**Testing Phase**

In our project we have used a Content Management System provided by Total.js. Total.js is a node-js framework hence all of its files are divided into modules. Each module contains the functions that are passed an object containing the present user state. While testing the code we implemented Black-Box testing on each individual module and White-Box testing on the internal functions.

Each file is stored as filename.spec.js along with the implementation file,

Sample Test file:

const fs = require('fs')

const request = require('request')

const expect = require('chai').expect

const page = require('./fileName')

describe('The fileName module', function () {

it('test case description', function \* () {

//Write test code

})

**Unit Testing:**

**Module: controller/default.js**

1. Function: view\_newsfeed

Test Code:

it( ‘returns the newsfeed.html file using render module’ ){

var result\_url = yield.self.url;

var result\_fileName = yield.self.path;

expect(result\_url).to.eql(‘/newsfeed’);

expect(result\_fileName).to.eql(‘/themes/default/views/newsfeed.html’);

};

1. Function: view\_homepage

Test Code:

it( ‘returns the index.html file using render module’ ){

var result\_url = yield.self.url;

var result\_fileName = yield.self.path;

expect(result\_url).to.eql(‘/’);

expect(result\_fileName).to.eql(‘/themes/default/views/index.html’);

};

1. Function: view\_contact

Test Code:

it( ‘returns the contact.html file using render module’ ){

var result\_url = yield.self.url;

var result\_fileName = yield.self.path;

expect(result\_url).to.eql(‘/contact’);

expect(result\_fileName).to.eql(‘/themes/default/views/contact.html’);

};

1. Function: view\_page

Test Code:

it( ‘returns the pages.js file using render module’ ){

var result\_url = yield.self.url;

var result\_fileName = yield.self.path;

expect(result\_url).to.eql(‘/’);

expect(result\_url).to.eql(‘/newsfeed);

expect(result\_url).to.eql(‘/contact’);

expect(result\_fileName).to.eql(‘/models/pages.js);

};

1. Function: file\_read

Test Code:

it( ‘reads a specific file from database’ ){

var req\_url = yield.req.url;

var id = yield.id;

var req\_ext = yield.req.extension;

expect(id).to.beDigit();

expect(req\_url).to.eql(‘/download/\*.\*’);

expect(req\_ext).to.noteql(‘jpg’);

expect(req\_ext).to.noteql(‘gif’);

expect(req\_ext).to.noteql(‘png’);

};

1. Function: file\_image

Test Code:

it( ‘reads a specific file from database’ ){

var req\_url = yield.req.url;

var id = yield.id;

var req\_ext = yield.req.extension;

expect(id).to.beDigit();

expect(req\_url).to.eql(‘/download/\*.\*’);

expect(req\_ext).to.eql(‘jpg’,’gif’,’png’);

};

1. Function: file\_read

Test Code:

it( ‘reads a specific file from database’ ){

var req\_url = yield.req.url;

var id = yield.id;

var req\_ext = yield.req.extension;

expect(id).to.beDigit();

expect(req\_url).to.eql(‘/download/\*.\*’);

expect(req\_ext).to.noteql(‘jpg’);

expect(req\_ext).to.noteql(‘gif);

expect(req\_ext).to.noteql(‘png’);

};

**Module: controller/api.js**

1. Function: json\_products\_categories

Test Code:

it( ‘populates the categories object’ ){

var result\_obj = yield.F.global.categories;

if(result\_obj.length > 0)

return true;

else

return false;

};

1. Function: json\_products\_query

Test Code:

it( ‘queries database for product details’ ){

var result\_query = yield.self.query;

if(result\_query.length > 0)

return true;

else

return false;

};

1. Function: json\_products\_read

Test Code:

it( ‘calls get request with a product id’ ){

var result\_id = yield.id;

expect(‘result\_id’).to.beDigit();

};

1. Function: json\_orders\_create

Test Code:

it( ‘creates a order entry in the database’ ){

var result\_id = yield.self.body.iduser;

expect(‘result\_id’).to.beDigit();

expect().to.becalledWith(self.callback());

};

1. Function: json\_orders\_read

Test Code:

it( ‘calls get request with an order id’ ){

var result\_id = yield.id;

expect(‘result\_id’).to.beDigit();

};

1. Function: json\_contact

Test Code:

it( ‘saves contact form’ ){

expect().to.becalledWith(self.callback());

};

**Module: controller/eshop.js**

1. Function: view\_products

Test Code:

it( ‘queries all products from database’ ){

var result\_search = yield.options.search;

var result\_page = yield.options.page;

expect(‘result\_search’).to.eql(yield.self.query.q);

expect(‘result\_page).to.eql(yield.self.query.page);

};

1. Function: view\_product\_categories

Test Code:

it( ‘queries database for product categories ){

var result\_category = yield.category;

var result\_repCategory = yield.repository.category;

if(result\_category.length > 0)

{

expect(‘result\_repCategory).to.eql(result\_category);

}

else

return false;

};

1. Function: view\_products\_details

Test Code:

it( ‘calls get request to get a products details’ ){

var result\_linker = yield.options.linker;

expect(‘result\_ linker).to.eql(yield.linker);

expect().to.becalledWith(self.$get);

};

1. Function: view\_checkout

Test Code:

it( ‘calls get request to get checkout list’ ){

var result\_linker = yield.options.linker;

expect(‘result\_ linker).to.eql(yield.linker);

expect().to.becalledWith(self.$get);

};

1. Function: view\_login

Test Code:

it( ‘populates user object’ ){

var result\_user = yield.user;

if(result\_user.length > 0)

return true;

else

return false;

};

1. Function: redirect\_account\_logoff

Test Code:

it( ‘closes current session and redirects to homepage’ ){

expect.to.becalledWith(Model(‘\*’).logoff());

expect(yield.self.sitemap.url).to.eql(‘account’);

};

1. Function: view\_blogs

Test Code:

it( ‘opens blog page’ ){

expect(yield.options.category).to.eql(‘Blogs’);

expect(yield.self.sitemap.url).to.eql(‘account’);

};

1. Function: view\_blogs\_details

Test Code:

it( ‘opens blog page’ ){

expect(yield.options.category).to.eql(‘Blogs’);

expect(yield.options.linker).to.eql(yield.linker);

};

**Module: controller/manager.js**

1. Function: redirect\_logoff

Test Code:

it( ‘closes current session and redirects to manager homepage’ ){

expect(yield.self.sitemap.url).to.eql(‘$manager’);

};

1. Function: json\_query

Test Code:

it( ‘Reads all items’ ){

var result\_query = yield.self.query;

var stub = yield.self;

if(result\_query.length > 0)

expect(stub).to.be.calledWith(stub.query, stub.callback());

else

return false;

};

1. Function: json\_save

Test Code:

it( ‘Save specific Item’ ){

var stub = yield.self.body;

expect(stub).to.be.calledWith(stub.callback());

};

1. Function: json\_remove

Test Code:

it( ‘Removes an items’ ){

var stub = yield.self;

expect(stub).to.be.calledWith(stub.body.id, stub.callback());

};

1. Function: json\_clear

Test Code:

it( ‘Clears all items’ ){

var stub = yield.self;

expect(stub).to.be.calledWith(‘clear’, stub.callback());

};

1. Function: json\_dashboard

Test Code:

it( ‘populates the dashboard model object’ ){

var result\_model = yield.model;

var result\_counter = yield.counter;

var stub = yield.self;

expect(result\_model.webcounter.today).to.eql(result\_counter.today());

expect(result\_model.webcounter.online).to.eql(result\_counter.online());

expect(result\_model.webcounter.today.pages).to.beDigit();

expect(result\_model.webcounter.stats).to.eql(yield.stats);

expect(stub).to.be.calledWith(result\_model);

};