

Máster Universitario en Software de Sistemas Distribuidos y Empotrados

Sistema para el Bienestar Emocional

PROYECTO FIN DE MÁSTER

Victor Manuel Domínguez Rivas

Julio de 2023

Máster Universitario en Software de Sistemas Distribuidos y Empotrados

Sistema para el Bienestar Emocional

PROYECTO FIN DE MÁSTER

Autor: Victor Manuel Domínguez Rivas

Directoras: Sandra Gómez Canaval, Gema Bello Orgaz

Julio de 2023

Tus agradecimientos aquí.

Índice

1. Introducción	2
1.1. Motivación	2
1.2. Objetivos	2
1.3. Metodología	2
1.4. Estructura del documento	2
2. Contexto	3
2.1. Estado del arte	3
2.2. Tecnologías utilizadas	3
3. Marco teórico	4
4. Especificación de requisitos	5
4.1. Introducción	5
4.1.1. Propósito	5
4.1.2. Alcance	5
4.2. Descripción	5
4.2.1. Perspectiva del Producto	5
4.2.2. Funciones del Producto	5
4.2.3. Características de los Usuarios	5
4.2.4. Restricciones	5
4.2.5. Suposiciones y Dependencias	5
4.2.6. Requisitos Futuros	5
4.3. Requisitos Específicos	6
4.3.1. Interfaces Externas	6
4.3.2. Funciones	6
4.3.3. Requisitos de Rendimiento	6
4.3.4. Restricciones de Diseño	6
4.3.5. Atributos del Sistema	6
4.3.6. Otros Requisitos	6
5. Diseño del sistema	7
6. Implementación del sistema	8
7. Pruebas del sistema	9
8. Resultados	10

9. Gestión del proyecto	11
9.1. Planificación	11
9.2. Seguimiento	11
9.3. Presupuesto	11
10. Conclusiones	12
10.1. Conclusiones técnicas	12
10.2. Conclusiones sociales	12
10.3. Balance del aprendizaje	12
10.4. Reflexión final	12
11. Líneas futuras	13
Bibliografía	14
A. Código fuente del proyecto	15
A.1. App	15
A.1.1. Demo.kt	15

Índice de tablas

Índice de figuras

Índice de código

A.1. Demo.kt	15
------------------------	----

Glosario

- **Android:** Sistema Operativo para dispositivos móviles basado en Linux. Está desarrollado por Google.

Resumen

Resumen aqui.

Palabras clave: Palabras clave

Abstract

Abstract here.

Keywords: Keywords

Capítulo 1

Introducción

1.1. Motivación

[1]

1.2. Objetivos

1.3. Metodología

1.4. Estructura del documento

Capítulo 2

Contexto

2.1. Estado del arte

2.2. Tecnologías utilizadas

Capítulo 3

Marco teórico

Capítulo 4

Especificación de requisitos

4.1. Introducción

4.1.1. Propósito

4.1.2. Alcance

4.2. Descripción

4.2.1. Perspectiva del Producto

4.2.2. Funciones del Producto

4.2.3. Características de los Usuarios

4.2.4. Restricciones

4.2.5. Suposiciones y Dependencias

4.2.6. Requisitos Futuros

4.3. Requisitos Específicos

4.3.1. Interfaces Externas

4.3.2. Funciones

4.3.3. Requisitos de Rendimiento

4.3.4. Restricciones de Diseño

4.3.5. Atributos del Sistema

4.3.6. Otros Requisitos

Capítulo 5

Diseño del sistema

Capítulo 6

Implementación del sistema

Capítulo 7

Pruebas del sistema

Capítulo 8

Resultados

Capítulo 9

Gestión del proyecto

9.1. Planificación

9.2. Seguimiento

9.3. Presupuesto

Capítulo 10

Conclusiones

10.1. Conclusiones técnicas

10.2. Conclusiones sociales

10.3. Balance del aprendizaje

10.4. Reflexión final

Capítulo 11

Lineas futuras

Bibliografía

- [1] V. M. Dominguez Rivas, *Generación de cuadros impresionistas mediante Redes Neuronales*. Madrid: Universidad Politécnica de Madrid, 2020.

Apéndice A

Código fuente del proyecto

A.1. App

A.1.1. Demo.kt

```
1  /* Block comment */
2  package hello
3  import kotlin.collections.* // line comment
4
5  /**
6   * Doc comment here for 'SomeClass'
7   * @see Iterator#next()
8   */
9  @Deprecated("Deprecated class")
10 private class MyClass<out T : Iterable<T>>(var prop1 : Int) {
11     fun foo(nullable : String?, r : Runnable, f : () -> Int,
12         fl : FunctionLike, dyn: dynamic) {
13         println("length\nis ${nullable?.length} \e")
14         val ints = java.util.ArrayList<Int?>(2)
15         ints[0] = 102 + f() + fl()
16         val myFun = { -> "" };
17         var ref = ints.size
18         ints.lastIndex + globalCounter
19         ints.forEach lit@ {
20             if (it == null) return@lit
21             println(it + ref)
22         }
23         dyn.dynamicCall()
24         dyn.dynamicProp = 5
25     }
26
27     val test = """hello
28                 world
29                 kotlin"""
30 }
```

```
31     override fun hashCode(): Int {
32         return super.hashCode() * 31
33     }
34 }
35
36 fun Int?.bar() {
37     if (this != null) {
38         println(message = toString())
39     }
40     else {
41         println(this.toString())
42     }
43 }
44
45 var globalCounter : Int = 5
46     get = field
47
48 abstract class Abstract
49
50 object Obj
51
52 enum class E { A, B }
53
54 interface FunctionLike {
55     operator fun invoke() = 1
56 }
57
58 open class Bar {
59     protected lateinit var x: String
60
61     tailrec infix fun thing(x: Int): Int {
62         // ...
63     }
64 }
65
66 inline fun <reified T: Any> doStuff() = T::class
67
68 const val NUMBER = 42
```

Código A.1: Demo.kt