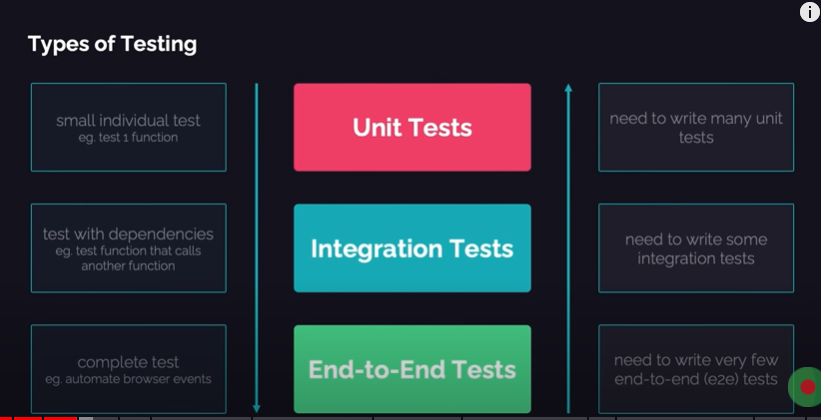
what is unit and integration tests:

Unit testing means **testing individual modules of an application in isolation** (without any interaction with dependencies) to confirm that the code is doing things right. Integration testing means checking if different modules are working fine when combined together as a group.

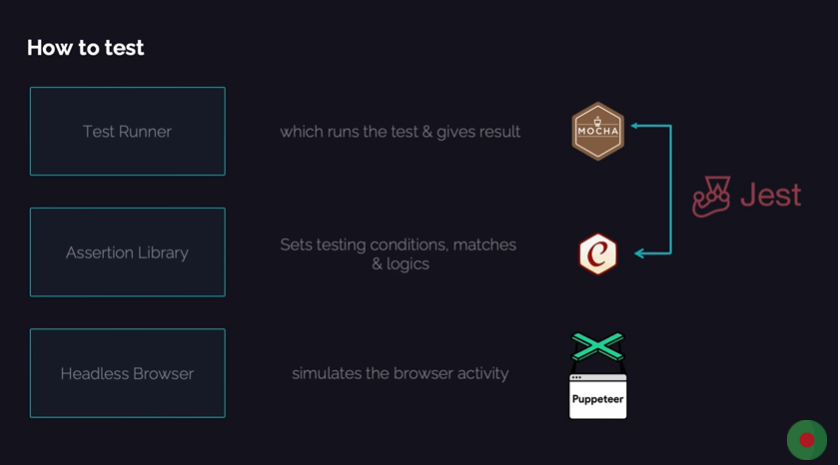
What is testing?

Software testing is important because if there are any bugs or errors in the software, they can be identified early and fixed before the software product is delivered.



How to test?

C = chi



Porject for understing:

Setup project

yarn init –y

Test gulo amra korbo development a amder production a kintu er dorkar nai.

yarn add –D jest @types/jest

or

yarn add --dev jest

yarn add @types/jest

amk jeno sugstion tai tai install korlam = @types/jest

Yarn er kono package install korte = yarn add name

Yarn er kono package uninstall korte = yarn remove name

In npm = npm install name, npm uninstall name

yarn add webpack webpack-cli –D

webpack = Webpack diya amder project k bundle kore felte pari jeno je kono jaigai use korte pari

webpack.config.js

const path = require('path');

module.exports = {

    entry: './js/app.js',

    output: {

        path: path.resolve(\_\_dirname),

        filename: "\_\_bundle.js"

    }

}

Index.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Testing in JavaScript</title>

    <link rel="stylesheet" href="./css/style.css" />

  </head>

  <body>

    <div class="container">

      <div class="form">

        <h1>Submit your article</h1>

        <input type="text" name="userid" id="userid" placeholder="User ID" />

        <input type="email" name="title" id="title" placeholder="Title" />

        <textarea

          name="article"

          id="article"

          placeholder="Type your article"

        ></textarea>

        <input

          type="button"

          value="Submit"

          class="btn-grad"

          id="addNewPostBtn"

        />

      </div>

      <div class="article-list result">

        Your created articles are listed below.

      </div>

    </div>

    <script src="./\_\_bundle.js"></script>

  </body>

</html>

{

  "name": "javascript-code-testing",

  "version": "1.0.0",

  "main": "index.js",

  "scripts": {

    "build": "webpack --mode production",

    "test": "jest --watchAll"

  },

  "license": "MIT",

  "dependencies": {

    "@types/jest": "^27.0.2"

  },

  "devDependencies": {

    "jest": "^27.3.1",

    "webpack": "^5.64.1",

    "webpack-cli": "^4.9.1"

  }

}

Yarn build.

yarn test or npm run test jai run kori akhon jest command run hobe.

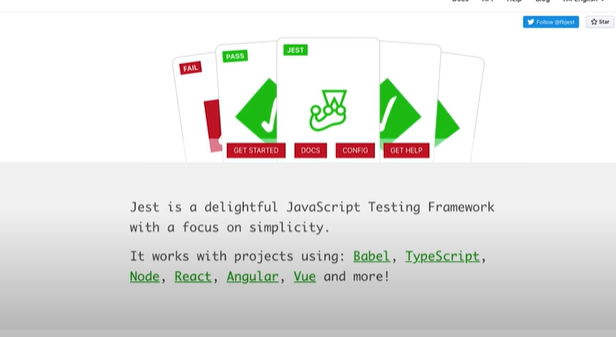
--watchAll = watchAll likle bar bar test er shomoi yarn test run korte hobe na.

Webpake ai khane app.js & lib.js k ek shate bundle kore diyase

Jest project a globally available hoiya jai.

Amr kono string value k trim korte parbo but kono number value k trim korte parbo na.

All are done with setup And Now testing things begine:



Fast test case:

Lib.test.js

const { generateResult } = require('../js/lib')

//test runner function

test("testing generate result", () => {//ki test korte chassi ai khane bolbo

    expect(generateResult("1", "This is a test")).toBe(

      "User ID: 1 created an article titled This is a test"

    );

})

Another unit test:

const { generateResult, validateInput } = require("../js/lib");

//unit test for generateResult()

test("testing generate result function", () => {

  expect(generateResult("1", "This is a test")).toBe(

    "User ID: 1 created an article titled This is a test"

  );

});

//unit test for validateInput()

test("testing validateInput function", () => {

  expect(validateInput(1, true, true)).toBeTruthy();

});

Integration test:

//integration test for checkAndGenerate()

test("testing checkAndGenerate function", () => {

  expect(checkAndGenerate(1, "Title 1", "This is a test")).toBe(

    "User ID: 1 created an article titled Title 1"

  );

});

End-to-End test:

E2e:

yarn add -D puppeteer

Chrome browaer k use kore e library puppeteer banana hoiyase. (chrome browser open sourse)

Puppeteer er shob kaj async.

//e2e testing for addPost()

test("checking e2e for addpost", async (done) => {

  const browser = await puppeteer.launch({

    headless: false,  //browser open hosse je amra dekte pabo na

    slowMo: 80, //onek taratari kaj kore dekte e pabo na tai slowMotom kore dilam

    args: ["--window-size=1920, 1080"],

  });

  const page = await browser.newPage();

  await page.goto("http://127.0.0.1:5500/");

  await page.click('input#useid');

  await page.type('input#userid', "1");

  await page.click('input#title');

  await page.type("input#title", "Article 1");

  await page.click('textarea#artcle');

  await page.click('textarea#artcle', "Article 1 description");

  await page.click('#addNewPositionBtn');

  await browser.close();

  done();

}, 20000)  //10 sec kore dilam

What is mocking?

Mocking mane kono kisu copy/clone kora.

browser a byDefault fetch thake js a install korte hoi na. But axious third party library.

Jodi amon hoi form filup korer por server a data save korbo save korer por je output ashbe ta show korbo browser a. Ai rokom case a mocking perfect.

So akhon request er jonno ekta library use korbo. And req er jonno fetch use korbo. But node js a byDefault fetch nai fetch install korte hobe.

Yarn add node-fetch

Akhon server dorkar akhon server to r banabo na a jonno json placeholder use korbo je khane fake api banate pari.

Website: jsonplaceholder

Response.json() kore amra data take json kori.

JSON.parse() ekta data k actual js object baniya dai

Ja korlam amra…. Amra je node-fetch library ta anici shai package tar ekta mock banalam.

Jest onek smark she byDefault joto package install kora ase she gulo k mock toiri kore automatically Jodi amra she tar mock baniya rakhi. byDefault gulo k bolte hoi na nijader banana hole bolte hoi.

Nijesho je kono function er e mock banana jai. jemon forEach , callback function she tar o amra chile mock toiri korte pari.

Express Application testing: From Andrew

Npm I supertest

User.test.js:

const request = require('supertest')

const app = require('../src/app')

test('Shouls sign a new user', async () => {

    await request(app).options('/users').send({

        name: "Mila",

        email: 'maksuda@gmail.com',

        password: 'MyPassword123'

    }).expect(201)

})

const userOne = {

    name: 'Mila',

    email: 'mila@gmail.com',

    password: 'mila000'

}

test('Shuld be signup a new user', async () => {

    await request(app).post("/user").send({

      name: "Mila",

      email: "mila@gmail.com",

      password: "mila000",

    });

})

test("Shuld login exsting user", async () => {

  await request(app).post("/user/login").send({

    email: userOne.email,

    password: userOne.password,

  });

});