***Precondition for installing webdriver io***

? Where is your automation backend located? On my local machine

? Which framework do you want to use? mocha

? Do you want to use a compiler? No!

? Where are your test specs located? ./test/specs/\*\*/\*.js

? Do you want WebdriverIO to autogenerate some test files? Yes

? Do you want to use page objects (https://martinfowler.com/bliki/PageObject.html)? Yes

? Where are your page objects located? ./test/pageobjects/\*\*/\*.js

? Which reporter do you want to use? spec

? Do you want to add a plugin to your test setup?//wait for()

? Do you want to add a service to your test setup? (Press <space> to select, <a> to toggle all, <i> to invert selection, and <enter> to proceed)

>(\*) chromedriver

? What is the base url? http://localhost

? Do you want me to run `npm install` Yes

To start the test, run: $ npm run wdio

***To get Auto Suggestion..***

TO get AutoSuggestion create one .json file and copy paste below mentioned code

{

    "compilerOptions": {

        "types": [

            "node",

            "webdriverio/sync",

            "@wdio/mocha-framework"

        ]

    },

    "include" : [

        "test/specs/\*.js",

        "\*\*/\*.json",

        "node\_modules/@wdio/sync",

        "node\_modules/@wdio/mocha-framework"    ]}

To Run specs files parallelly:

Add Dependencies in package.json

"@wdio/selenium-standalone-service": "^7.7.4"

And replace this in wdio.conf.js=>service

services: ['selenium-standalone'],

To run specs files in different platform:

Modify as below in configuration file

For Chrome:

maxInstances: 5,

        // //

         browserName: 'chrome',

         acceptInsecureCerts: true,

For FireFox:

{

//     maxInstances: 5,

//     browserName: 'firefox',

//     acceptInsecureCerts: true,

// }

To Disable Chrome Options:

{ maxInstances: 5,

        // //

         browserName: 'chrome',

         acceptInsecureCerts: true,

         'goog:chromeOptions':

        {

              "excludeSwitches" : ["disable-popup-blocking"],"args" : ["incognito"] },

        'goog:chromeOptions': {

            prefs: {

              // 0 - Default, 1 - Allow, 2 - Block

              'profile.managed\_default\_content\_settings.notifications': 1

            }

          }

To stop and see application :

Await browser.debug();

**To Export and Import .js file**:

**From :**

const b= <http://localhost:8888>

Module.exports={b}

If we have function: function createEmp()

{

Console.log(“Emp1 is created”);

}

**Destination:**

Let Credentials=require(“../specs/demo”)

Console.log(Credentials.b)

Credentials.abc();

**To run particular test in it block**

Npx wdio run ./wdio.conf.js –mochaOpts.grep reg

No need to mention particular TC name

Bail:

It is used to make sure it will stop execution after particular set of TC are failed.

Ex: if I mention bail=1, it will stop execution when one of testcases gets failed.. it wont go for next TestCases for executions.

**ALLURE REPORT GENERATION**

**Step 1:** WebDriver IO document 🡺 Reporters 🡺 Allure Reporter🡺scroll down to command line 🡺 Right click on Allure-command-line-tool and open in new tab🡺 **npm install -g allure-commandline --save-dev**

Copy the second link and run it in terminal

**Step 2:** Installation

Copy the dev dependencies and paste it in package.json and give “npm install” command in the terminal

**"@wdio/allure-reporter": "^7.0.0"**

**Step 3**: Configuration

In the reporters section, add the following code and change the Screenshot to **“false”**

**reporters: [['allure', {  
 outputDir: 'allure-results',  
 disableWebdriverStepsReporting: true,  
 disableWebdriverScreenshotsReporting: true,  
 }]],**

**Step 4:** Run the test script, 🡺 “**allure results**” folder will be generated

**allure generate (path off allure results folder) ie allure generate allure-results”**

**Step 5:** To open the report give the command **“allure open”**

**Step 6:** To override the present report

**allure generate --clean allure-results** is used to override the present report

**Screenshot generation for failed test cases**

Change the **disableWebdriverScreenshotsReporting: true** to false Navigate to the WebDriver IO document🡺copy the code from the document and add the function in AfterTest hook

**async function (step, scenario, { error, duration, passed }, context) {  
 if (error) {  
 await browser.takeScreenshot();  
 }  
}**

 reporters: [['allure', {

        outputDir: 'allure-results',

        disableWebdriverStepsReporting: true,

        disableWebdriverScreenshotsReporting: false,

    }]],

afterTest:

    // function(test, context, { error, result, duration, passed, retries }) {

    // },

    async function (step, scenario, { error, duration, passed }, context) {

        if (error) {

          await browser.takeScreenshot();

        }

      },

**Video Report generation**

**Step 1**: Installation

Go to the WebDriver IO document 🡺 video Reporter🡺 installation division and execute the command “**npm install wdio-video-reporter**”

**Step 2**: configuaration (importing the video library) on top of the “wdio.config” file

**const video = require('wdio-video-reporter');**

**step 3:**

**Step 4:** Copy the reporters code from the document (**Using with allure code**) and **change the outputDir** to “(path of allujre results)”

   reporters: [

        [video, {

          saveAllVideos: false,       // If true, also saves videos for successful test cases

          videoSlowdownMultiplier: 3, // Higher to get slower videos, lower for faster videos [Value 1-100]

        }],

        ['allure', {

          outputDir: 'allure-results',

          disableWebdriverStepsReporting: true,

          disableWebdriverScreenshotsReporting: true,

        }],

      ],