A yellow sign with white text

Description automatically generatedA yellow sign with black text

Description automatically generated

Junit has the above limitation, in junit - feature files run in parallel and not scenarios.

Second problem is cross browser testing. We may have to maintain different packages for testing same feature on different browsers.

# Add the compiler plugin to achieve maven lifecycle commands-

A screen shot of a computer code

Description automatically generated

A close-up of a computer code

Description automatically generated

# Third is maven fail safe plugin-

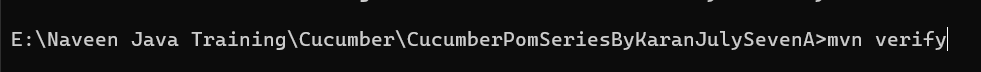
We can write any value for the goal field as per our convenience.

Jumbo plugin which needs to be added-

A blue screen with white text

Description automatically generated

Open cmd and go to the project location. use the below code and hit enter.



Build success-

A screenshot of a computer

Description automatically generated

Open report-

A screenshot of a computer

Description automatically generated

For failed case we get screenshot-

A screenshot of a website

Description automatically generated

Pdf report-

A screenshot of a graph

Description automatically generated

Index file also fine-

A screenshot of a computer

Description automatically generated

# Lets see the run with thread –

A screenshot of a computer

Description automatically generated

Run using mvn verify-

Don’t use mvn clean install.

A screenshot of a computer

Description automatically generated

Refresh project.

We get new folder here.

A screenshot of a computer

Description automatically generated

Open index.html.

We get such type of output.

Mine is not coming properly, for Naveen it works.

A screenshot of a computer

Description automatically generated

See naveens output-

A screenshot of a computer

Description automatically generated

# Lets increase one more feature file and see if thread count increases-

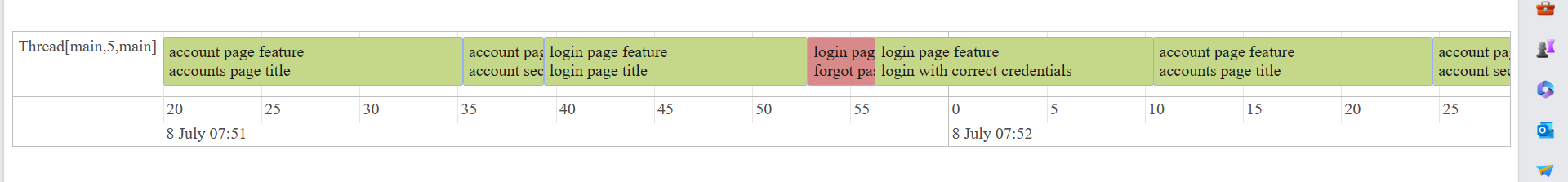
A computer screen shot of a yellow banana

Description automatically generated

Output-

A screenshot of a computer

Description automatically generated



Cucumber report-

A screenshot of a computer

Description automatically generated

# Project structure-

A screenshot of a computer

Description automatically generated

# Codes from this chapter-

|  |
| --- |
| Runner file-  package com.myTestRunner;  import org.junit.runner.RunWith;  import io.cucumber.junit.Cucumber;  import io.cucumber.junit.CucumberOptions;  @RunWith(Cucumber.class)  @CucumberOptions(  features = { "src/test/resources/appFeatures/" },  glue = { "stepDefinitions", "appHooks" },  //this adapter needed for the new extent reports.  //we can generate timeline html reports for threads using timeline plugin. we can know how many threads participated.  //we can see what threads present in a pool, which threads executed which scenario.  //we can give name for timeline .. this is the folder which will be created once test runs.  plugin = { "pretty",  "com.aventstack.extentreports.cucumber.adapter.ExtentCucumberAdapter:",  "timeline:test-output-thread/"}    )  public class MyTestRunner {  } |