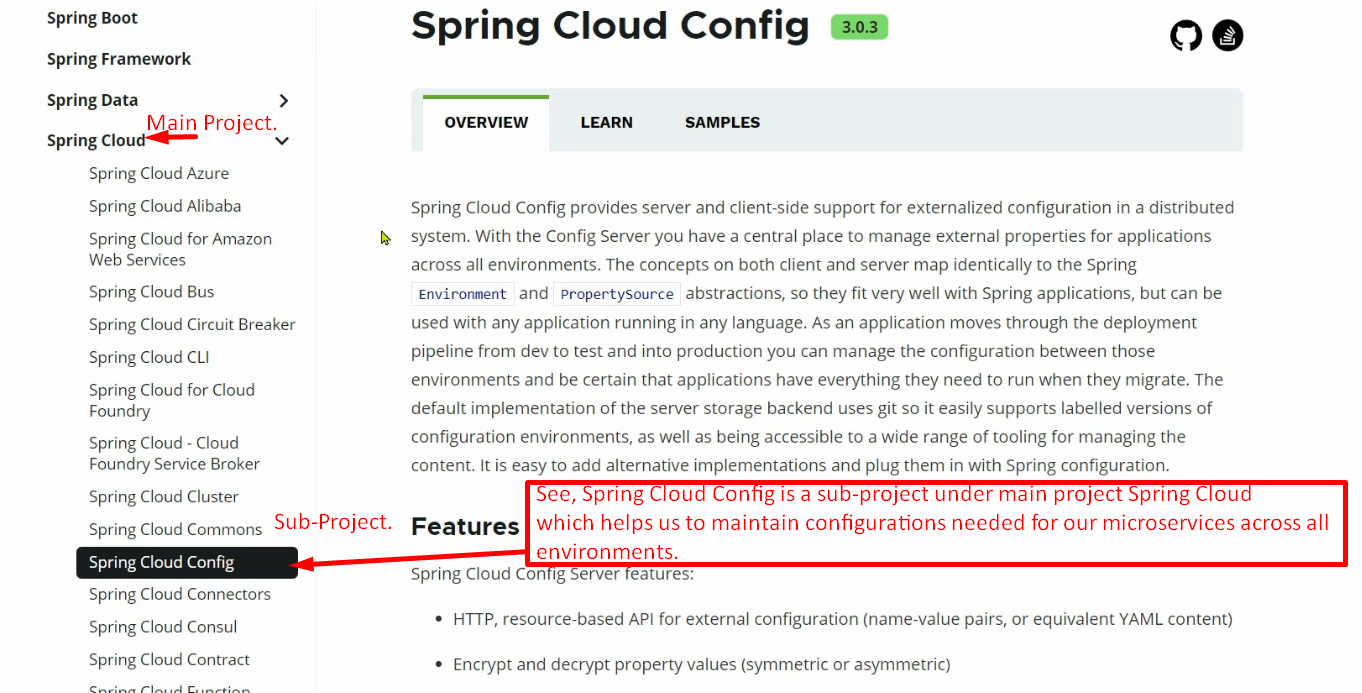
1. Spring Cloud provides many tools and projects to write a microservice quickly and effectively.
2. **Spring Cloud Config** is one of such **sub-projects** from Spring Cloud that helps us to maintain a configuration management service with the approach and architecture that we discussed in the previous lecture.
3. **What is Spring Cloud Config**?
   1. This is sub-project from Spring Cloud which provides server and client-side support for externalize configurations inside a distributed system.
   2. **What does “Server and Client Side Support” mean?**
   3. It means that you should have a server which maintains configuration by reading your all configurations from central repo such as Github and once this server is up and running, all the clients (microservices) should have some mechanism to connect to this Config server so that they can get the properties they need **based on the environment and microservice name** that they are running with the help of Docker Images.  
      **Microservice Name**: Config Server holds configurations for all Microservices.   
      So, we need to tell microservice name for which we’re requesting.  
      **Environment**: One microservice has different configurations for different environment, so need to tell for which environment.
   4. So, with the Config server, as I said, you will have a central place **to manage and expose** all your properties across all environments and all micro services that you have inside your app.
   5. We will create a docker container from the microservice code base (Github) **regardless of environment** on which we’re going to run the container.   
      Then depending on the environment in which we are going to run the container, our microservice will pull the environment specific configuration from the Config server.   
       So, now Docker container will behave as DEV container, UAT Container or PROD Container based on the configurations, the container fetches from the Config Management Service.
4. 
5. Spring Cloud Config Sub-Project two kinds of features:
   1. **Spring Cloud Config Server Features**:
      1. Annotation @EnableConfigServer to convert a microservice into Spring Cloud Config Server.
      2. Reading configuration files with extensions properties and YAML.
      3. Exposing HTTP RESTful endpoints for Config Clients to read respective configurations.
      4. Encryptions and Decryptions.
   2. **Spring Cloud Config Client Features:**
      1. Ability to bind your microservice to Config server to read configurations.
      2. Encryptions and Decryptions.
6. 