### 6.3 **BDD - Behavioral Driven Development.**

TDD - Test Driven Development. - In TDD, Developers will write dev code for features. -

Testers - Exhaustive, all the scenarios, In Terms of Business.

\

BDD is evolved process from TDD.

### **BDD - Behavior Driven Development. -**

BDD is a approach to Develop , test the software which aims to bridge the gap between business people and technical.

Business people - Business Analyst,

Technical - Devs, Tester, Devops

There is constant communication, feedback loops bw the 3 amigos.

And there are Rapid small iterations of software feature developments using the collaboration of these 3 amigos.

The focus is more on the Behavioural aspect of the product.

We are talking about the 3 Amigos - Developers, Testers, BA/ Product owners.

These 3 amigos always work together - There are always having constant communication, constant feedback is given to each other

For every Sprint/feature/ user story -> We are having the perspective of Testers(User’s point of view ) , Business perspective(BA) , Feasibility analysis (DEveloper’s perspective.)

Advantages :

1. Robust Product - 3 amigos perspective
2. Constant communication.
3. Constant Feedback is given.
4. Speed of Development is increased.
5. The product releases faster.
6. **Testing also involves business analysts/ Product owners.**

### Cucumber is a framework , BDD is not a framework. BDD is a approach

### 

### 6.3b BDD features -

* This approach involves the usage of shared language that enhances communication between various tech and non-tech teams.
* Tests are more user-focused and based on the system’s behaviour.
* In BDD, *“Given-When-Then”* is the proposed approach for writing test cases.
* Focuses on defining ‘behaviour’ rather than defining ‘tests’.
* Enhances communication among the members of cross-functional product teams.
* Helps reach a wider audience via the usage of non-technical language.

### 6.3c **What is Cucumber?**

Cucumber is a **tool that supports behaviour-driven development (BDD)**. It runs automated **acceptance tests** written in BDD format.

UAT - USer acceptance testing - This is done by Customers, Business people, End user. They want to make sure that, product is ready from the End user POV.

It explains test steps and application behaviour using the Cypress Gherkin language in simple English.

### Why use Cucumber for testing?

Cucumber is important as a testing tool for the following reasons:

* Cucumber is open-source and free to use.
* Using Cucumber, QAs can write your test scripts in multiple languages such as Java, Ruby, .NET, Python, etc.
* It integrates with Selenium, Ruby on Rails, Cypress, Watir, and other web-based testing tools.
* Cucumber is one of the most widely used BDD tools.

Cucumber Framework Components :

1. **Feature files** - Tests using Plain English format - Gherkin syntax (**Given, When, And, Then**) – Business people will write tests for User Acceptance(UAT).
2. **Step Definitions** - Cypress code - which translates your feature files into cypress code.
3. Runner file. - this is used to run the cucumber project. (Not required for Cypress)

**Feature files - which are having .feature extension.**

**Given - Preconditions, any inputs.**

**WHen - All the user actions**

**Then - Expected condition (Conclusion)**

**Feature** : Redbus Bus booking.

**Scenario** : To validate booking a bus from Bangalore to Mumbai

**Given** Open the browser and Enter redbus URL.

**When** User enters From city as Bangalore

**And** USer enters To city as Mumbai

**And** user enters current date , clicks on Search button

**Then** List of Buses should be displayed.

**Feature : simple form**

**Scenario : To validate all the fields in simple form.**

**Given User has opened the browser and entered URL**

**When User enters firstname,lastname,email, contact, message**

**And user clicks on submit button.**

**Then Alert with a success message is displayed.**

**Feature : webtables**

**Scenario : To validate a text from webtable.**

**Given User has opened the browser and entered URL**

**When User views the Sortable table**

**Then ‘Accepted’ should be visible in 2nd row , 2nd coloumn.**

Installation and configuration :

**Step 1:** Install Cucumber for Cypress

Run the following command to install the Cucumber for Cypress package:

npm install —-save-dev cypress-cucumber-preprocessor

**Step 2 :** Add the configuration Cypress environment files as follows in config.js :

const cucumber = require('cypress-cucumber-preprocessor').default

on('file:preprocessor', cucumber())

const { defineConfig } = require("cypress");

const cucumber = require('cypress-cucumber-preprocessor').default;

async function setupNodeEvents(on,config){

on('file:preprocessor', cucumber());

return config;

}

module.exports = defineConfig({

e2e: {

setupNodeEvents,

specPattern: 'cypress/UAT/features/\*.{js,feature}'

},

});

**Step 3 : add dependencies in package. Json as follows :**

**"cypress-cucumber-preprocessor": {**

**"nonGlobalStepDefinitions": false,**

**"stepDefinitions": "cypress/support/step\_definitions"**

**}**

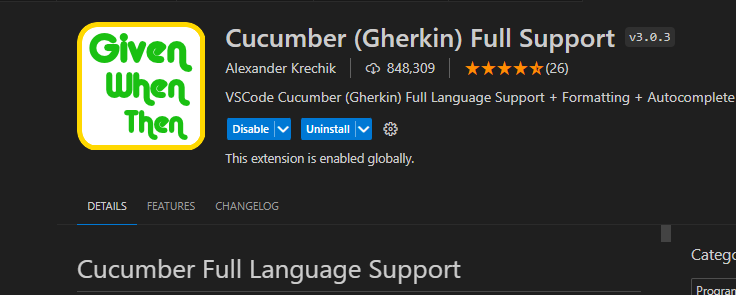
**"cypress-cucumber-preprocessor": {**

**"nonGlobalStepDefinitions": false,**

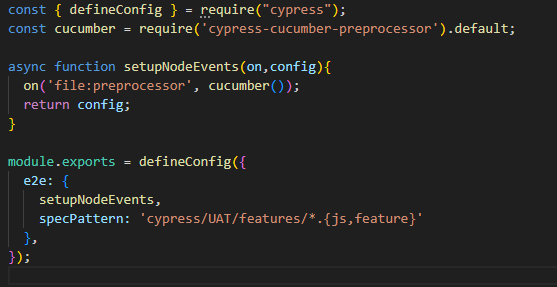
**"stepDefinitions": "cypress/support/step\_definitions"**

**}**

**Step 4 : Add cucumber extension in your VS code.**

****

**Step 5 : Change the spec pattern to detect the feature files accordingly.**

****

**Feature file :**

**Feature: Simple form**

**Scenario: to validate simple form**

**Given user opens the Simple form url**

**When user enters firstname ,lastname, email, message**

**And clicks on submit**

**Then An alert saying successful message has to be displayed.**

**Step definition :**

**import { Before,Given,When,Then,And } from "cypress-cucumber-preprocessor/steps"**

**Given('user opens the Simple form url',()=>{**

**// cypress code**

**cy.visit('https://v1.training-support.net/selenium/simple-form')**

**})**

**When('user enters firstname ,lastname, email, message',()=>{**

**cy.get('#firstName').type('Akhsay')**

**cy.get('#lastName').type('Koulgi')**

**cy.get('#email').type('ak@gmail.com')**

**cy.get('#number').type('9090909090')**

**})**

**And('clicks on submit',()=>{**

**cy.get("[value='submit']").click()**

**})**

**Then('An alert saying successful message has to be displayed.',()=>{**

**cy.on("window:alert",(msg)=>{**

**expect(msg).to.eq('Thank You for reaching out to us, Akshay Koulgi')**

**return true; // this will make the alert click on ok**

**})**

**})**