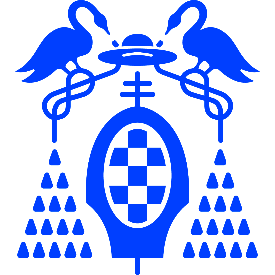
Trabajo en grupo 3: automated testing frameworks for php

Grupo T8

Universidad de Alcalá

Desarrollo con Tecnologías Emergentes



Daniel Manzano Estébanez

Inés López Baldominos

José María Sanz Górriz

Adrián de la Hoz Casanova

Cristian Abellán Madrigal

Contenido

[1. Autores del trabajo, planificación y entrega 2](#_Toc8075498)

[1.1 Autores 2](#_Toc8075499)

[1.2 Planificación 2](#_Toc8075500)

[1.3 Entrega 2](#_Toc8075501)

[3. Criterios de comparación en la implementación 3](#_Toc8075502)

[3.1 Criterio 1: Dificultad en obtener e instalar el software necesario 3](#_Toc8075503)

[3.2 Criterio 2: Recursos necesarios para el funcionamiento de las herramientas de testeo automático 3](#_Toc8075504)

[3.3 Criterio 3: Tiempo de respuesta funcionamiento del sistema 3](#_Toc8075505)

[3.4 Criterio 4: Información para utilizar la herramienta 3](#_Toc8075506)

[3.5 Criterio 5: Implementación intuitiva 3](#_Toc8075507)

[7. Comparación de la implementación de las tecnologías 4](#_Toc8075508)

[8. Conclusiones 5](#_Toc8075509)

# 1. Autores del trabajo, planificación y entrega

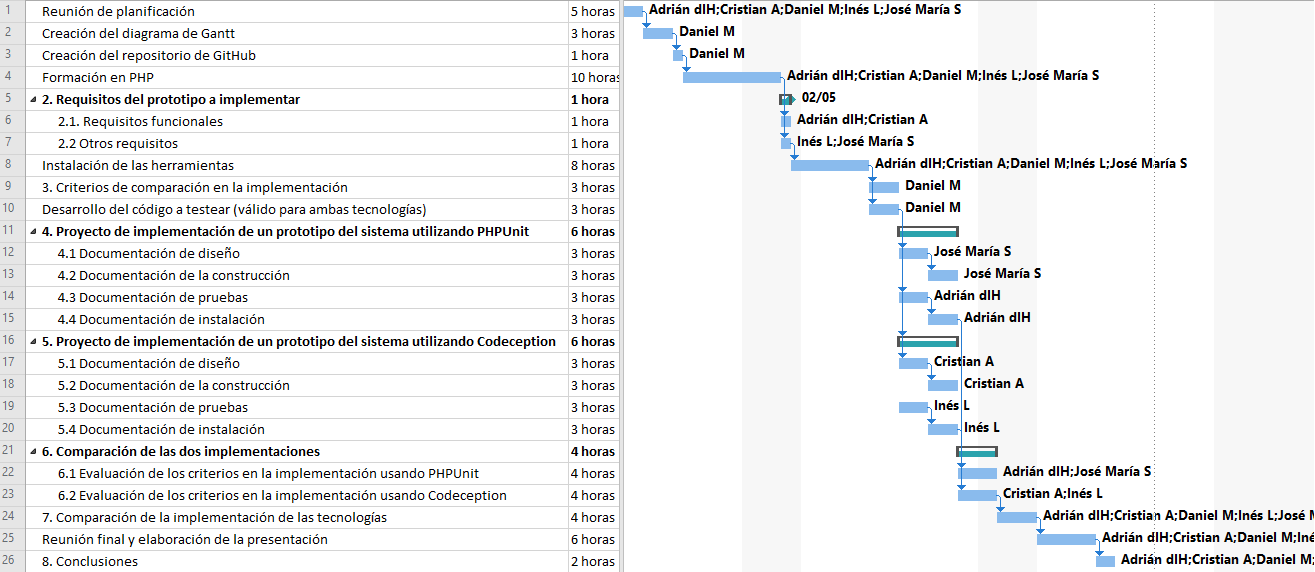
## 1.1 Autores

Grupo T8

* Daniel Manzano Estébanez (Coordinador)
* Cristián Abellán Madrigal
* Inés López Baldominos
* José María Sanz Górriz
* Adrián de la Hoz Casanova

## 1.2 Planificación

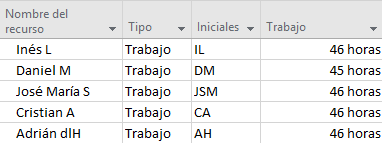
Con la ayuda de Microsoft Project, se ha elaborado un diagrama de Gantt en el que se han registrado las diferentes tareas de este trabajo y su duración, que han sido asignadas a los diferentes miembros del grupo. Este archivo se encuentra en el repositorio de GitHub que se indica en el apartado 1.3. Esta es la planificación realizada.

**

*\* Las tareas 1, 4, 8, 24, 25 y 26 han sido llevadas a cabo por todos los miembros del equipo.*

Las tareas siguen la estructura de este trabajo y, además, se han añadido tareas correspondientes a la preparación de este, formación en PHP, instalación de las herramientas y la reunión de finalización.

Con el reparto de tareas, se ha logrado cumplir con las 45 horas exigidas para este trabajo, que se corresponde con el 30% de la asignatura. A continuación se puede ver a los miembros del grupo en Microsoft Project con el total de horas de cada uno.

**

## 1.3 Entrega

El trabajo realizado se encuentra disponible en la siguiente dirección de GitHub: <https://github.com/daniel-manzanoe/TG3>.

# 3. Criterios de comparación en la implementación

## 3.1 Criterio 1: Dificultad en obtener e instalar el software necesario

Nombre del criterio: Dificultad en obtener e instalar el software necesario

Descripción: Complejidad encontrada en el proceso de conseguir los programas necesarios e instalarlos para realizar los test automáticos en PHP.

Tipo de valor: Numérico (horas).

## 3.2 Criterio 2: Recursos necesarios para el funcionamiento de las herramientas de testeo automático

Nombre del criterio: Recursos necesarios para el funcionamiento de las herramientas de testeo automático.

Descripción: Cantidad de herramientas software que hemos tenido que instalar para poder realizar los test automáticos en PHP.

Tipo de valor: Numérico (cantidad de herramientas).

## 3.3 Criterio 3: Tiempo de respuesta funcionamiento del sistema

Nombre del criterio: Tiempo de respuesta del sistema

Descripción: Tiempo que tarda en realizar el testeo automático. Este criterio va a evaluarse en función del tiempo que ambas herramientas tardan en realizar el testeo de un mismo código.

Tipo de valor: Numérico (segundos).

## 3.4 Criterio 4: Información para utilizar la herramienta

Nombre del criterio: información para utilizar la herramienta

Descripción: Puntuación en función de la cantidad y calidad de la información encontrada para poner en marcha los test.

Tipo de valor: Numérico (1-5).

## 3.5 Criterio 5: Implementación intuitiva

Nombre del criterio: implementación intuitiva de los tests

Descripción: facilidad para utilizar las funciones de las herramientas, resolver los fallos producidos e interpretar los resultados.

Tipo de valor: Numérico (1-5).

# 7. Comparación de la implementación de las tecnologías

|  |  |  |  |
| --- | --- | --- | --- |
| CRITERIOS | PHPUnit | Codeception | COMENTARIOS |
| C1 Dificultad instalación SW | 3 horas | 4 horas | Se ha tardado una hora más en Codeception, mayor complejidad. |
| C2 Recursos necesarios | 3 herramientas | 3 herramientas | Necesitamos la misma cantidad de herramientas- |
| C3 Tiempo de respuesta | 0,000 segundos | 0,007 segundos | En el código PHP creado se tarda más en realizarse las pruebas en Codeception, al ser un código tan pequeño los tiempos son mínimos, tanto es así, que en PHPUnit nos da 0,000 segundos. |
| C4 Información disponible | 2 | 1 | Se ha encontrado algo más de información en PHPUnit, pero aun así la cantidad de información encontrada ha sido muy reducida y difícil de encontrar. |
| C5 Implementación intuitiva | 2 | 1 | Como era más difícil encontrar la información en Codeception, ante un problema nos ha costado más resolverlo, y encima hemos tenido más problemas con Codeception para hacer funcionar sus tests. |

# 8. Conclusiones

Tal y como se explicó en el trabajo anterior PHPUnit y Codeception presentan funcionalidades similares.

Codeception es más aconsejable para aplicaciones grandes ya que la velocidad que ofrece para pruebas complejas es mayor. También ofrece una mayor usabilidad a la hora de entender los test de prueba. Sin embargo, el esfuerzo requerido para la puesta en marcha y la configuración es mucho mayor. Esto es debido a que tiene mayor complejidad, pero también a que hay mucha menos documentación disponible.

En este caso en concreto, la aplicación que se ha desarrollado es pequeña y los tests requeridos no implican una dificultad excesiva por lo que la mejor opción es PHPUnit.