(..)

Requirements

- A cloned version of the code repository (https://github.com/platform-acceleration-lab/apps-cloud-native-evolution-code)

What you will learn/review

- How to analyze the dependency graph for a well structured monolith.
- How to push a well structured monolith to PCF

Deploy a well structured monolith to PCF

1. Create a branch from the `v1` tag of the code repository (https://github.com/platform-acceleration-lab/apps-cloud-native-evolution-code).

   ```
   git checkout –b my–solution v1
   ```

2. Open the project in IntelliJ.

3. Familiarize yourself with the dependency graph of `applications/ums` by analyzing the `build.gradle` files. You should be able to draw the dependency graph of the system starting with `applications/ums`. Your graph will include boxes for `applications/ums`, `components/billing`, `components/email`, `components/payments`, `components/subscriptions`.

4. Build and deploy the UMS application (applications/ums/build/libs/ums-0.0.1-SNAPSHOT.jar) to PCF using a random route.

```
./gradlew build
cf push ums --random-route -p applications/ums/build/libs/ums-0.0.
1-SNAPSHOT.jar
```

5. Use `curl` to confirm that the application is running successfully. Test both the creation of a subscription as well as the subscription list endpoints with the following commands.

```
curl -i -H "Content-Type:application/json" -d '{"userId":"user-123
", "packageId":"package-123"}' [your route]/subscriptions
curl -i [your route]/subscriptions
```

Your responses will look similar to this:

```
HTTP/1.1 201 Created
Date: Wed, 07 Dec 2016 00:50:26 GMT
Server: Apache-Coyote/1.1
X-Vcap-Request-Id: 6ac681f0-3b02-49c2-5a49-e1f033828e2e
Content-Length: 0
Connection: keep-alive
```

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Date: Wed, 07 Dec 2016 00:52:56 GMT
Server: Apache-Coyote/1.1
X-Vcap-Request-Id: f76e9a45-e9f2-424a-7081-affa1b515ee2
Content-Length: 97
Connection: keep-alive

[{"userId":"user-123","packageId":"package-123"}]
```

Assignment

Once you are done with this section and the application is deployed and working on PCF, you can submit the assignment using the `submitWellStructuredMonolith` gradle task.

It requires the `umsUrl` project property, for example:

```
cd ~/workspace/assignment-submission
./gradlew submitWellStructuredMonolith -PumsUrl=http://my-ums.cfapps.io
```

(https://pivotal.io)