BDD(Behavior Driven Development)

* Design the application from the perspective of the end users
  + Benefits to BDD
  + Design the UI
  + Prevents you for over or under developing the application
    - Very common problem in unguided development is writing features that no one uses/needs
    - Or is done already in another feature
  + Solidifies your requirements
    - Can help guide your backend design
      * For example you might realize what information needs to be persisted/how your entities should look

Steps to BDD

1. Generate your user stories
2. Creating acceptance criteria/ designing the UI of the application
3. Write the Cucumber feature files
4. Design/development

* TDD
  + You write the tests and then create the code to pass those tests
* BDD
  + You write the acceptance criteria and create the application the meet that acceptance criteria’

Cucumber

* BDD framework
  + Written in Gherkin syntax
* Keywords
  + Given – The state of the application
    - The Unregistered Guest is on the home page
      * Is\*
  + When – A user action
    - The Manager clicks on show employees button
      * Clicks\*, mouseover\*, types in\*
  + Then – Some sort of verification (something the user can see)
    - Reimbursement successfully submitted modal should pop up
      * Should\*
  + Scenario – Basically a user story
  + Scenario Outline – Parameterized user story
  + Feature – collection of scenarios (usually related to an epic)
  + Background – a series of steps to be run before each scenario
* Setting up Cucumber in Java
  + Download the appropriate WebDriver for automating a browser
  + Programmatically set up your Java application to use that driver
  + Write the feature files
  + Write the step implementations
    - The actual methods that get called during each step in a scenario
    - Make the scenario steps a concrete action
  + Write a runner file that connects the feature files to the step implementations
    - Should have a setup and tear down
    - @CucumberOptions(features = "src/test/resources", glue = "dev.ranieri.steps")

Selenium

* Selenium can be used to automate web browsers
* WebDriver is the core interface of Selenium
  + Allows you to get elements and interact with the page
* Waiting for elements
  + ImplicitWait – wait a couple second for any element to appear
  + ExmplicitWait – Can specify a certain for a time for a specific element to appear
  + Thread.Sleep – halts your code running for a while
    - Generally avoid if possible
* POM
  + Creating a POM
    - Pass in a Webdriver into the constructor
    - Use the PageFactory method to populate the web elements in the page
  + Ways to find elements
    - @FindBy
      * Id
      * Name
      * className
      * xpath
        + The directions to the element
      * css