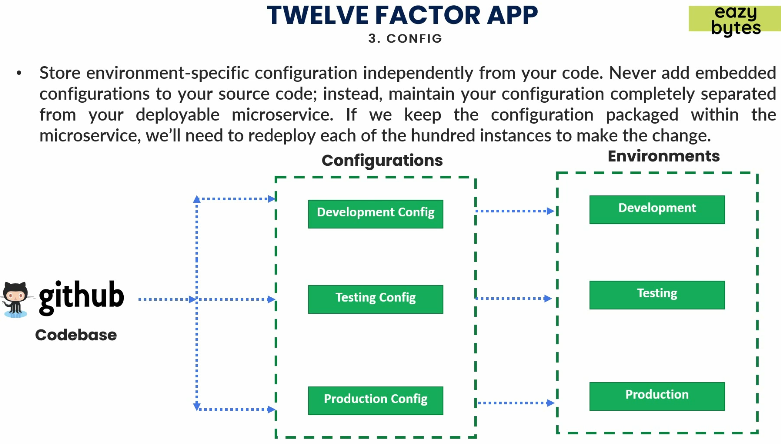
1. **Agenda**:
   1. Let’s focus on how to handle the challenge we’re going to face when building a microservice that is how to do configuration/property management needed for hundreds of your micro-services.
2. From Factor12 App (Cloud Native App), we know we have to maintain its configuration management like  
   we should store all our environment specific configurations independently from your code, which means never ever embed any of those configurations in your source code.
3.   
   If you put configurations outside of your github code repository and in a central repository, if you follow this approach, whenever you build a docker image from the code base, using the same docker image you can deploy that in any environment. The only thing you have to handle is that you have to feed the configuration based on your environment from your external system into your docker image/container during the start up of your microservice.
4. Following are the challenges we’re going to face when externalize the configurations.
   1. **Separation of Configurations/Properties**:
      1. First challenge is how to separate all your configurations and properties needed by all microservices.
      2. Configurations/Properties like DB credentials, SMPT Folder Location.
      3. These properties would change from environment to environment.
      4. If you hardcode in your microservice as per environment, you need to regenerate another docker image as per new environment.
      5. So, put all configurations/properties in a central repository.
   2. **How to inject configurations/properties**:
      1. How to inject those configurations into microservices when they are staring up based on environment.
   3. **Maintain Configurations/Properties**:
      1. How to maintain all these configurations/properties in central repository.
      2. How to version them.
      3. When any property is changed, how to inform all the running microservices about those changes instead of restarting microservices.
5. 