**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

Facultad de Ingeniería en Electricidad y Computación

**Ingeniería en software 2**

Paralelo #3

**Documentación del proyecto parte 1**

**Grupo:** Service-Soft

**Miembros del grupo:**

Scarlet Angelina Espinoza Moreno

**Fecha de entrega:**

26/06/2021

**Nombre del profesor:**

Ing. Monica K. Villavicencio Cabezas

**Término Académico:**

2021-1S

Contents

[1- Git Repository (e.g. Bitbucket, GitHub) for Code and Documents. 3](#_Toc75708023)

[2- Use of a Build Automation Tool for the Project. 3](#_Toc75708024)

[3- Evidence of adherence to SCRUM: 4](#_Toc75708025)

[4- Coding Standard Documentation Detecting and Enforcing Coding Standards. 10](#_Toc75708026)

[5- Preemptive Error Detection 16](#_Toc75708027)

[6- Definition and Execution of Test Cases. 20](#_Toc75708028)

[7- System Deployment Guide. 21](#_Toc75708029)

[7.1- System Deployment Guide App Movil 21](#_Toc75708030)

[7.2- System Deployment Guide App Web 21](#_Toc75708031)

[8- Video showing the sprint review process. 22](#_Toc75708032)

[9- Software system deployed into a production environment. 22](#_Toc75708033)

# Git Repository (e.g. Bitbucket, GitHub) for Code and Documents.

We work in 3 gitHub because the product owner recommends that so for that reason, we have 3 links:

* **Web App:** <https://github.com/scarletespinozaMoreno/my-app>
* **Movil App:** <https://github.com/gejuriera/Sprint2Soft2>
* **Panel Admin:** <https://github.com/albinisma96/AgregarPromociones>

For the sprint#2 was not necessary do the panel control for the admin but we decided to do it to continuous.

# Use of a Build Automation Tool for the Project.

We use node.js in special the use o npm to make a build automation of the project this action create a folde that contain all the project ready to build directaly.

Graphical user interface, application

Description automatically generated

# Evidence of adherence to SCRUM:

|  |  |
| --- | --- |
| **ASISTENTES** | |
| **NOMBRES** | **CARGO** |
| Victor Alvarado | Jefe de Proyecto |
| Christian Portilla | Líder de Proyecto |
| Scarlet Epinoza | Scrum Master |
| Albin Arias | Desarrollador |
| Leonardo Castro | Product Owner |
| Génesis Riera | Desarrollador |

TASK#7

Graphical user interface, text, application, Word

Description automatically generated

Graphical user interface, text, application, email

Description automatically generatedSUBTASKS#7

Graphical user interface, text, application

Description automatically generatedATTACHMENTS

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, application

Description automatically generatedTASK#8  
SUBTAKS#8

Graphical user interface, application, website

Description automatically generatedAttachments

Graphical user interface, text, application, email

Description automatically generatedTASK#9

Graphical user interface, text, application, email

Description automatically generatedSUBTAKS#9

ATTACHMENT

Graphical user interface, text, application

Description automatically generated

Timeline

Description automatically generatedProduct Backlog finished , task in general

Timeline

Description automatically generated with medium confidencePer subtaks#6

Timeline

Description automatically generatedPer subtaks#7

Per subtaks#8

Text

Description automatically generatedPersubtaks#9

Graphical user interface, text, application, chat or text message

Description automatically generated

# Graphical user interface, application Description automatically generatedCoding Standard Documentation Detecting and Enforcing Coding Standards.

The Coding Standard in the project was realized with **SonarQube**, it is is a platform for evaluating source code. It is free software and **uses various static code analysis tools such as Checkstyle, PMD or FindBugs** to obtain metrics that can help improve the quality of a program's code.

**Evidence from the web:**

**Chart

Description automatically generated with low confidenceCalcification that sonar gave to our app web:**

**Graphical user interface, text, application

Description automatically generated**Graphical user interface, text, application

Description automatically generated**Some Rules applied:**

**Bugs:**

Chart, bubble chart

Description automatically generated**Grafic:**

Graphical user interface, text, application, email

Description automatically generated**Measures:**

**Evidence from the App Movil:**

**Calcification that sonar gave to our app movil:**

Chart

Description automatically generated

**Some Rules applied:**

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated**Bugs:**

Chart

Description automatically generated**Grafic:**

Graphical user interface, text, application

Description automatically generated**Measures:**

# Preemptive Error Detection

**PMD** is a static source code analyzer. It finds common programming flaws like unused variables, empty catch blocks, unnecessary object creation, and so forth. It’s mainly concerned with **Java and Apex.**

**The first step we must do is install "Apex PMD", "Java test runner" and "Maven for Java '**

A screenshot of a computer

Description automatically generated with medium confidence

Then you go to the view section and open the command palette and open the command “ Mave: Create maven project”. We choose the latest version and then we write the name of the folder in which the files will be found with the analysis of the errors of the project.

A picture containing text, screenshot, black

Description automatically generated

Then we will use the command "Maven: Create Maven Project", from there you select the option that allows you to open a java application, you choose the latest version in this case 2.0.0, well that's how you are creating the tests for the project. the name of the folder in this case I put “Teste” A screenshot of a computer

Description automatically generated

And the teste folder of the project is automatically opened in visual studio, ntonces in the test folder some files are automatically created, but we must create the ‘rules.xml” file.<rule ref="category/java/documentation.xml" /> That line of code extracts the Java documentation in which all the error sections that you may have in your project are found so as not to be putting all of them because there are quite a few

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidenceOnce you have that you go back to the command palette and there you put "ferences: Open user settings. You look for the option" Apex: PMD: Additional Classes Paths, there you are going to add the Apex path of the roules.xml file.

Once you add it, you go to the App.java test document and put the Apex Static Alalysis option on workspace, and there you have the errors.

A screenshot of a computer

Description automatically generated with medium confidence

We can see that the "test" folder was created in the project.Graphical user interface, application, table

Description automatically generated

# Definition and Execution of Test Cases.

Tool used: JEST

Text

Description automatically generated

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

# System Deployment Guide.

For the system Deployment Guide, we made two guides and also 2 videos to provide a evidence and how the apps works.

**System Deployment Guide Video App Movil**: <https://www.youtube.com/watch?v=ZayJHYwC0tA>

**System Deployment Guide Video App web:** [**https://youtu.be/O55gWDmriZ4**](https://youtu.be/O55gWDmriZ4)

## 7.1- System Deployment Guide App Movil

Look in the archive of the project, there is a .pdf

## 7.2- System Deployment Guide App Web

Look in the archive of the project, there is a .pdf

# Video showing the sprint review process.

* Link of the video: <https://drive.google.com/file/d/1fQZYG1I-j9jY83Jjj8wXlPPbr_MiY-1K/view?usp=sharing>
* If the link before does not work you can also download in the git repository:

# Software system deployed into a production environment.

Production environment for the App web was in the hosting of firebase.

**Link of the app web deployed:** <https://hosteria-95a60.firebaseapp.com/>

**App Movil apk:** Look in the GitHub repository <https://github.com/scarletespinozaMoreno/Rrecursos-Documentacion-Proyecto-Ing2-ServiceSoft>

there is a .apk