UMGC Capstone Project Proposal Management System (CaPPMS)

Test Report

Version 1.0

Prepared By:

Bereket Tamrat

Ephrem Kefle

Kathryn Stewart

Marc Bueno

Tarun Lava

Yonas Mekete

Test Report Approvals

Name	Signature	Date
Professor Dr. Mir Assadullah		
Project Manager Kathryn Stewart		

Revision History

Date	Version	Description
11/3/2020	1.0	Initial Test Report

Table of Contents

List of Figures

List of Tables

1. Introduction

The purpose of this Test Report is to outline the purpose, scope, test strategies, and results from the UMGC Capstone Project Proposal Management System (CaPPMS). This document will cover the specific test cases that validate and verify the system in accordance with the Software Requirement Specification (SRS) and the Test Plan.

1.1. Background

The UMGC Capstone Project Proposal Management System (CaPPMS) is a web-based application which allows customer, clients, and former students to submit detailed proposals of projects for consideration to be designed and implemented by UMGC SWEN 670 students as well as track the stages during the approval process.

1.2. References

The following reference documents are applicable to this document.

Table 1 Reference Documents

Document	Version
CaPPMS Software Requirement Specification	2.0
CaPPMS Project Plan	3.0
CaPPMS Requirements Matrix	2.0

1.3. Definitions, Acronyms and Abbreviations

Below are the terms and abbreviations used in this document.

Table 2 Acronyms

Acronym	Definition
CaPPMS	Capstone Project Proposal Management System
CCB	Change Control Board
SDLC	Software Development Life Cycle
SRS	Software Requirement Specification
STP	Software Test Plan

Acronym	Definition
SWEN	Software Engineering
UMGC	University of Maryland Global Campus

2. Test Summary

The test was executed on 11/1/2020. The following key stakeholders were in attendance:

- Professor: Dr. Mir Assadullah
- Test Conductor: Yonas Mekete & Kathryn Stewart
- Quality Assurance: Tarun Lava

There was a decision to not implement one requirement, REQ-1.11: As the professor, I want a way to reset my password. This was not implemented due to a lack of time. This resulted in a deviation from the initial test plan.

As planned there was no performance testing completed for this test.

2.1. Scope

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as expected. The CaPPMS was designed and developed with two main areas to unit testing, namely the front-end (developed in Angular) and the back-end testing consisting of a RESTApi available through the various controllers (Unit-tested individually).

Each backend unit was tested separately before integrating into front-end modules to test the interfaces between front-end and backend in the CaPPMS. The two key criteria for unit test to be complete are when the code coverage is 100% and all possible inputs are tested. Code coverage refers to covering test cases to ensure the entire internal code logic is tested. Data coverage refers to the testing of wide range of data that is accepted by the program, usually done by boundary testing where data ranging from accepted limits of boundary lowest to highest is passed to the program as inputs for testing.

The front-end test was developed using protractor which works together with Selenium to provide automated testing that can simulate an actor's interactions with an Angular program running in a browser.

Table 3 Frontend Test Summary

Test ID	Test Name
1.	Navigate to the Homepage
2.	Navigate to the About page
3.	Navigate to the FAQ page
4.	Show Form Submit Process
5.	Show Form Error Messages
6.	Show Form Cancel Process
7.	Login
8.	Search and Filter
9.	Show Professor Form Error Messages
10.	Show Professor Cancel Button
11.	Edit Detail
12.	View Details
13.	export
14.	Create New Account
15.	Reset Password
16.	Delete Proposal

The back-end tests have followed the format below, in which each Test Case has a title, description, status, priority, method being tested, also mention clearly the initial configuration, software configuration, steps to recreate and expected behavior. Please note that the FileUpload controller was created but not implemented within this version of the CaPPMS. The unit tests are included for completeness.

Table 4 Backend Test Summary

Test ID	Test Name
1.	Usr_Type Test Case 1
2.	Usr_Type Test Case 2
3.	Usr_Type Test Case 3
4.	Usr_Type Test Case 4
5.	Status Test Case 1
6.	Status Test Case 2
7.	Status Test Case 3
8.	Status Test Case 4
9.	FAQ Test Case 1
10.	FAQ Test Case 2
11.	FAQ Test Case 3
12.	FAQ Test Case 4
13.	Users Test Case 1
14.	Users Test Case 2
15.	Users Test Case 3
16.	Users Test Case 4
17.	Users Test Case 5
18.	Project Test Case 1
19.	Project Test Case 2
20.	Project Test Case 3
21.	Project Test Case 4
22.	Project Test Case 5
23.	FileUpload Test Case 1
24.	FileUpload Test Case 2
25.	FileUpload Test Case 3
26.	FileUpload Test Case 4
27.	FileUpload Test Case 5

2.2. Metrics

There are 16 front end test cases that were planned for execution. There are 27 back end test cases that were planned for execution. Of these all test cases were executed.

Table 5 Test Case Metrics

Test tool	Planned Test Cases	Executed Test Cases	Passed Test Cases	Failed Test Cases
Selenium	16	16	16	0
JUnit	27	27	27	0

2.3. Defects

The CaPPMS currently has 9 defects opened against the frontend code and 1 defect open against the backend code. The details of these issues can be found in the GitHub repositories here: https://github.com/umgc/umgc.idea.tracker.git and https://github.com/umgc/umgc.idea.tracker.ui.git.

The listing of issues is included here but the most recent status will always be contained in the repository.

Table 6 Current Defect List

Defect Number	Defect Description	Priority	Codebase
16	Protractor Test: New proposal is not stored in the database	Low	umgc.idea.tracker.ui.git
17	Professor reset password function	Medium	umgc.idea.tracker.ui.git
18	Filter function by status	Low	umgc.idea.tracker.ui.git
19	Protractor test: New proposal is not stored in the database	Low	umgc.idea.tracker.ui.git

Defect Number	Defect Description	Priority	Codebase
20	Success message always displayed	Low	umgc.idea.tracker.ui.git
21	Dropdown box visibility	Low	umgc.idea.tracker.ui.git
23	Dynamic environment variables	High	umgc.idea.tracker.ui.git
25	Project Description Text Box Modifications	Low	umgc.idea.tracker.ui.git
26	Attachments	Low	umgc.idea.tracker.ui.git
22	Dynamic environment variables	High	umgc.idea.tracker.git

The definitions used for the classification of the issues are defined in the table below.

Table 7 Defect Definitions

Severity	Priority	Definition
Severity 3	Low	Not urgent or trivial. Minimal business impact.
Severity 2	Medium	Time-sensitive. Non-critical functionality loss or the software is still usable with a workaround.
Severity 1	High	Time-critical. Substantial loss of service or data loss.

2.4. Test Environment

The following tools were used to test the system:

- Junit5
- Protractor 7.0.0
- Microsoft Word, as part of Office 365

The CaPPMS system will be tested on the following operating system:

- Windows 10

The CaPPMS system will be tested on the following web browsers:

- Firefox 81.0
- Chrome 85.0.4183.121

2.5. Conclusion

The CaPPMS system successfully completed testing and met 96% of the functional requirements. The only requirement failing is REQ-1.11: As the professor, I want a way to reset my password. This requirement was not implemented and the link removed from the login page. The system is ready to be used locally but the high priority action items need to be addressed before the system can be used in a deployed environment.

3. Test Scripts

3.1. Front end script, spec.js

```
const { browser } = require("protractor");

//spec.js
describe('CaPPMS Homepage', function() {

  //REQ-1.1 As an unauthenticated user, I want to access the SWEN 670 home page via the internet.
  it('Navigate to the Homepage', function() {
     browser.driver.manage().window().maximize();
     browser.get('http://localhost:4200/home/');
     expect(browser.getCurrentUrl()).toBe('http://localhost:4200/home');
     browser.sleep(4000)

    });

    //REQ-1.2 As an unauthenticated user, I want to learn about the SWEN 670 capstone project.
    it('Navigate to the About page', function() {
      browser.get('http://localhost:4200/about/');
      expect(browser.getCurrentUrl()).toBe('http://localhost:4200/about');
      browser.sleep(4000)
    });

    //REQ-
1.6 As an unauthenticated user, I want to view frequently asked questions (FAQ) regarding the capstone projec
t.
    it('Navigate to the FAQ page', function() {
      browser.get('http://localhost:4200/faq/');
```

```
    expect(browser.getCurrentUrl()).toBe('http://localhost:4200/faq');
    browser.sleep(4000)
  });

  //REQ-1.3 As an unauthenticated user, I want to submit a project proposal idea for consideration.
  it('Show Form Submit Process', function(){
    browser.get('http://localhost:4200/home/');
    expect(browser.getCurrentUrl()).toBe('http://localhost:4200/home');

    browser.sleep(5000);
    browser.findElement(by.id("title")).sendKeys('Ms.');
    browser.findElement(by.id("first_name")).sendKeys('Kathryn');
    browser.findElement(by.id("Last_name")).sendKeys('Stewart');
    browser.findElement(by.id("phone_number")).sendKeys('123-456-1234');
    browser.findElement(by.id("email")).sendKeys('kathryn@gmail.com');
    //element(by.cssContainingText('usertypeselect', 'Sponsor')).click();
    //element(by.id("usertypeselect").sendKeys("Sponsor"));
    browser.findElement(by.id("project_title")).sendKeys('Kathryns Title');
    browser.findElement(by.id("project_description")).sendKeys('Kathryns Description');
    browser.findElement(by.id("project_website")).sendKeys('www.kathryn.com');
    browser.findElement(by.buttonText('Submit')).click();
    browser.sleep(4000);
  });

  //REQ-
1.5 As an unauthenticated user, I want to receive notification that required data has not been entered.
  it('Show Form Error Messages', function(){
    browser.get('http://localhost:4200/home/');
    expect(browser.getCurrentUrl()).toBe('http://localhost:4200/home');
    browser.findElement(by.id('first_name')).sendKeys('');
    browser.findElement(by.id("title")).sendKeys('');
    browser.findElement(by.id("first_name")).sendKeys('');
    browser.findElement(by.id("Last_name")).sendKeys('');
    browser.findElement(by.id("phone_number")).sendKeys('');
    browser.findElement(by.id("email")).sendKeys('');
    browser.findElement(by.id("project_title")).sendKeys('');
    browser.findElement(by.id("project_description")).sendKeys('');
    browser.findElement(by.id("project_website")).sendKeys('');
    //element(by.cssContainingText('option', 'Sponsor')).click();
    browser.findElement(by.buttonText('Submit')).click();
    browser.sleep(4000);
  });

  //REQ-1.4 As an unauthenticated user, I want to cancel a project proposal idea before submission.
  it('Show Form Cancel Process', function(){
```

```
  browser.get('http://localhost:4200/home/');
  expect(browser.getCurrentUrl()).toBe('http://localhost:4200/home');
  browser.sleep(2000);
  browser.findElement(by.name("title")).sendKeys('Ms.');
  browser.findElement(by.id("first_name")).sendKeys('Kathryn');
  browser.findElement(by.id("Last_name")).sendKeys('Stewart');
  browser.findElement(by.id("phone_number")).sendKeys('123-456-1234');
  browser.findElement(by.id("email")).sendKeys('kathryn@gmail.com');
  browser.findElement(by.id("project_title")).sendKeys('Kathryns Title');
  browser.findElement(by.id("project_description")).sendKeys('Kathryns Description');
  browser.findElement(by.id("project_website")).sendKeys('www.kathryn.com');
  //element(by.cssContainingText('option', 'Sponsor')).click();
  browser.findElement(by.buttonText('Cancel')).click();
  browser.sleep(4000);
});

//REQ-1.8 As the professor, I want an option to login to the system.
// REQ-1.12 As the professor, I want to login to view additional project details.
it('Login', function() {
  browser.get('http://localhost:4200/login/');
  expect(browser.getCurrentUrl()).toBe('http://localhost:4200/login');
  browser.findElement(by.id("username")).sendKeys('professoradmin');
  browser.findElement(by.id("password")).sendKeys('professorpassword');
  browser.sleep(1000);
  browser.findElement(by.buttonText('Login')).click();
  browser.sleep(1000);
});

//REQ-1.7 As an authenticated user, I want to view submitted project proposal ideas.
//REQ-1.13 As the professor, I want to search based on project status, i.e. Approved.
//REQ-1.14 As the professor, I want to filter based on project status, i.e. Approved.
it('Search and Filter', function() {
  browser.findElement(by.id("mat-input-0")).sendKeys('Kat');
  browser.sleep(2000);
});



//REQ-1.17 As the professor, I want to receive notification that required data has not been entered.
it('Show Professor Form Error Messages', function(){
  browser.findElement(by.buttonText('Update')).click();
  browser.findElement(by.id("project.user.title")).clear();
  browser.findElement(by.id("project.user.first_name")).clear();
  browser.findElement(by.id("project.user.last_name")).clear();
  browser.findElement(by.id("project.user.phone_number")).clear();
```

```
  browser.findElement(by.id("project.user.email")).clear();
  browser.findElement(by.id("project_title")).clear();
  browser.findElement(by.id("project_description")).clear();
  browser.findElement(by.id("project_website")).clear();
  browser.findElement(by.id("comment")).clear();
  browser.sleep(4000);
  browser.findElement(by.id("project.user.title")).sendKeys('');
  browser.findElement(by.id("project.user.first_name")).sendKeys('');
  browser.findElement(by.id("project.user.last_name")).sendKeys('');
  browser.findElement(by.id("project.user.phone_number")).sendKeys('');
  browser.findElement(by.id("project.user.email")).sendKeys('');
  browser.findElement(by.id("project_title")).sendKeys('');
  browser.findElement(by.id("project_description")).sendKeys('');
  browser.findElement(by.id("project_website")).sendKeys('');
  browser.findElement(by.id("comment")).sendKeys('');
  //element(by.cssContainingText('option', 'Sponsor')).click();
  browser.findElement(by.buttonText('Submit')).click();
  browser.sleep(4000);

});


 //REQ-1.18 As the professor, I want to cancel edits made to a specific proposal.
 it('Show Professor Cancel Button', function(){
  browser.findElement(by.buttonText('Update')).click();
  browser.sleep(2000);
  browser.findElement(by.name("project.user.title")).sendKeys('Ms.');
  browser.findElement(by.id("project.user.first_name")).sendKeys('Kathryn Updated');
  browser.findElement(by.id("project.user.last_name")).sendKeys('Stewart Updated');
  browser.findElement(by.id("project.user.phone_number")).sendKeys('123-456-1234');
  browser.findElement(by.id("project.user.email")).sendKeys('kathrynupdated@gmail.com');
  browser.findElement(by.id("project_title")).sendKeys('Kathryns Title Updated');
  browser.findElement(by.id("project_description")).sendKeys('Kathryns Description Updated');
  browser.findElement(by.id("project_website")).sendKeys('www.kathrynupdated.com');
  //element(by.id('usertypeselect', 'Sponsor')).click();
  //element(by.id('usertypeselect', 'Approved')).click();
  browser.findElement(by.buttonText('Cancel')).click();
  browser.sleep(4000);
 });

//REQ-1.16 As the professor, I want to add private comments to a specific proposal.
//REQ-1.19 As the professor, I want to edit the details of a specific proposal.
//REQ-1.22 As the professor, I want to be able to add a reason for rejection to a specific proposal.
it('Edit Details', function() {
  browser.findElement(by.buttonText('Update')).click();
```

```
    browser.sleep(2000);
    browser.findElement(by.name("project.user.title")).clear();
    browser.findElement(by.id("project.user.first_name")).clear();
    browser.findElement(by.id("project.user.last_name")).clear();
    browser.findElement(by.id("project.user.phone_number")).clear();
    browser.findElement(by.id("project.user.email")).clear();
    browser.findElement(by.id("project_title")).clear();
    browser.findElement(by.id("project_description")).clear();
    browser.findElement(by.id("project_website")).clear();
    //browser.findElement(by.cssContainingText('usertypeselect', 'Sponsor')).click();
    browser.findElement(by.id("comment")).clear();
    //browser.findElement(by.id('statusselect').sendKeys('Approved')).click();
    browser.findElement(by.name("project.user.title")).sendKeys('Ms.');
    browser.findElement(by.id("project.user.first_name")).sendKeys('Kathryn Updated');
    browser.findElement(by.id("project.user.last_name")).sendKeys('Stewart Updated');
    browser.findElement(by.id("project.user.phone_number")).sendKeys('123-456-1234');
    browser.findElement(by.id("project.user.email")).sendKeys('kathrynupdated@gmail.com');
    browser.findElement(by.id("project_title")).sendKeys('Kathryns Title Updated');
    browser.findElement(by.id("project_description")).sendKeys('Kathryns Description Updated');
    browser.findElement(by.id("project_website")).sendKeys('www.kathrynupdated.com');
    element(by.id('usertypeselect', 'Sponsor')).click();
    //element(by.cssContainingText('usertypeselect', 'Sponsor')).click();
    browser.findElement(by.id("comment")).sendKeys('This is a private comment that only the professor can s
ee and edit. ');
    element(by.id('statusselect', 'Approved')).click();
    browser.findElement(by.buttonText('Submit')).click();
    browser.sleep(2000);
  });

  //REQ-1.15 As the professor, I want to view the details of a specific proposal.
  it('View Details', function() {
    browser.sleep(1000);
    browser.findElement(by.buttonText('Details')).click();
    browser.sleep(1000);
  });



  //REQ-1.21 As the professor, I do not want the private comments to be included in the export.
  //REQ-1.23 As the professor, I want to export a specific proposal in Word format.
  it('export', function() {
    browser.findElement(by.buttonText('Export to word doc')).click();
    //long wait here to open word doc manualy
    browser.sleep(7000);
    browser.navigate().back();
```

```
  });

  //REQ-1.9 As the professor, I want to be able to create an additional authenticated account.
  //REQ-
1.10 As the professor, I want to be notified if a new authenticated account already exists when creating.
      it('Create New Account', function() {
        browser.get('http://localhost:4200/newadmin');
        browser.findElement(by.id("user.first_name")).sendKeys('Kathryn');
        browser.findElement(by.id("user.last_name")).sendKeys('Stewart');
        browser.findElement(by.id("user.email")).sendKeys('kathryn@gmail.com');
        browser.findElement(by.id("account.password")).sendKeys('password');
        browser.findElement(by.id("password2")).sendKeys('password');
        browser.findElement(by.buttonText('Register')).click();
        browser.sleep(2000);

        browser.findElement(by.id("user.first_name")).sendKeys('Kathryn');
        browser.findElement(by.id("user.last_name")).sendKeys('Stewart');
        browser.findElement(by.id("user.email")).sendKeys('kathryn@gmail.com');
        browser.findElement(by.id("account.password")).sendKeys('password');
        browser.findElement(by.id("password2")).sendKeys('password');
        browser.findElement(by.buttonText('Register')).click();
        browser.sleep(2000);
  });

  //REQ-1.11 As the professor, I want a way to reset my password.
  it('Reset Password', function() {
    //Functionality not implemented
  });

//REQ-1.20 As the professor, I want to delete a specific proposal.
  it('Delete Proposal', function() {

    browser.get('http://localhost:4200/proposal');
    //Test breaks the database. Run manually.
    //browser.findElement(by.buttonText('Delete')).click();

    });
});
```

3.2. Back end scripts

3.2.1. Usr_Type Controller

3.2.1.1. Test Case 1

Test case name: GetAllUserType

Method being tested:

@GetMapping("/usr_type")
List<Usr_Type> getAllUsr_Type()

Short Description: This method returns a list of Usr_Types

Input Data to constructor and/or method you are testing: None

Expected Results: Http 200 OK - List with all user types stored in the database.

3.2.1.2. Test Case 2

Test case name: CreateUserType

Method being tested:

@PostMapping("/usr_type")

public Usr_Type createUsr_Type(@RequestBody Usr_Type usr_type)
Short Description: This method creates a new User_Type and returns the same as the method return value.

Input Data to constructor and/or method you are testing: id (auto_increment), utype_descr

Expected Results: Http 200 OK + newly created user type

3.2.1.3. Test Case 3

Test case name: GetUserTypeById

Method being tested:
@GetMapping("/usr_type/{id}")

public ResponseEntity<Usr_Type> getUsr_TypeById(@PathVariable Long id)
Short Description: This method returns a Usr_Types Object matching the id

Input Data to constructor and/or method you are testing: id

Expected Results: HTTP 200 OK + usr_type object

3.2.1.4. Test Case 4

Test case name: UpdateUserType

Method being tested:
@PutMapping("/usr_type/{id}")

public ResponseEntity<Usr_Type> updateUsr_Type(@PathVariable Long id, @RequestBody Usr_Type usr_typeDetails)
Short Description: This method updates the Usr_Type referred to by “id”. Input data include utype_descr_

Expected Results: http code 200 OK. ResponseEntity<Usr_Type>

3.2.2. Status Controller

3.2.2.1. Test Case 1

Test case name: GetAllStatus

Method being tested:

@GetMapping("/status")

public List<Status> getAllStatus()

Short Description: This method returns a list of Status

Input Data to constructor and/or method you are testing: None

Expected Results: Http 200 OK - List with all Status stored in the database.

3.2.2.2. Test Case 2

Test case name: CreateStatus

Method being tested:

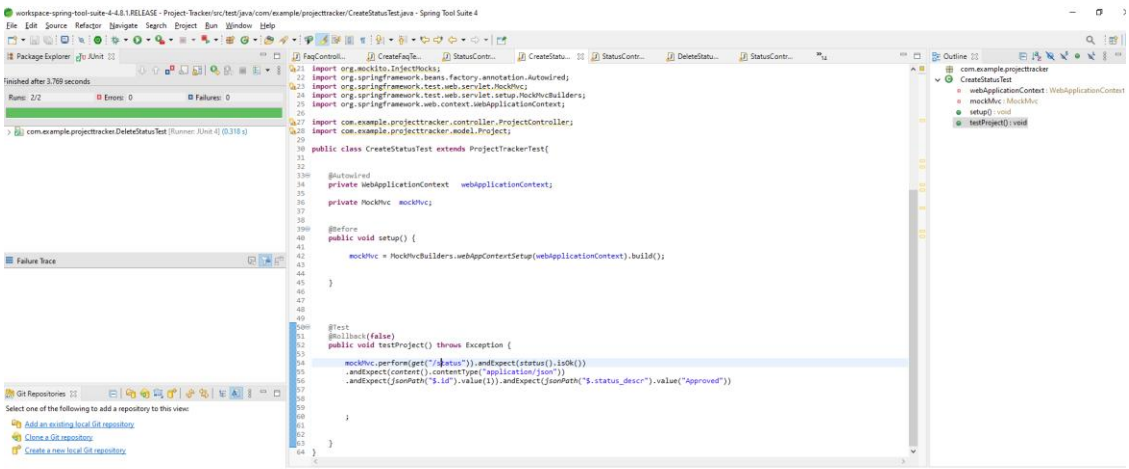
@PostMapping("/status")

public Status createStatus(@RequestBody Status status)

Short Description: This method creates a new Status and returns the same as the method return value.

Input Data to constructor and/or method you are testing: id (auto_increment), status_descr

Expected Results: Http 200 OK + newly created status



3.2.2.3. Test Case 3

Test case name: GetStatusById

Method being tested:

@GetMapping("/status/{id}")

public ResponseEntity<Status> getStatusById(@PathVariable Long id)

Short Description: This method returns a Status Object matching the id

Input Data to constructor and/or method you are testing: id

Expected Results: HTTP 200 OK + Status object

3.2.2.4. Test Case 4

Test case name: UpdateStatus

Method being tested:

@PutMapping("/status/{id}")

public ResponseEntity<Status> updateStatus(@PathVariable Long id, @RequestBody Status statusDetails)

Short Description: This method updates the Status referred to by “id”. Input data include status_descr

Expected Results: http code 200 OK. ResponseEntity<Status>

3.2.2.5. Test Case 5

Test case name: DeleteStatus

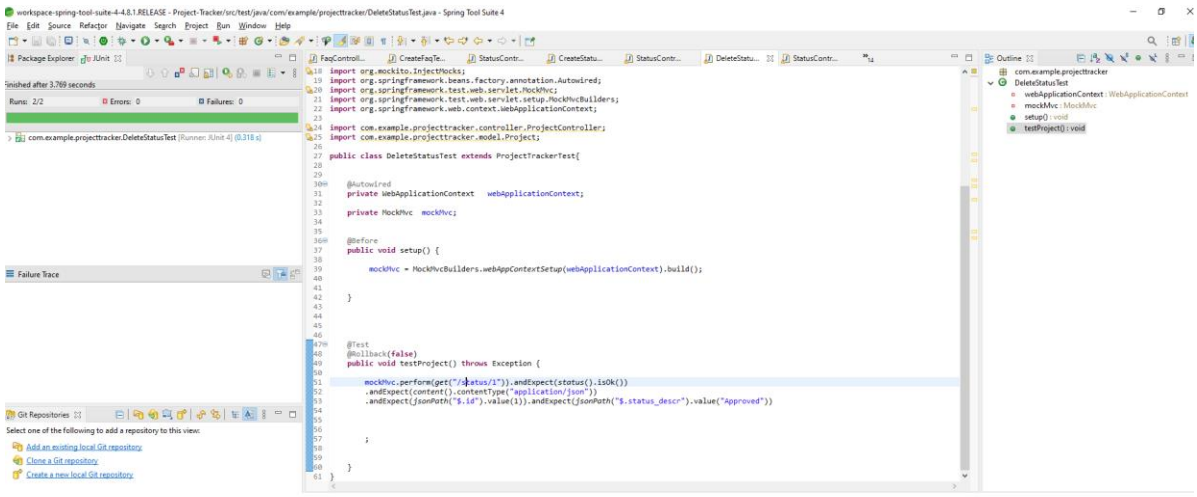
Method being tested:

@DeleteMapping("/status/{id}")

public ResponseEntity <Map<String, Boolean>> deleteStatus(@PathVariable Long id)Short

Description: This method deletes the Status referred to by “id”. Input data include: id

Expected Results: http code 200 OK. ResponseEntity<Status, true>



3.2.3. FAQ Controller

3.2.3.1. Test Case 1

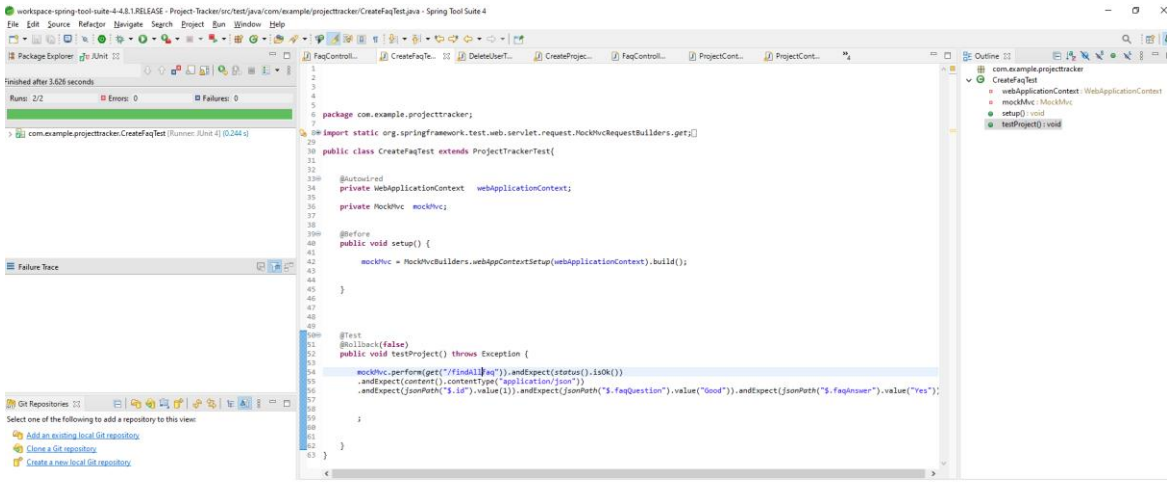


Figure 1 FAQ Test Case 1

Test case name: GetAllFAQ

Method being tested:

@GetMapping("/faq")

public List<FAQ> getAllFAQ()

Short Description: This method returns a list of FAQ

Input Data to constructor and/or method you are testing: None

Expected Results: Http 200 OK - List with all FAQ stored in the database.

3.2.3.2. Test Case 2

Test case name: CreateFAQ

Method being tested:

@PostMapping("/faq")

public FAQ createStatus(@RequestBody FAQ faq)

Short Description: This method creates a new FAQ and returns the same as the method return value.

Input Data to constructor and/or method you are testing: id (auto_increment), faq_question, faq_answer

Expected Results: Http 200 OK + newly created FAQ

3.2.3.3. Test Case 3

Test case name: GetFAQById

Method being tested:
@GetMapping("/faq/{id}")

public ResponseEntity<FAQ> getFAQById(@PathVariable Long id)
Short Description: This method returns a FAQ Object matching the id

Input Data to constructor and/or method you are testing: id

Expected Results: HTTP 200 OK + FAQ object

3.2.3.4. Test Case 4

Test case name: UpdateFAQ

Method being tested:
@PutMapping("/faq/{id}")

public ResponseEntity<FAQ> updateStatus(@PathVariable Long id, @RequestBody FAQ faqDetails)
Short Description: This method updates the FAQ referred to by “id”. Input data include faq_question_faq_answer

Expected Results: http code 200 OK. ResponseEntity<FAQ>

3.2.4. User Controller

3.2.4.1. Test Case 1

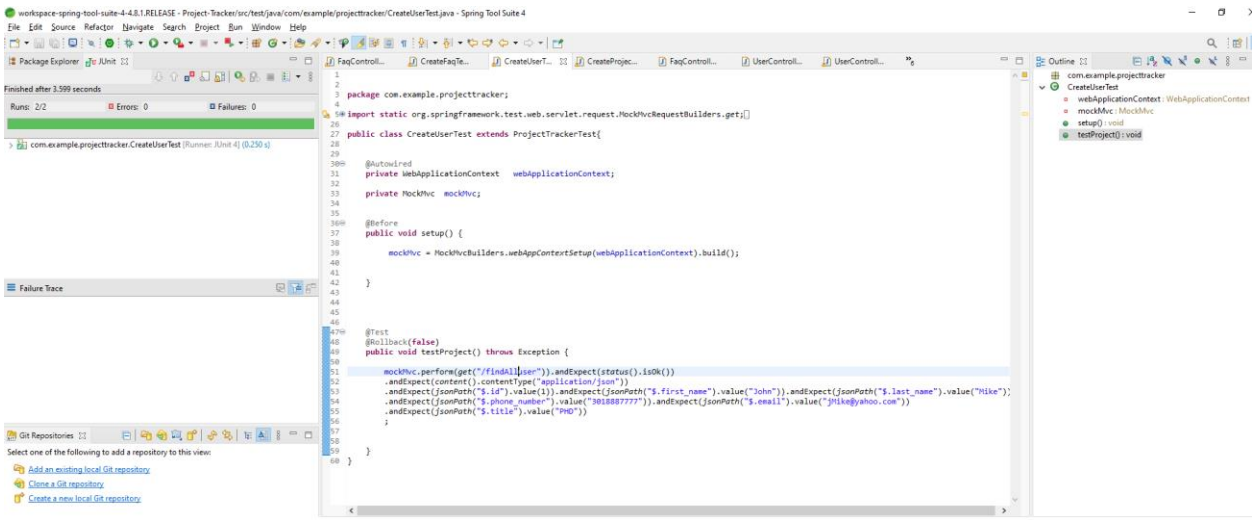


Figure 2 User Test Case 1

Test case name: GetAllUsers

Method being tested:

@GetMapping("/user")

public List<User> getAllUser(){

Short Description: This method returns a list of all Users

Input Data to constructor and/or method you are testing: None

Expected Results: Http 200 OK - List with all user stored in the database.

3.2.4.2. Test Case 2

Test case name: CreateUser

Method being tested:

@PostMapping("/user")

public User createUser(@RequestBody User user)

Short Description: This method creates a new User and returns the same as the method return value.

Input Data to constructor and/or method you are testing: id (auto_increment), first_name, last_name, phone_number, email, title, usr_type, website, empl_id, account

Expected Results: Http 200 OK + newly created user

3.2.4.3. Test Case 3

Test case name: GetUserById

Method being tested:
@GetMapping("/user/{id}")

public ResponseEntity<User> getUserById(@PathVariable Long id)
Short Description: This method returns a User Object matching the id

Input Data to constructor and/or method you are testing: id

Expected Results: HTTP 200 OK + user object

3.2.4.4. Test Case 4

Test case name: UpdateUserById

Method being tested:
@PutMapping("/user/{id}")

public ResponseEntity<User> updateUser(@PathVariable Long id, @RequestBody User userDetails)
Short Description: This method updates the User referred to by “id”. Input data include first_name, last_name, phone_number, email, title, usr_type, website, empl_id, account

Expected Results: http code 200 OK. ResponseEntity<User >

3.2.4.5. Test Case 5

Test case name: DeleteUserById

Method being tested:
@DeleteMapping("/user/{id}")

public ResponseEntity <Map<String, Boolean>> deleteUser(@PathVariable Long id)
Short Description: This method deletes the User referred to by “id”.

Expected Results: http code 200 OK. ResponseEntity<User >

3.2.5. Project Controller

3.2.5.1. Test Case 1

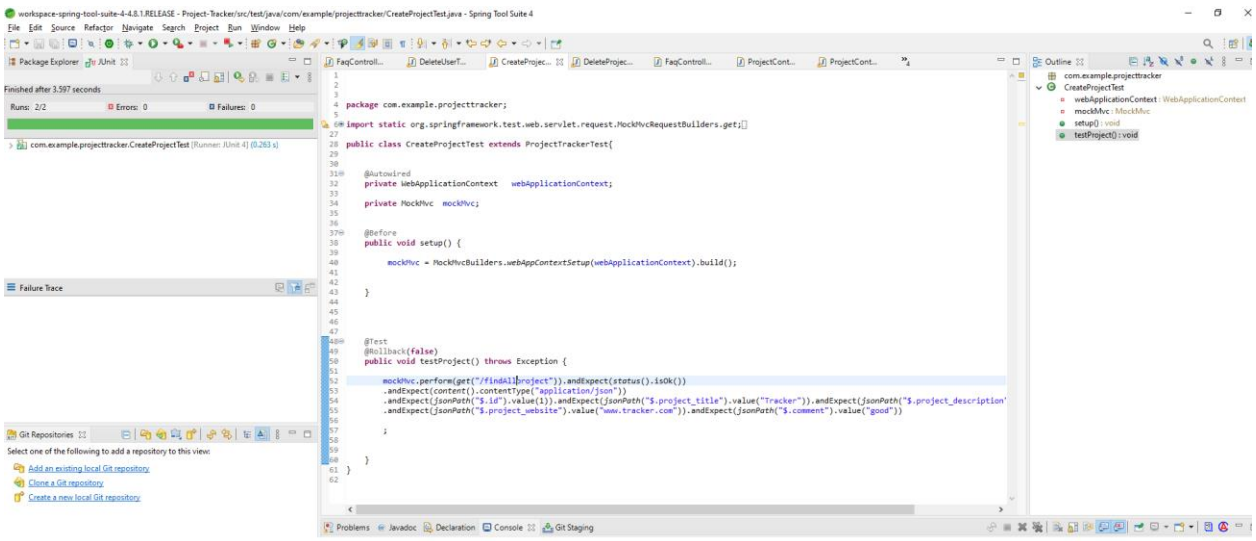


Figure 3 Project Test Case 1

Test case name: GetAllProject

Method being tested:

@GetMapping("/project")

public List<Project> getAllProject()

Short Description: This method returns a list of Projects

Input Data to constructor and/or method you are testing: None

Expected Results: Http 200 OK - List with all projects stored in the database.

3.2.5.2. Test Case 2

Test case name: CreateProject

Method being tested:

@PostMapping("/project")

public Project createProject(@RequestBody Project project)

Short Description: This method creates a new Project and returns the same as the method return value.

Input Data to constructor and/or method you are testing: id (auto_increment), project_description, project_title, project_website, project_status, users, attachments

Expected Results: Http 200 OK + newly created project

3.2.5.3. Test Case 3

Test case name: GetProjectById

Method being tested:
@GetMapping("/project/{id}")

public ResponseEntity<Project> getProjectById(@PathVariable Long id)
Short Description: This method returns a Project Object matching the id

Input Data to constructor and/or method you are testing: id

Expected Results: HTTP 200 OK + project object

3.2.5.4. Test Case 4

Test case name: UpdateProject

Method being tested:

@PutMapping("/project/{id}")
public ResponseEntity<Project> updateProject(@PathVariable Long id, @RequestBody Project projectDetails)
Short Description: This method updates the Project referred to by “id”. Input data include project_description, project_title, project_website, project_status, users, attachments

Expected Results: http code 200 OK. ResponseEntity<Project>

3.2.5.5. Test Case 5

Test case name: DeleteProjectById

Method being tested:
@DeleteMapping("/project/{id}")

public ResponseEntity <Map<String, Boolean>> deleteProject(@PathVariable Long id)
Short Description: This method updates the Project referred to by “id”.

Expected Results: http code 200 OK. ResponseEntity<Project>

3.2.6. File Upload Controller

3.2.6.1. Test Case 1

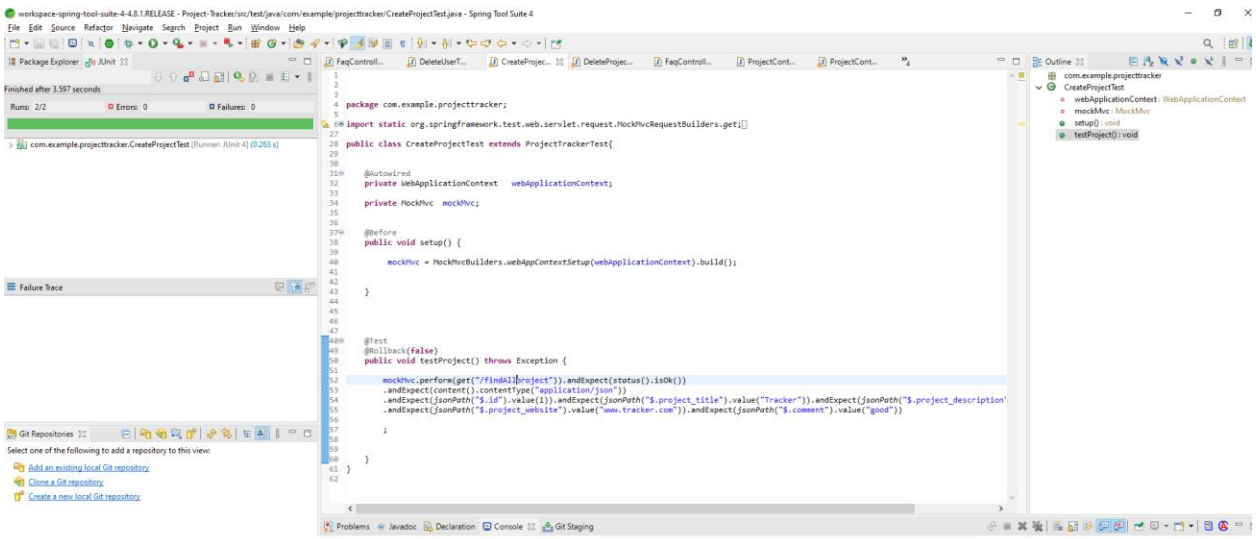


Figure 4 FileUpload Test Case 1

Test case name: GetAllProject

Method being tested:

@GetMapping("/project")

public List<Project> getAllProject()

Short Description: This method returns a list of Projects

Input Data to constructor and/or method you are testing: None

Expected Results: Http 200 OK - List with all projects stored in the database.

3.2.6.2. Test Case 2

Test case name: CreateProject

Method being tested:

@PostMapping("/project")

public Project createProject(@RequestBody Project project)

Short Description: This method creates a new Project and returns the same as the method return value.

Input Data to constructor and/or method you are testing: id (auto_increment), project_description, project_title, project_website, project_status, users, attachments

Expected Results: Http 200 OK + newly created project

3.2.6.3. Test Case 3

Test case name: GetProjectById

Method being tested:

@GetMapping("/project/{id}")

public ResponseEntity<Project> getProjectById(@PathVariable Long id)

Short Description: This method returns a Project Object matching the id

Input Data to constructor and/or method you are testing: id

Expected Results: HTTP 200 OK + project object

3.2.6.4. Test Case 4

Test case name: UpdateProject

Method being tested:

public ResponseEntity<Project> updateProject(@PathVariable Long id, @RequestBody Project projectDetails)

Short Description: This method updates the Project referred to by “id”. Input data include project_description, project_title, project_website, project_status, users, attachments

Expected Results: http code 200 OK. ResponseEntity<Project>

3.2.6.5. Test Case 5

Test case name: DeleteProjectById

Method being tested:

@DeleteMapping("/project/{id}")

public ResponseEntity <Map<String, Boolean>> deleteProject(@PathVariable Long id)

Short Description: This method updates the Project referred to by “id”.

Expected Results: http code 200 OK. ResponseEntity<Project>

4. Test Execution Reports

Table 8 Protractor Front End Test

Test Name:	spec.js
Test Start Date & Time:	11/1/2020
Test End Date & Time:	11/1/2020
Test Conductor:	Kathryn Stewart
Software Versions:	
Test Environment:	Local
Quality Assurance:	Tarun Lava
Other Witnesses:	Yonas Mekete, Ephrem Kefle, Bereket Tamrat, Marc Bueno, Roy Gordon, Mir Assadullah
Deviations:	None
Issues Opened:	None
Action Items Opened:	Remove the forced failure of the tests to ensure a clean run prior to deployment.
Status:	☐ Pass ☒ Pass w/exceptions ☐ Fail
Test Conductor Signature/ Date:	*Kathryn Stewart* 11/1/2020
QA Signature/ Date:	*Tarun Lava* 11/1/2020
Customer Signature/ Date:	Roy