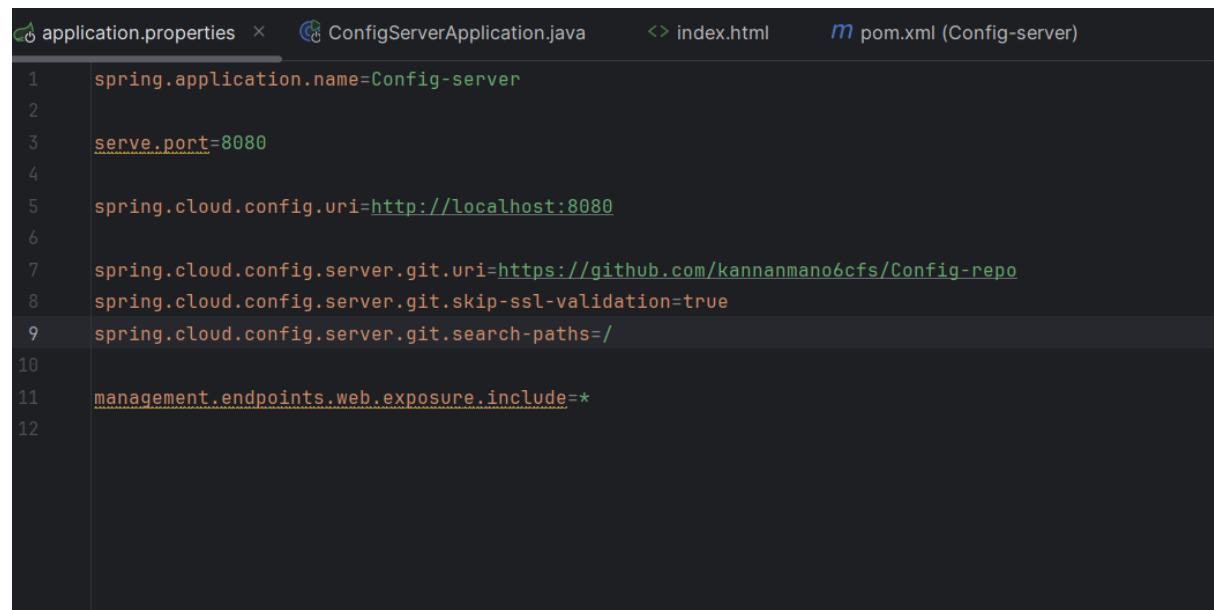


1. Create an application for Cloud Config Server and add config server dependency or go to the pom.xml file, add these dependency code

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-config-server</artifactId>
</dependency>
```

2. Go to the properties file of Config Server and these below configuration properties



```
1  spring.application.name=Config-server
2
3  server.port=8080
4
5  spring.cloud.config.uri=http://localhost:8080
6
7  spring.cloud.config.server.git.uri=https://github.com/kannanmano6cfs/Config-repo
8  spring.cloud.config.server.git.skip-ssl-validation=true
9  spring.cloud.config.server.git.search-paths=/
10
11  management.endpoints.web.exposure.include=*
12
```

3. Use ConfigServerApplication file and add the annotation @EnableConfigServer as below:



```
package com.example.configserver;

> import ...

@SpringBootApplication
@EnableConfigServer
public class ConfigServerApplication {

    > public static void main(String[] args) { SpringApplication.run(ConfigServerApplication.class, args); }

}
```

4. Create an another Spring boot application as ConfigClient with the config client dependency or add the below dependency in pom.xml file

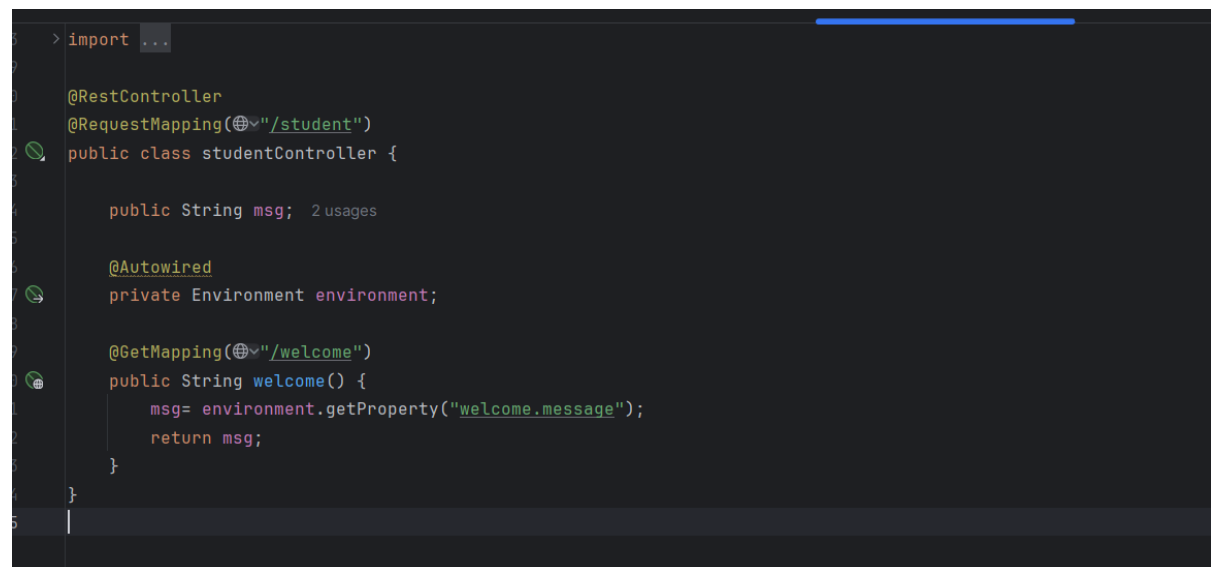
```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-config</artifactId>
</dependency>
```

5. Go to the properties and add the configuration for the client as below:

A screenshot of an IDE showing the 'application.properties' file. The file contains the following configuration: spring.application.name=configclient1, server.port=8081, spring.config.import=configserver:http://localhost:8080, and management.endpoints.web.exposure.include=*. The IDE tabs at the top show 'application.properties', 'pom.xml (configclient1)', 'Configclient1Application.java', and 'studentController.java'.

```
1 spring.application.name=configclient1
2
3 server.port=8081
4
5 spring.config.import=configserver:http://localhost:8080
6
7 management.endpoints.web.exposure.include=*
```

6. Create a controller for the demo purpose as below:

A screenshot of an IDE showing the 'studentController.java' file. The code includes imports for RestController, RequestMapping, and Autowired. It defines a StudentController class with a welcome() method that fetches a message from the environment. The IDE tabs at the top show 'studentController.java' and 'Configclient1Application.java'.

```
> import ...

@RestController
@RequestMapping("/student")
public class studentController {

    public String msg; 2 usages

    @Autowired
    private Environment environment;

    @GetMapping("/welcome")
    public String welcome() {
        msg= environment.getProperty("welcome.message");
        return msg;
    }
}
```

7. Add a properties in the github repository which mentioned in the properties file of cloud config server application

<https://github.com/kannanmano6cfs/SBAug2025/blob/main/cloudclient.properties>

8. Run the cloud config server and cloud client applications and explore the api resource, it will fetch the properties from github through cloud config server

<http://localhost:8081/student/welcome>