SOFTWARE
ARCHITECTURE

Escuela de Ingeniería Informática

Universidad de Oviedo

Lab 8

BDD and Acceptance tests

2025-26

Pablo González
Jose Emilio Labra Gayo
Diego Martín
Celia Melendi

Acceptance tests and BDD

- Tests that can be run in front of the client
  - If the tests pass, the product is accepted
- Behaviour-Driven Development (BDD)
  - Variant of TDD
    - Acceptance test driven development
  - Behaviour = User Stories
  - Also known as: *Specification by example*
  - Goal: Executable specifications
- Some tools:
  - cucumber, jBehave, concordion

BDD - User Stories

- Simple
- Readable by domain experts (business people)
- Approved by domain experts
- Other advisable characteristics:
  - Independent (with no strong relationships)
  - Negotiable (with no specific details)
  - Valuable for the customer
  - Estimable (to add them to Sprints)
  - Small (or consider division)
  - Testable (automatic tests)

User story structure

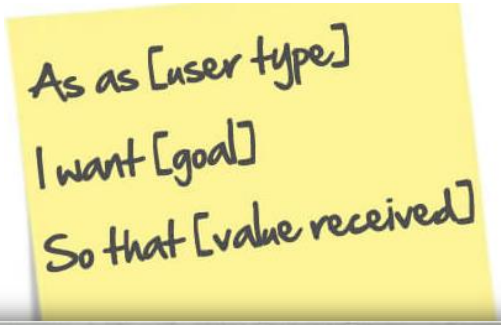
Feature: *Title (one line describing the story)*

The following structure is recommended:

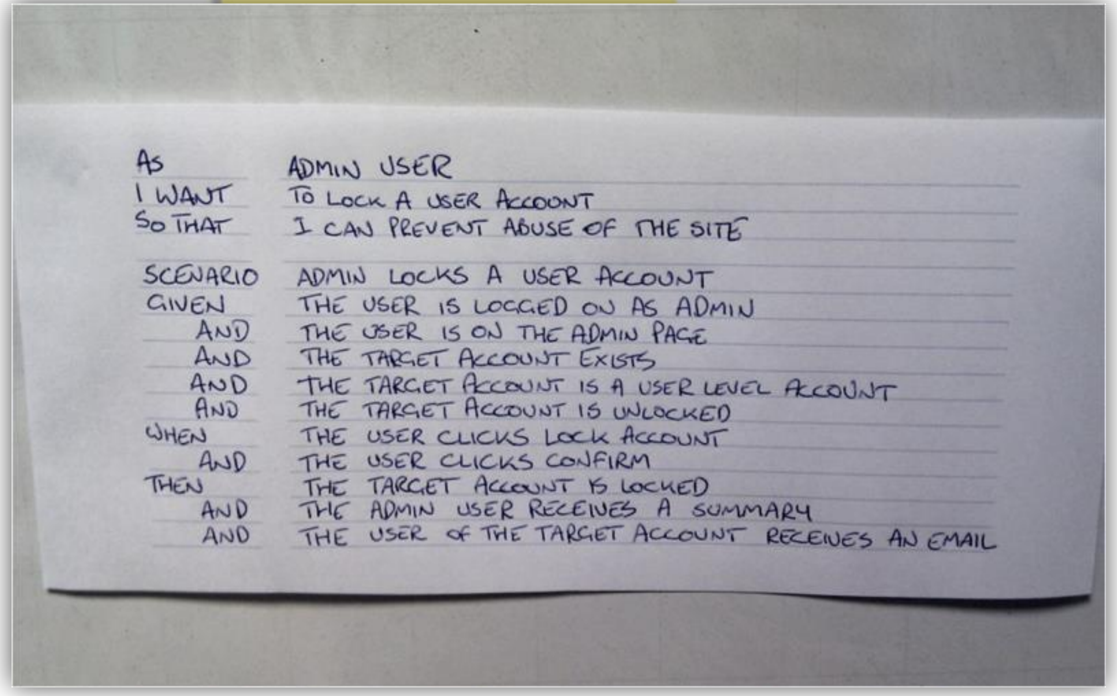
As a [role]
I want [feature]
So that [benefit]

Scenarios

Given [Context]
And [Some more context]
when [Event]
then [Outcome]
And [Another outcome]

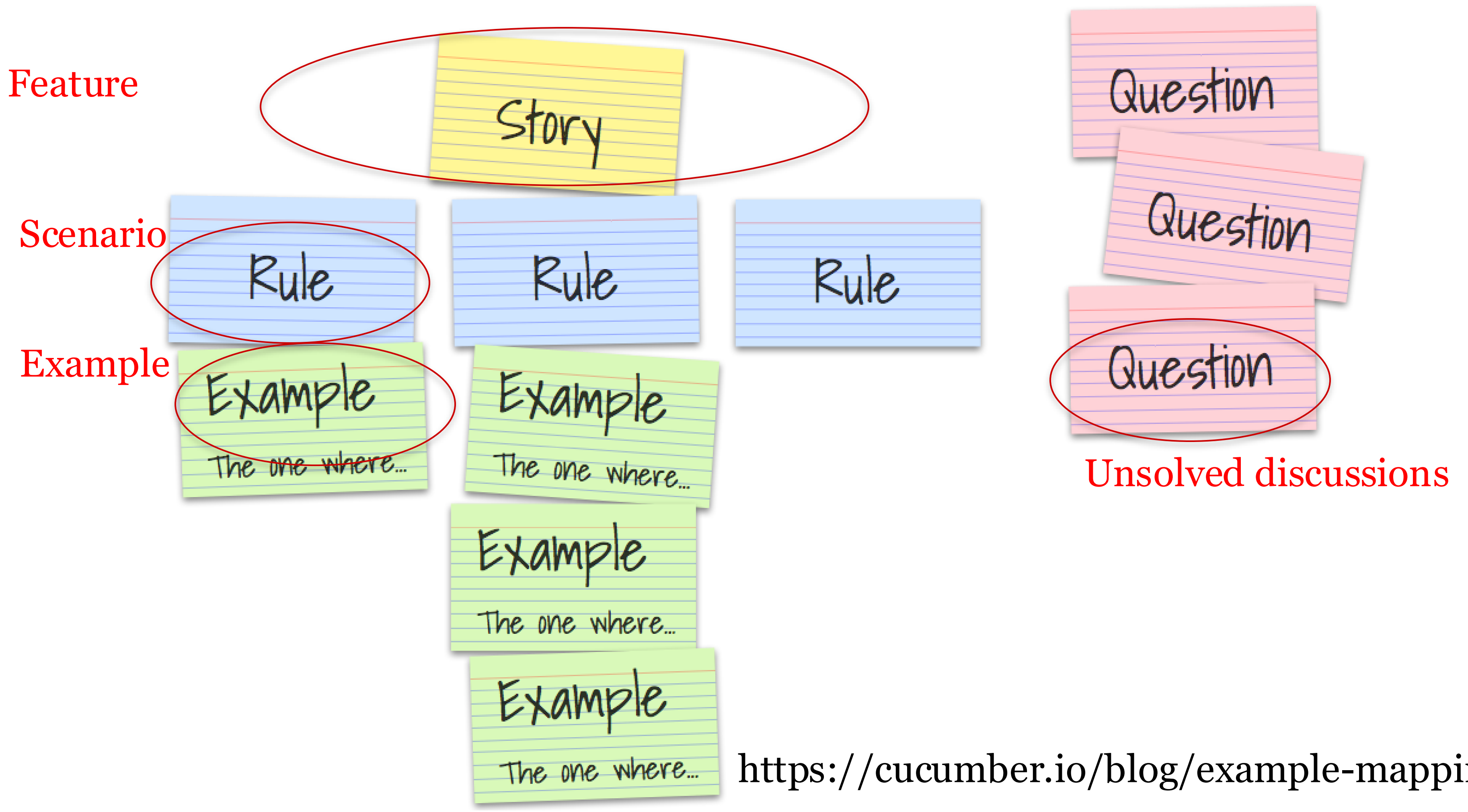


As as [user type]
I want [goal]
So that [value received]



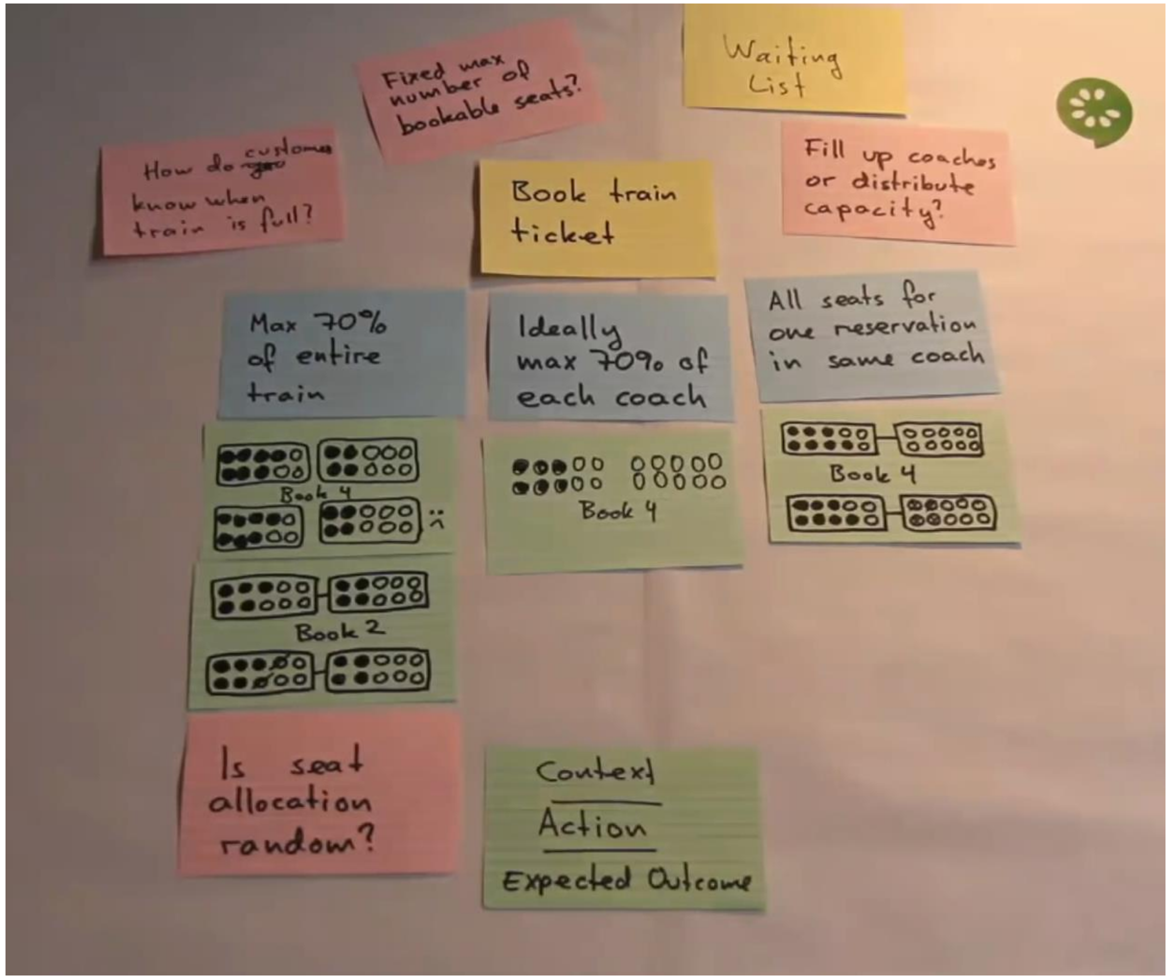
AS	ADMIN USER
I WANT	TO LOCK A USER ACCOUNT
SO THAT	I CAN PREVENT ABUSE OF THE SITE
SCENARIO	ADMIN LOCKS A USER ACCOUNT
GIVEN	THE USER IS LOGGED ON AS ADMIN
AND	THE USER IS ON THE ADMIN PAGE
AND	THE TARGET ACCOUNT EXISTS
AND	THE TARGET ACCOUNT IS A USER LEVEL ACCOUNT
AND	THE TARGET ACCOUNT IS UNLOCKED
WHEN	THE USER CLICKS LOCK ACCOUNT
AND	THE USER CLICKS CONFIRM
THEN	THE TARGET ACCOUNT IS LOCKED
AND	THE ADMIN USER RECEIVES A SUMMARY
AND	THE USER OF THE TARGET ACCOUNT RECEIVES AN EMAIL

BDD – Example Mapping



https://cucumber.io/blog/example-mapping-introduction/

BDD – Example Mapping


Fixed max number of bookable seats?
Waiting List
How do customer know when train is full?
Book train ticket
Fill up coaches or distribute capacity?
Max 70% of entire train
Ideally max 70% of each coach
All seats for one reservation in same coach
Book 4
Book 4
Book 4
Book 2
Is seat allocation random?
Context
Action
Expected Outcome

Introducing example mapping [video]

BDD using Cucumber

- Cucumber = developed in Ruby (2008)
  - RSpec (Ruby), jbehave (Java)
  - Based on Gherkin
    - internal language to define user stories
  - Web: `http://cukes.info`
  - Support for multiple languages
- Java: cucumber-jvm
  - `https://github.com/cucumber/cucumber-jvm`

BDD using cucumber

- Features define some functionality
  - Gherkin language
    https://cucumber.io/docs/gherkin/
    Can be used in several languages
- User stories are linked to step definitions
  - Step definitions can be run to validate user stories

BDD using cucumber

- Feature: Describes a system feature
  - A feature can have several scenarios
  - Scenario:
    - How must the system behave in some context
      - *Given*: Prepares scenario
      - *When*: Interact with the system
      - *Then*: Checks the state
      - *Examples*: Specific data

BDD

- Step by step guide to a user story
  - Install Cucumber
  - Write a first scenario in Gherkin
  - Write steps definitions in a chosen programming language
  - Run cucumber

BDD with cucumber

- Depends on programming language/environment
  - Java/Javascript/Python/…
  - Installation: https://cucumber.io/
- React: https://github.com/Arquisoft/yovi_0
  - Cucumber: Module to define user stories in Gherkin

    ```
    $ npm install --save-dev cucumber
    ```
  - Playwright. Module to run the tests in a browser
    - There are other alternatives as Puppeteer.

    ```
    $ npm install --save-dev playwright
    ```
  - Start-server-and-test: Module for launching the infraestructure

    ```
    $ npm install --save-dev start-server-and-test
    ```

BDD

- User Story example using Node.js

```
Feature: Register. Validate the register form
```

Feature

```
Scenario: The user is not registered in the site
  Given the register page is open
  When I enter "Alice" as the user name and submit
   I should see a welcome m essage contining "Hello Alice
```

Scenario

webapp/test/e2e/features/register.feature

BDD

Webapp/src/test/e2e/steps/register.steps.js

```
import { Given, When, Then } from '@cucumber/cucumber'
import assert from 'assert'

Given('the register page is open', async function () {
  const page = this.page
  if (!page) throw new Error('Page not initialized')
  await page.goto('http://localhost:5173')
})

When('I enter {string} as the username and submit', async function (username) {
  const page = this.page
  if (!page) throw new Error('Page not initialized')
  await page.fill('#username', username)
  await page.click('.submit-button')
})

Then('I should see a welcome message containing {string}', async function (expected) {
  const page = this.page
  if (!page) throw new Error('Page not initialized')
  await page.waitForSelector('.success-message', { timeout: 5000 })
  const text = await page.textContent('.success-message')
  assert.ok(text && text.includes(expected), `Expected success message to include "${expected}", got: "${text}"`)
})
```

BDD [Browser Configuration]

- webapp/test/e2e/support.mjs
  - Configures how to launch the browser to perform the tests
  - Can be also configured with other browsers.
  - We use **headless=true** (by default) to run the tests in the CI system but we can change it to false to run them locally.
  - The **slowMo** parameter is useful to slowdown the tests and see what is happening

```
Before(async function () {
  // Allow turning off headless mode and enabling slow motion/devtools via env vars
  const headless = true
  const slowMo = 0
  const devtools = false

  this.browser = await chromium.launch({ headless, slowMo, devtools })
  this.page = await this.browser.newPage()
})
```

BDD [Configuration – Launching the system]

`webapp/package.json`

- Configures how to launch the system
  - For testing this app we need the backend and the webapp
- We use the `start-server-and-test` library
  - This library accepts pairs of parameters (run command, url to test)
- This four scripts are in charge of running the E2E tests
  - Excuting **npm run test:e2e** will launch the infrastructure and run the tests

```
"test:e2e:run": "cucumber-js --import test/e2e/support/setup.mjs --import test/e2e/steps/**/*.mjs test/e2e/features --format progress",
"test:e2e": "start-server-and-test start:all http://localhost:5173 test:e2e:run",
"test:e2e:install-browsers": "npx playwright install",
"start:all": "concurrently \"npm run dev\" \"npm --prefix ../users start\""
```

Other example cucumber + selenium + java Spring boot from previous years:

https://github.com/arquisoft/votingSystem0

Browser-based tests

- Browser automation
  - https://cucumber.io/docs/reference/browser-automation
- Several systems
  - Selenium WebDriver - http://docs.seleniumhq.org/
  - Capybara - http://teamcapybara.github.io/capybara/
  - Watir - https://watir.com/
  - Serenity - http://serenity-bdd.info

Selenium

- Selenium IDE: Allows to record actions
  - Firefox and Chrome plugins
- Generates code to execute those actions

Bibliography and links

- User Story Mapping by Jeff Patton
  - **User Story Mapping: Discover the Whole Story, Build the Right Product, 1st Edition**
    https://www.amazon.com/User-Story-Mapping-Discover-Product/dp/1491904909
- User stories
  - Scrum. Historias de Usuario (Fernando Llopis, Universidad de Alicante)
    https://fernandollopis.dlsi.ua.es/?p=39
  - User stories with Gherkin and Cucumber (Michael Williams)
    https://medium.com/@mvwi/story-writing-with-gherkin-and-cucumber-1878124c284c
  - Cucumber 10 minutes tutorial (JS)
    https://docs.cucumber.io/guides/10-minute-tutorial/
- Browser based tests
  - Automated UI Testing with Selenium and JavaScript
    https://itnext.io/automated-ui-testing-with-selenium-and-javascript-90bbe7ca13a3