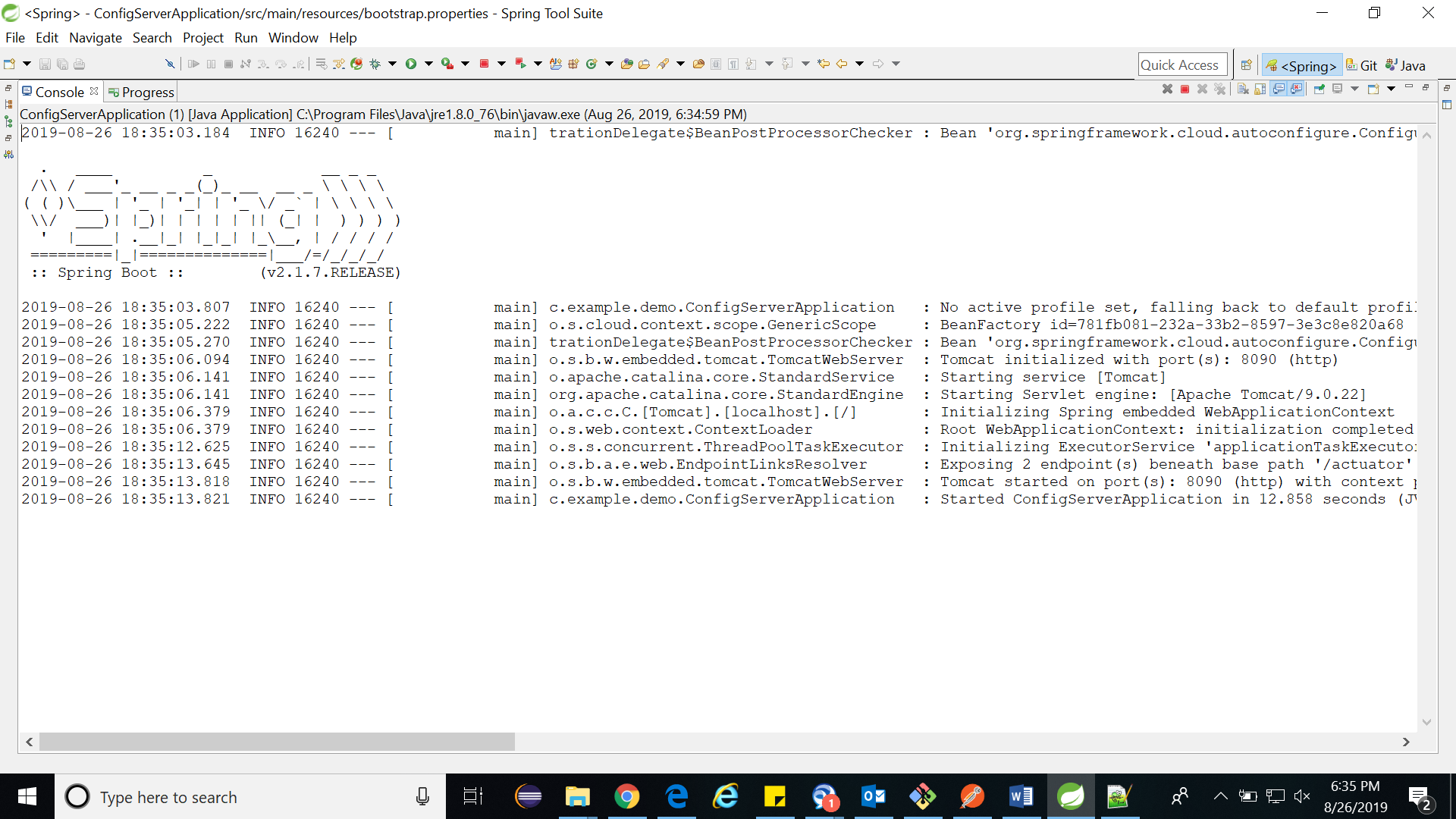
spring.cloud.config.server.git.password=

spring.cloud.config.server.git.uri=https://github.com/AkankshaGarg6/publictry.git

#If you have create a file inside a folder in git repo :-

spring.cloud.config.server.git.searchPaths= configuration // folder name

1. After that start the config server application :-



1. Now, Create a maven project that will be config client application and add this dependency to it.

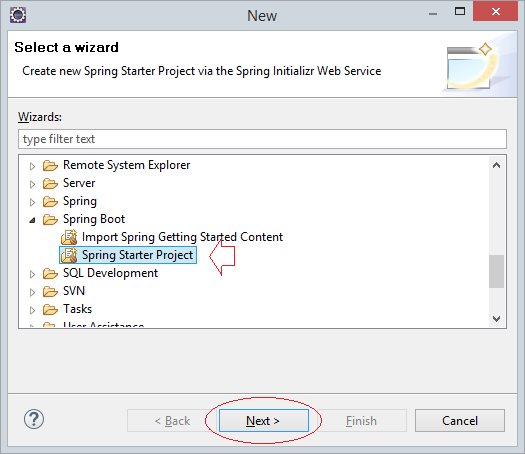
<dependency>

<groupId>org.springframework.cloud</groupId>

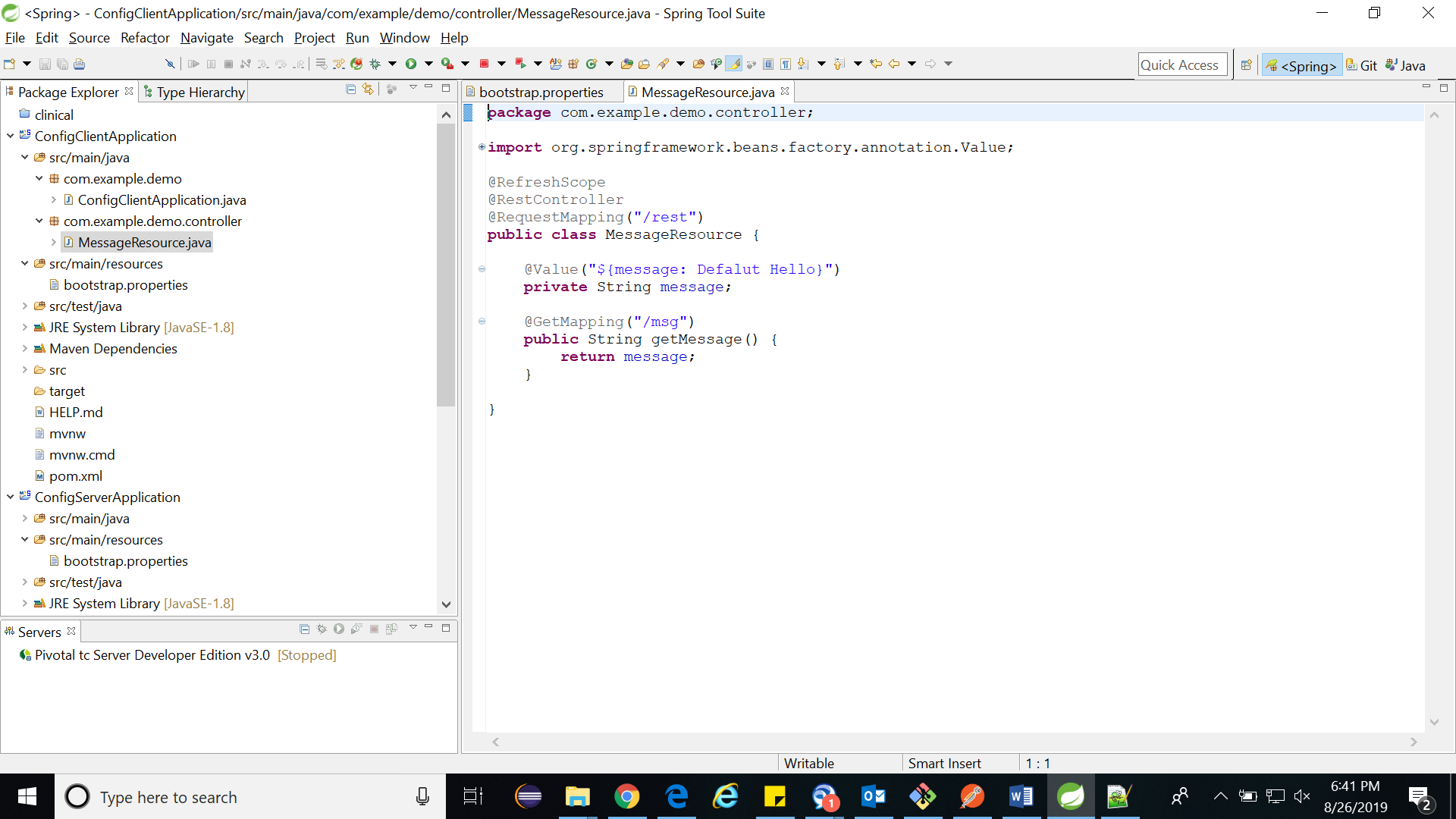
<artifactId>spring-cloud-starter-config</artifactId>

</dependency>

The name of the Spring Starter is the**Config client.**



1. **Create a controller class in config client and add your lines of code there.**



1. In the file bootstrap.properties , these parameters should be provided :-

spring.application.name=config-client

spring.profiles.active=dev //Profile name if you want to provide

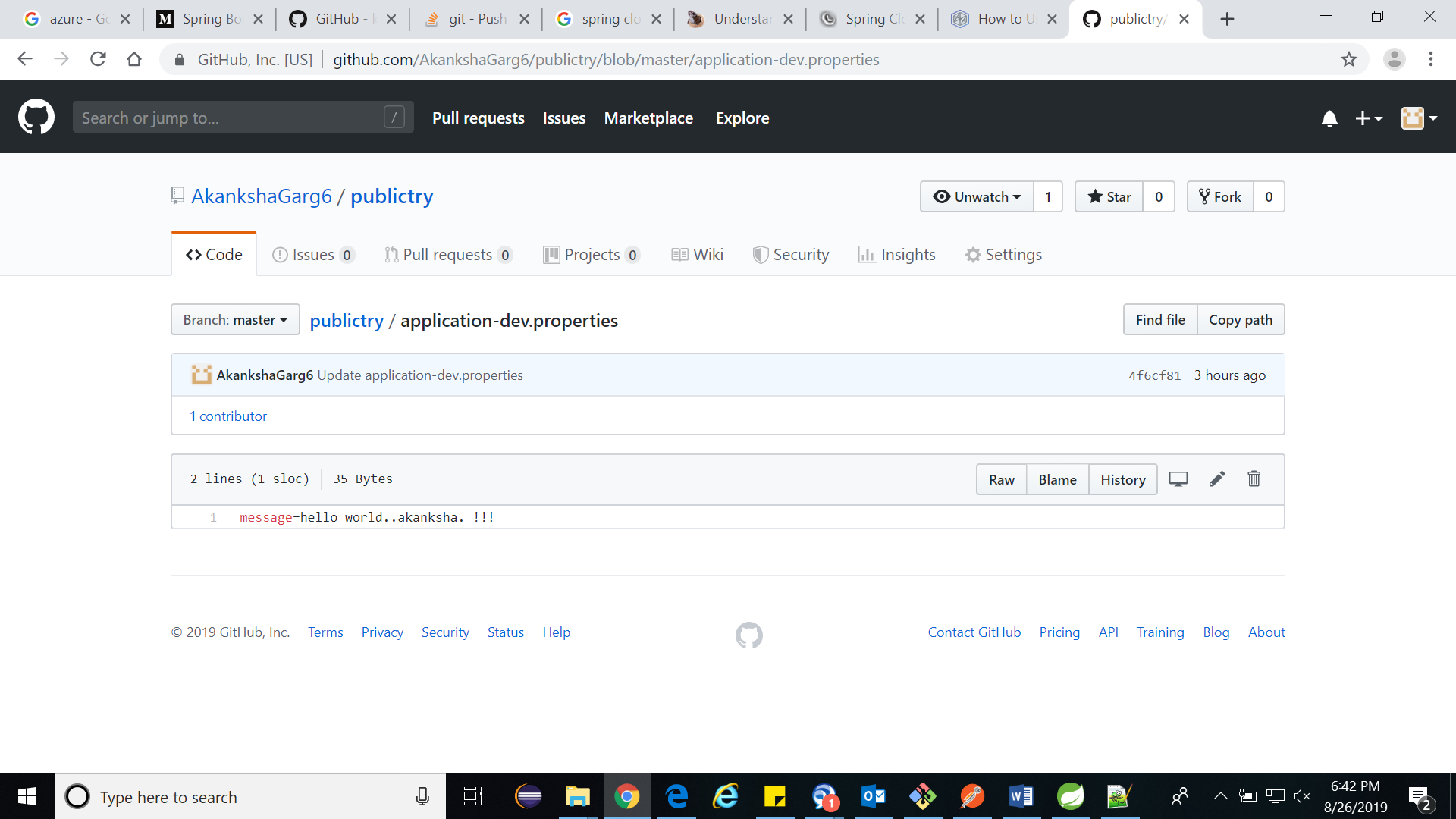
management.endpoints.web.exposure.include=\*

#Config server port number from where it will read the properties

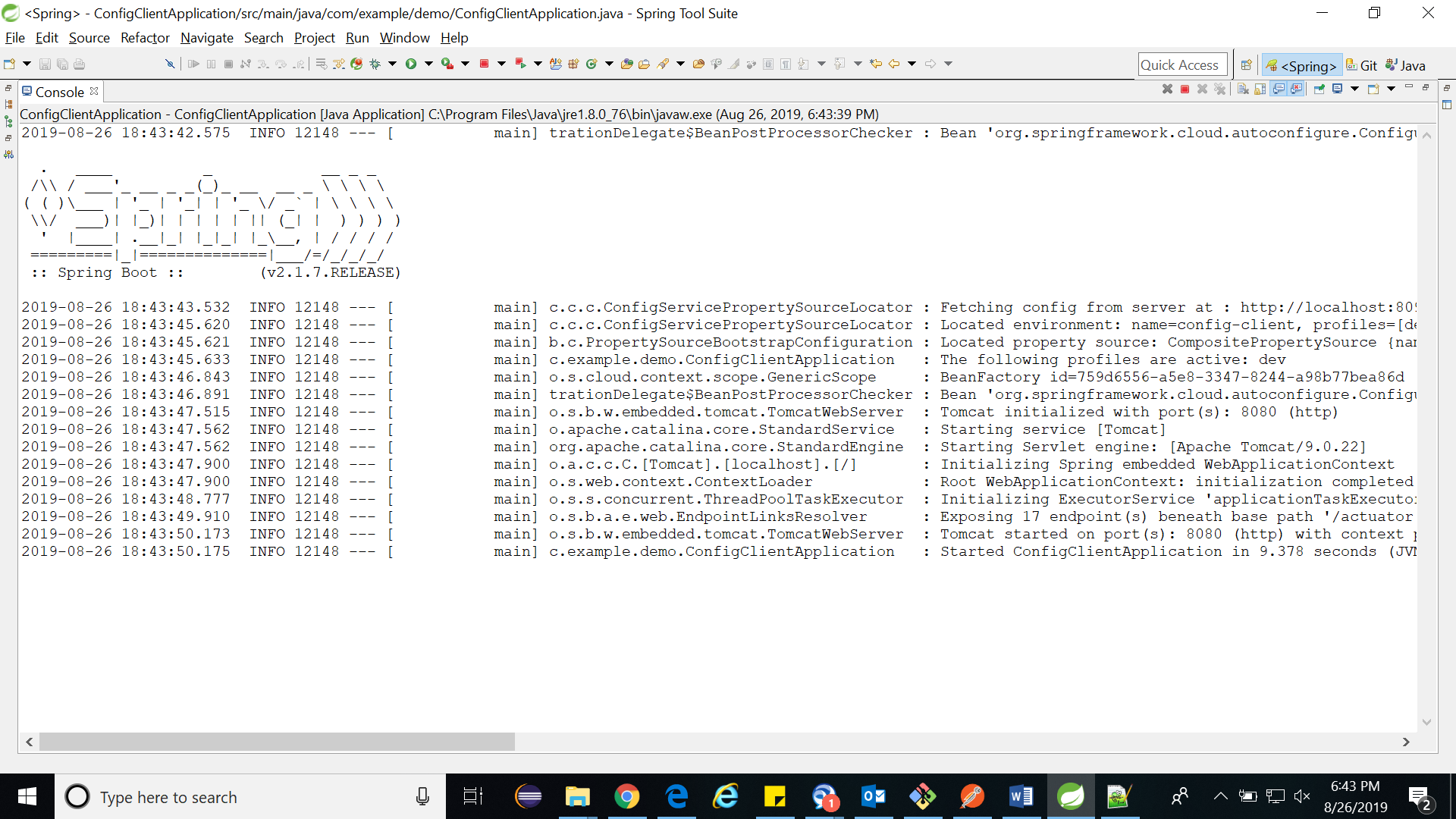
spring.cloud.config.uri=http://localhost:8090

endpoints.actuator.enabled=true

**Note :- Message is the variable whose value we have given in application.properties in git remote repository.**



1. **Then start the application i.e. Config client application.**

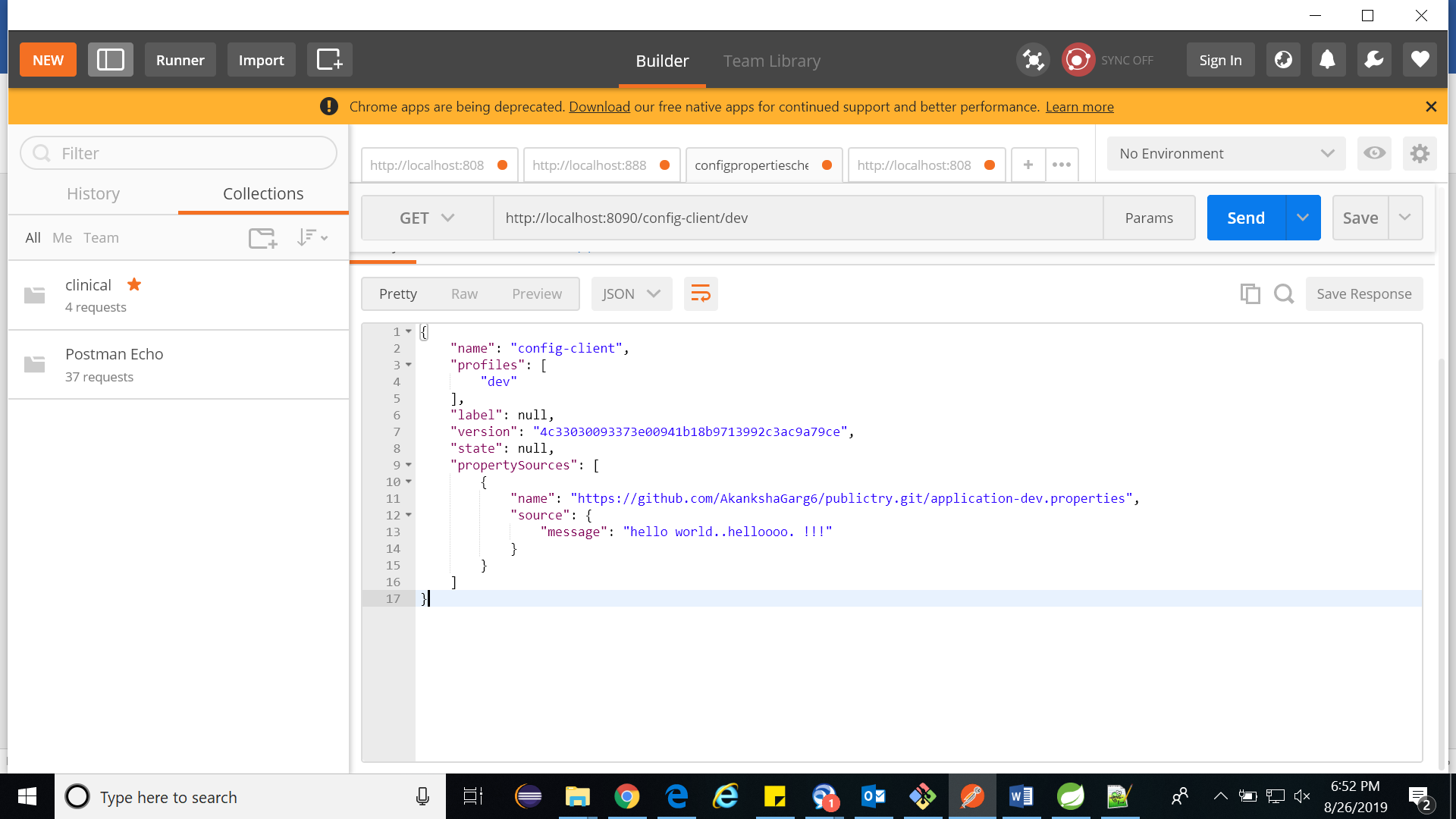


1. **Go to the POSTMAN , Hit the URL to check the properties that you have configured on application.properties in remote git repository**

<http://localhost:8090/config-client/dev>

* **/config-client --- application name of config client given in bootstrap.properties.**
* **/dev ---- if the profile in config client is DEV else DEFAULT.**

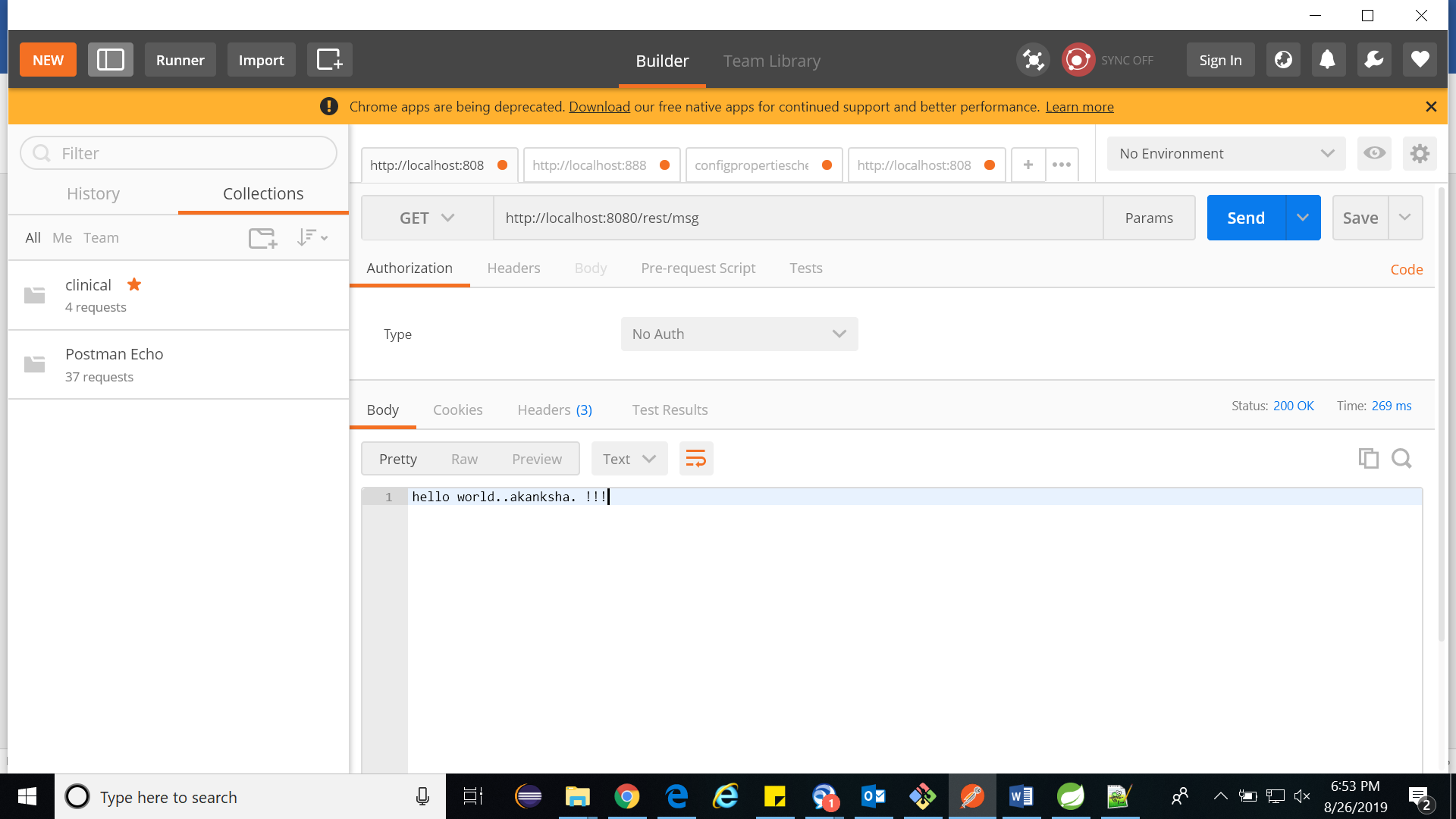
1. **Output will be :-**



1. **Then hit the URL for cnfig client application :-**

<http://localhost:8080/rest/msg>

1. **Output will be :-**



**So it is reading the property file from remote git repository. 😊**

**Role of Actuator**

**Add the actuator dependency in your pom.xml of config client application and also add @RefreshScope annotation to your controller class then we will not have to redeploy the application everytime after making the modification in application.properties file in git repo :-**

**Note :- This dependency work for Variables values modification that are declared if you have made the modification in DB Credential or change in port number then you will have to redeploy the application manually.**

<dependency>

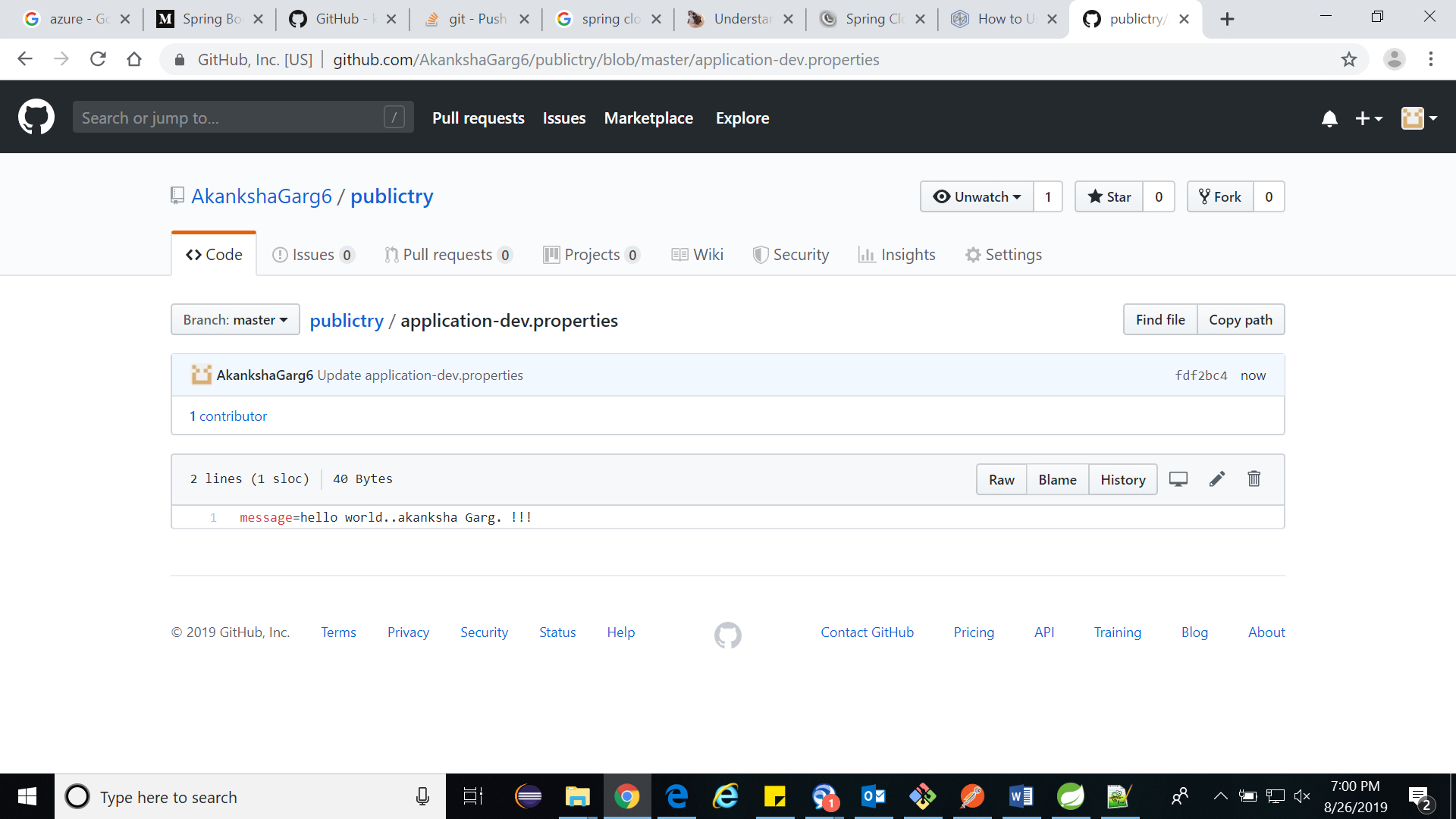
<groupId> org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

1. **Then Change your application.properties file in git repo :-**

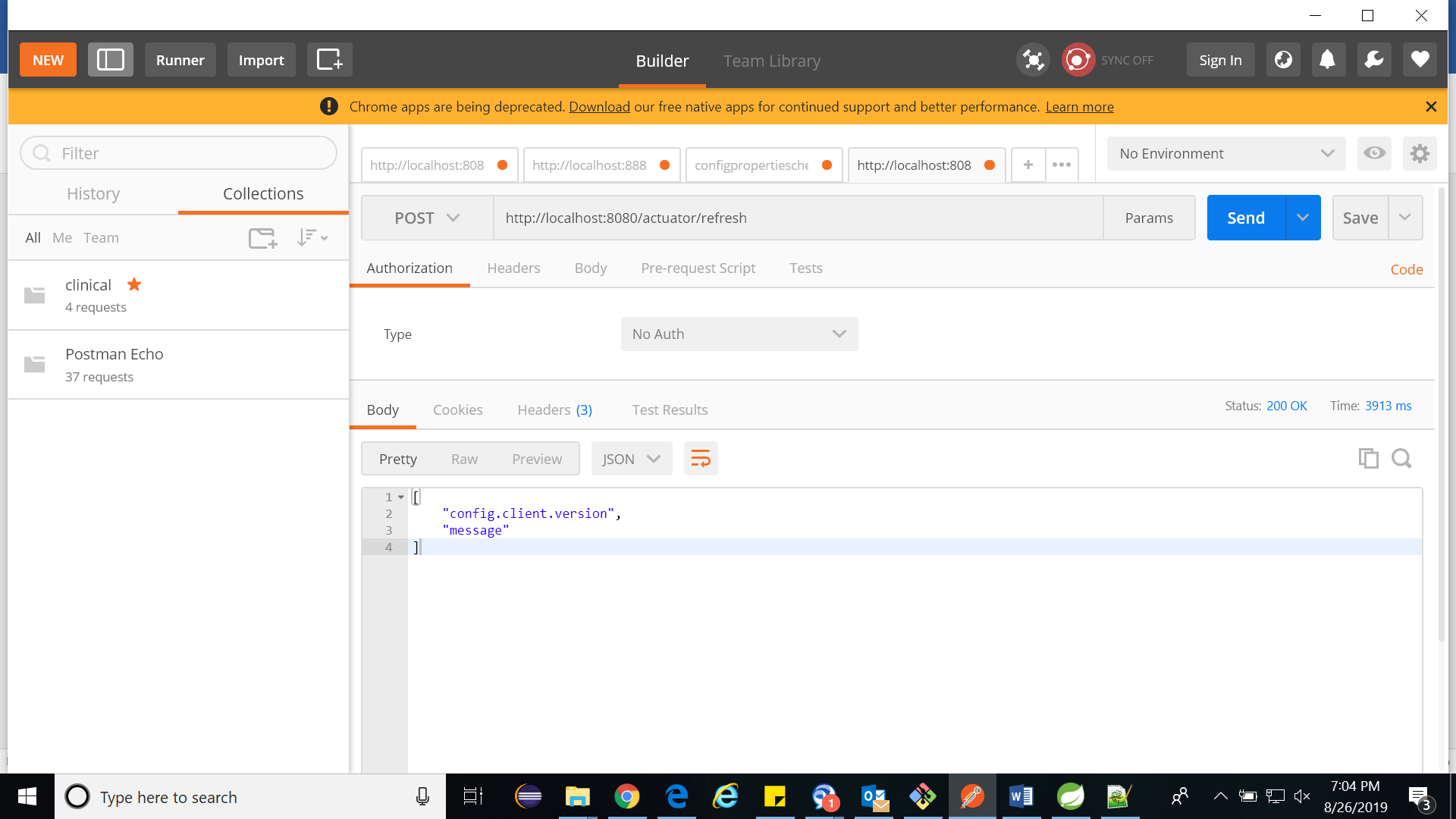
**Garg has been added to Variable Message.**



1. **Then hit the URL :-**

<http://localhost:8080/actuator/refresh>

**Port number :- Config client application port number**



1. **Then hit your config client application and you will able to see the modified value without redeploying your config client application. 😊**

<http://localhost:8080/rest/msg>

