**FURTHER READING/TODO**

**1. Read up on 12 Factor Apps**

**2. Read up on the API Gateway Pattern**

**3. Listen to SE podcasts**

**4. Take some Udemy classes on Springboot and Angular**

**5. Read up on the h2 DB**

**6. Read up on Test Driven Development**

**7. Do the “Tour of Heroes” tutorial by Angular**

**8. Learn how to test Angular components with Jasmine+Karma**

**9. Know how to merge to master branch and branch out for new features on git**

**FOR PRACTICE:**

Make a Springboot App like the movie app with an Angular frontend but from scratch and then deploy it OR at least set it up to be deployed.

Then make it more complex by adding another microservice.

Check to see if there is a microservice springboot tutorial.

**Define the following:**

**1. What is DevOps?**

A combination of software development (Dev) and operations (Ops). It is defined as a ***software engineering methodology*** which aims to integrate the work of software development and software operations teams by facilitating a culture of collaboration and shared responsibility.

“Plan -> Code -> Integrate -> Test -> Release -> Deploy -> Operate"

**2. What is Continuous Development Continuous Integration (CDCI)?** \*Note: One of many ***pipelines***

Continuous Integration (CI) is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early. Continuous Deployment (CD) is closely related to Continuous Integration and refers to the release into production of software that passes the automated tests.

The practices

• Maintain a single source repository

• Automate the build

• Make your build self-testing

• Every commit should build on an integration machine

• Keep the build fast

• Test in a clone of the production environment

• Make it easy for anyone to get the latest executable version

• Everyone can see what’s happening 

• Automate deployment

How to do it

• Developers check out code into their private workspaces

• When done, commit the changes to the repository

• The CI server monitors the repository and checks out changes when they occur

• The CI server builds the system and runs unit and integration tests

• The CI server releases deployable artefacts for testing

• The CI server assigns a build label to the version of the code it just built

• The CI server informs the team of the successful build

• If the build or tests fail, the CI server alerts the team

• The team fixes the issue at the earliest opportunity

• Continue to continually integrate and test throughout the project

Team responsibilities

• Check in frequently

• Don’t check in broken code

• Don’t check in untested code

• Don’t check in when the build is broken

• Don’t go home after checking in until the system builds

Software: Jenkins, Bamboo, Travis CI, TeamCity

The continuous delivery pipeline is an automated set of processes that use tools to compile, test, and deploy code for new software features. The purpose of a continuous delivery (CD) pipeline is to create a continuous management and release setting where bugs, compatibility issues, and security breaches are identified and fixed as early as possible. Also called the “deployment pipeline,” the CD pipeline is the vehicle that drives iterative software application development. It breaks down the strategy of continuous integration and continuous delivery into stages so development teams can gather results incrementally, at each stage of the build. Developers are then able to run tests and make other assessments without having to wait until the entire workflow is completed.

Stages:

“Commit Stage -> Automated Acceptance Testing-> Continuous Deployment -> Production Release”

**3. What is Jenkins?**

**4. What are Pivotal Web Services?**

**5. What is a boot jar?**

**6. What is boot run?**

**7. What are Selenium tests?**

**8. What is Zipkin?**

**9. What is Spring Sleuth?**

**10. What is SonarQube?**

**11. Look up Typescript**

**12. What is a unit test, integration test, smoke test, end to end test, component tests, api tests, gui tests, etc?**

**13. What is Docker?**

**14. config server**

**15. What is blue-green / red-black deployment?**

**16. Exceptions in Java**

**17. Call by value**

**18. Call by reference**

**19. Know when Java uses call by value or reference**

**20. Foreign key**

**21. Inner Join**

**22. Projection in Springboot**

**23. What is Chef or Puppet?**

**24. What are Microservices?**

**25. What is a pipeline in software development?**

**26. What is Gradle?**

**27. What are Gradle tasks?**

**28. What is Gradle dofirst and dolast?**

**29. What are Grade test sets?**