1. Talk about what our understanding of the problem is:
    a. We want a strategy to reduce the development cycle across the board down to 2 weeks from 2 months
    b. We want a working prototype to show how a devops pipeline could feasibly reduce this cycle
2. Talk about our strategy to reduce the lifecycle down to 2 weeks
    a. Break work into smaller chunks allowing for faster iterations
        i. More focused work means less context switching
        ii. Use iterative development tools like feature flags to allow for smaller incremental changes as opposed to larger merge requests
    b. Automate work as much as possible
        i. Reduce overhead for building, testing and deployment via an end to end devops pipeline
        ii. Means we can get feedback from changes much faster than having to rely on testers etc down the human pipeline
    c. An important part of this process is that it should also be introduced incrementally, instead of expecting instant adoption of the entire new process
        i. Should be introduced incrementally over a series of the new sprints
        ii. Should respect the current processes they have in place, keeping what works well and replacing with more efficient processes where necessary
3. Talk about our solution
    a. What have I done
        i. Created a pipeline on AWS using Jenkins, the blueocean plugin for a more convenient UI, maven for building the Java code, JUnit for testing docker for containerising the jenkins image, building testing and deployment of the JAR
        ii. Declarative pipeline with 4 stages to it
        iii. SCM polling
            1. Checks for changes in the git repository and if so finds and runs the Jenkinsfile
        iv. Build
            1. After finding changes, jenkins has docker instantiate a maven configured container ready for use with the other stages
            2. Builds without testing
            3. During build, this will download dependencies if they aren’t already present in the parent Jenkins container or use the existing ones
        v. Test
            1. Runs the automated tests
            2. Produces a report which is given in the UI
        vi. Deploy
            1. Produces an executable jar stored within the jenkins container within the maven repository
        vii. Every stage gives feedback on successes / failures through the jenkins UI

b. Benefits of using such a system? Why will this decrease development cycles?
    i. As soon as code is ready, it should be run through automatic testing as long as developers push code
    ii. No overhead necessary to communicate changes, can simply run on preconfigured testing / deployment environments