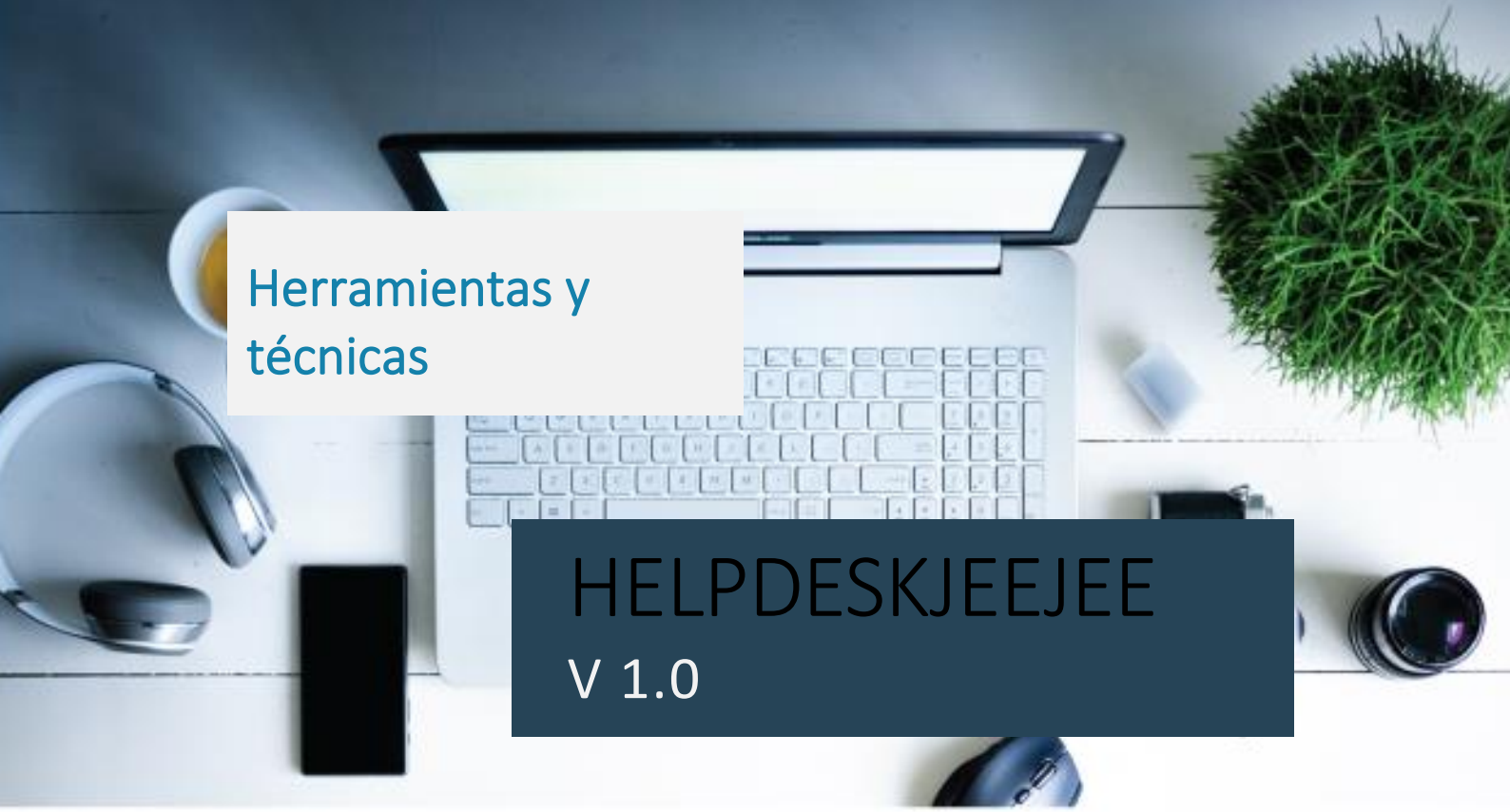


Herramientas y técnicas

HELPDESKJEEJEE

V 1.0

Desarrollo de un sistema Helpdesk para el registro y control de incidencias técnicas en el jurado electoral especial de Arequipa en elecciones congresales extraordinarias 2021.

Hoja de control

Organización	Jurado Electoral Especial de Arequipa Perú		
Proyecto	HELPDESKJEE		
Versión	1.0	**Fecha**	17/03/2021
Revisión		**Aprobación**	

Revisión

_____

Firma

Responsable: Jose Luis Caamal Ic

Aprobación

_____

Firma

Responsable: Roger Héctor Aranda Vega

Tabla de contenido

Herramientas y técnicas 1
Hoja de control 2
Revisión 2
Aprobación 2
1. Información introductoria 4
2. Metodología de desarrollo de software 5
2.1 Principios de desarrollo 5
2.2 Ciclo de vida 6
2.3 Roles 7
3 Consideraciones técnicas 8
3.1 Arquitectura basada en MVC 8
3.2 Programación orientada a objetos 9
3.2.1 Conceptos fundamentales 9
3.3 Lenguaje de programación php 11
3.3.1 Características 11
3.4 Utilerías 12
3.4.1 Visual Studio Code 12
3.4.2 Tortoise 12
3.4.3 Github 13
3.5 Base de datos 13
3.5.1 Tipos de compilación del servidor 14
4 Referencias 15

1. Información introductoria

Las herramientas y técnicas implementadas en el desarrollo del proyecto constituyen una guía para la construcción, seguimiento y actualización del sistema.

En el presente documento se describen las implementaciones del proyecto HELPDESKJEE, considerando que dichas herramientas fueron seleccionadas para el proyecto en el proyecto especifico que aborda este documento, se resguarda la responsabilidad de alteraciones que pudieran afectar el funcionamiento oprimo del sistema en el caso de implementarse tecnologías diferentes, miembros que deben ser adaptabas a la arquitectura del proyecto.

Secciones:

- ### Metodología de desarrollo de software

Describe el modelo implementado para la ejecución del proyecto, sus principales características, fases y roles.

- ### Consideraciones técnicas

Describe las herramientas técnicas utilizadas para el desarrollo del proyecto, desde su construcción e ambiente de implementación.

2. Metodología de desarrollo de software

El Proceso Racional Unificado o RUP (por sus siglas en inglés de Rational Unified Process) es un proceso de desarrollo de software desarrollado por la empresa Rational Software, actualmente propiedad de IBM (Pabón, 2021).

2.1 Principios de desarrollo

La Filosofía del RUP está basado en 6 principios clave que son los siguientes:

- Adaptar el proceso

El proceso deberá adaptarse a las necesidades del cliente ya que es muy importante interactuar con él. Las características propias del proyecto, el tamaño del mismo, así como su tipo o las regulaciones que lo condicionen, influirán en su diseño específico. También se deberá tener en cuenta el alcance del proyecto.

- Equilibrar prioridades

Los requisitos de los diversos participantes pueden ser diferentes, contradictorios o disputarse recursos limitados. Debe poder encontrarse un equilibrio que satisfaga los deseos de todos. Gracias a este equilibrio se podrán corregir desacuerdos que surjan en el futuro. Al igual esta metodología está acorde con UML, Unificado Modelo Lenguajes.

- Demostrar valor iterativamente

Los proyectos se entregan, aunque sea de un modo interno, en etapas iteradas. En cada iteración se analiza la opinión de los inversores, la estabilidad y calidad del producto, y se refina la dirección del proyecto así como también los riesgos involucrados.

- Colaboración entre equipos

El desarrollo de software no lo hace una única persona sino múltiples equipos. Debe haber una comunicación fluida para coordinar requisitos, desarrollo, evaluaciones, planes, resultados, etc.

- Enfocarse en la calidad

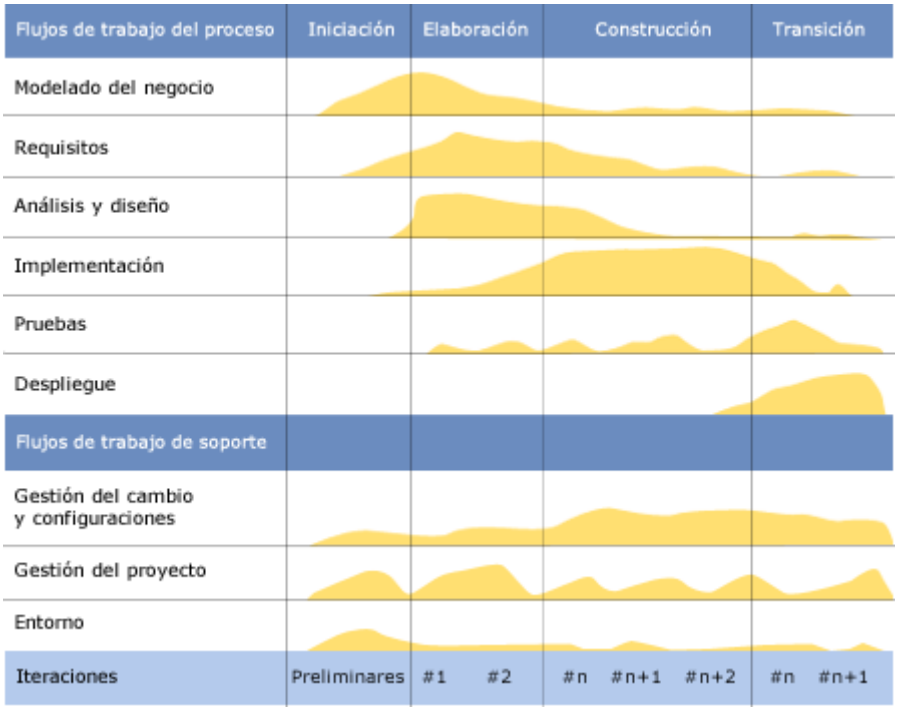
El control de calidad no debe realizarse al final de cada iteración, sino en todos los aspectos de la producción. El aseguramiento de la calidad forma parte del proceso de desarrollo y no de un grupo independiente, también es una estrategia de desarrollo de software.

- Elevar el nivel de abstracción

Este principio dominante motiva el uso de conceptos reutilizables tales como patrones de diseño del software, lenguajes 4GL o esquemas (frameworks) por nombrar algunos. Estos se pueden acompañar por las representaciones visuales de la arquitectura, por ejemplo con UML.

2.2 Ciclo de vida

El ciclo de vida RUP es una implementación del desarrollo en espiral. Fue creado ensamblando los elementos en secuencias semi-ordenadas. El ciclo de vida organiza las tareas en fases e iteraciones. RUP divide el proceso en cuatro fases, dentro de las cuales se realizan pocas pero grandes y formales iteraciones en número variable según el proyecto. En la Figura muestra cómo varía el esfuerzo asociado a las disciplinas según la fase en la que se encuentre el proyecto RUP.



Las primeras iteraciones (en las fases de Inicio y Elaboración) se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y al establecimiento de una baseline (línea base)2 de la arquitectura. Durante la fase de inicio las iteraciones hacen mayor énfasis en actividades de modelado del negocio y de requisitos.

En la fase de elaboración, las iteraciones se orientan al desarrollo de la baseline de la arquitectura, abarcan más los flujos de trabajo de requisitos, modelo de negocios (refinamiento), análisis, diseño y una parte de implementación orientado a la baseline de la arquitectura.

En la fase de construcción, se lleva a cabo la construcción del producto por medio de una serie de iteraciones.

Para cada iteración se seleccionan algunos Casos de Uso, se refinan su análisis y diseño y se procede a su implementación y pruebas. Se realiza una pequeña cascada para cada ciclo. Se realizan iteraciones hasta que se termine la implementación de la nueva versión del producto.

En la fase de transición se pretende garantizar que se tiene un producto preparado para su entrega a la comunidad de usuarios.

2.3 Roles

Un rol define el comportamiento y responsabilidades de un individuo, o de un grupo de individuos trabajando juntos como un equipo. Una persona puede desempeñar diversos roles, así como un mismo rol puede ser representado por varias personas.

Las responsabilidades de un rol son tanto el llevar a cabo un conjunto de actividades como el ser el dueño de un conjunto de artefactos.

A continuación se definen los roles en el proyecto ***HELPDESKJEE*** mediante la siguiente matriz RACI donde:

R – Responsable

A – aprobador

I – Informado

C – consultor

Actividades		**Participantes**				
RUP	**Fase**	**Stakeholder**	**Líder de proyecto**	**Analista**	**Dev1**	**Dev2**
Iniciación	Modelado de negocio	RA	RA	I	I	I
Elaboración	Requisitos	A	R	IC	I	I
	Análisis y diseño	A	C	R	I	I
Construcción	Implementación	A	C	I	R	R
	Pruebas	A	R	I	R	R
Transición	Despliegue	A	R	I	IC	IC

3 Consideraciones técnicas

HELPDESKJEE es una solución diseñada para ser operada mediante un componente web, basado en la arquitectura cliente-servidor.

Se implementan las siguientes herramientas de software para el desarrollo:

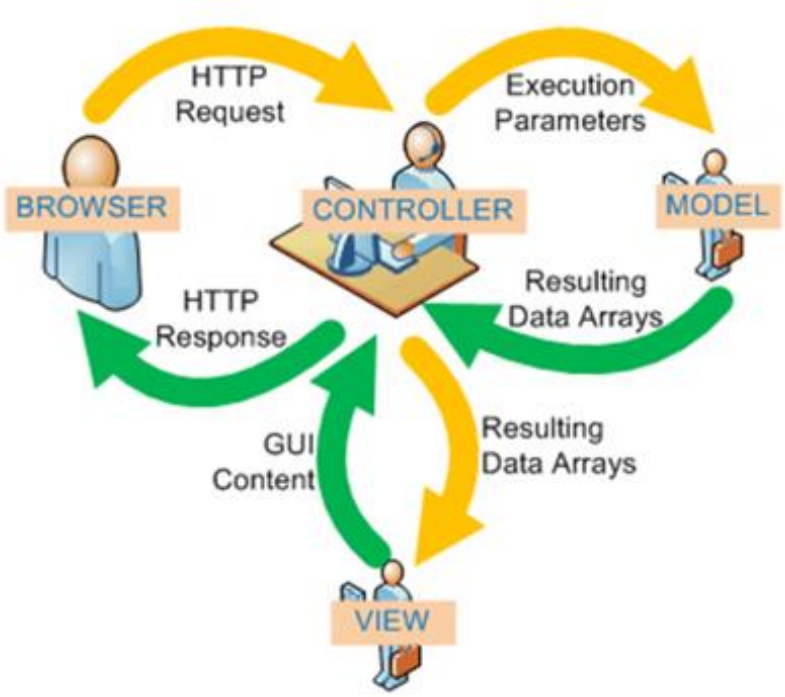
3.1 Arquitectura basada en MVC

Modelo Vista Controlador (MVC) es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos ("Modelo vista controlador (MVC). Servicio de Informática ASP.NET MVC 3 Framework", 2021).

Se trata de un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones, y sobre multitud de lenguajes y plataformas de desarrollo.

- El Modelo que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.
- La Vista, o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos interacción con éste.
- El Controlador, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.

El flujo que sigue el control generalmente es el siguiente:



3.2 Programación orientada a objetos

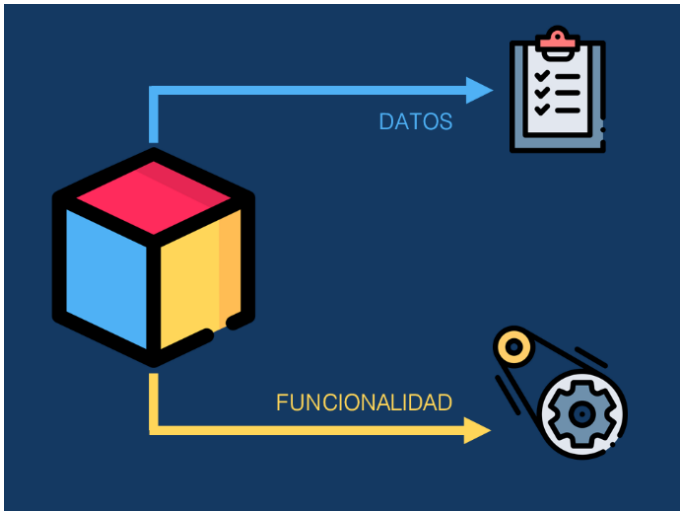
La Programación Orientada a Objetos (POO, en español; OOP, según sus siglas en inglés) es un paradigma de programación que viene a innovar la forma de obtener resultados. Los objetos se utilizan como metáfora para emular las entidades reales del negocio a modelar ("Qué es la programacion orientada a objetos", 2021).

Está basada en varias técnicas del sexenio: herencia, cohesión, abstracción, polimorfismo, acoplamiento y encapsulamiento.

Los objetos son entidades que tienen un determinado "estado", "comportamiento (método)" e "identidad":

La identidad es una propiedad de un objeto que lo diferencia del resto; dicho con otras palabras, es su identificador (concepto análogo al de identificador de una variable o una constante).

Los métodos (comportamiento) y atributos (estado) están estrechamente relacionados por la propiedad de conjunto. Esta propiedad destaca que una clase requiere de métodos para poder tratar los atributos con los que cuenta.



3.2.1 Conceptos fundamentales

La POO es una forma de programar que trata de encontrar una solución a estos problemas. Introduce nuevos conceptos, que superan y amplían conceptos antiguos ya conocidos. Entre ellos destacan los siguientes:

- **Clase**

Una clase es una especie de "plantilla" en la que se definen los atributos y métodos predeterminados de un tipo de objeto. Esta plantilla se crea para poder crear objetos fácilmente. Al método de crear nuevos objetos mediante la lectura y recuperación de los atributos y métodos de una clase se le conoce como instanciación.

- **Herencia**

Por ejemplo, herencia de la clase C a la clase D, es la facilidad mediante la cual la clase D hereda en ella cada uno de los atributos y operaciones de C, como si esos atributos y operaciones hubiesen sido definidos por la misma D. Por lo tanto, puede usar los mismos métodos y variables registrados como "públicos" (public) en C. Los componentes registrados como "privados" (private) también se heredan pero se mantienen escondidos al programador y solo pueden ser accedidos a través de otros métodos públicos. Para poder acceder a un atributo u operación de una clase en cualquiera de sus subclases pero mantenerla oculta para otras clases es necesario registrar los componentes como "protegidos" (protected), de esta manera serán visibles en C y en D pero no en otras clases.

- **Objeto**

Instancia de una clase. Entidad provista de un conjunto de propiedades o atributos (datos) y de comportamiento o funcionalidad (métodos), los mismos que consecuentemente reaccionan a eventos. Se corresponden con los objetos reales del mundo que nos rodea, o con objetos internos del sistema (del programa).

- **Método**

Algoritmo asociado a un objeto (o a una clase de objetos), cuya ejecución se desencadena tras la recepción de un "mensaje". Desde el punto de vista del comportamiento, es lo que el objeto puede hacer. Un método puede producir un cambio en las propiedades del objeto, o la generación de un "evento" con un nuevo mensaje para otro objeto del sistema.

- **Evento**

Es un suceso en el sistema. El sistema maneja el evento enviando el mensaje adecuado al objeto pertinente. También se puede definir como evento la reacción que puede desencadenar un objeto; es decir, la acción que genera.

- **Atributos**

Características que tiene la clase.

- **Mensaje**

Una comunicación dirigida a un objeto, que le ordena que ejecute uno de sus métodos con ciertos parámetros asociados al evento que lo generó.

- **Propiedad o atributo**

Contenedor de un tipo de datos asociados a un objeto (o a una clase de objetos), que hace los datos visibles desde fuera del objeto y esto se define como sus características predeterminadas, y cuyo valor puede ser alterado por la ejecución de algún método.

- **Estado interno**

Es una variable que se declara privada, que puede ser únicamente accedida y alterada por un método del objeto, y que se utiliza para indicar distintas situaciones posibles para el objeto (o clase de objetos). No es visible al programador que maneja una instancia de la clase.

- **Miembros de un objeto**

Atributos, identidad, relaciones y métodos.

- **Identificación de un objeto**

Un objeto se representa por medio de una tabla o entidad que esté compuesta por sus atributos y funciones correspondientes.

3.3 Lenguaje de programación php

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML ("PHP: Hypertext Preprocessor", 2021).

El código PHP es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá el resultado de ejecutar el script, aunque no se sabrá el código subyacente que era. El servidor web puede ser configurado incluso para que procese todos los ficheros HTML con PHP.

PHP puede combinarse con MySQL para trabajar con bases de datos, aunque también se pueden utilizar otros motores de base de datos como Microsoft SQL Server, PostgreSQL, MongoDB, entre otros.

3.3.1 Características

- Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- Es considerado un lenguaje fácil de aprender, ya que en su desarrollo se simplificaron distintas especificaciones, como es el caso de la definición de las variables primitivas, ejemplo que se hace evidente en el uso de php arrays.
- El código fuente escrito en PHP es invisible al navegador web y al cliente, ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador.
- Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- Capacidad de expandir su potencial utilizando módulos (llamados extensiones).
- Posee una amplia documentación en su sitio web oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- Tiene manejo de excepciones (desde PHP5).

3.4 Utilerías

Una utilidad es una herramienta que sirve de soporte para la construcción y ejecución de programas, en donde se incluyen las bibliotecas de sistema, middleware, herramientas de desarrollo, etc.

A continuaciónón se listan las utilerías implementadas en el desarrollo de ***HELPDESKJEE:***

3.4.1 Visual Studio Code

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos, refactorización de código y personalizable ("Visual Studio Code - Code Editing. Redefined", 2021).

Visual Studio Code se puede extender a través de complementos, disponible a través de un repositorio central. Esto incluye adiciones al editor y soporte de lenguajes.1 Una característica notable es la capacidad de crear extensiones que analizan código, como linters y herramientas para análisis estático, utilizando el Protocolo de Servidor de Idioma.

3.4.2 Tortoise

TortoiseGit es una interfaz de shell de Windows para Git y se basa en TortoiseSVN. Es de código abierto y se puede construir completamente con software disponible gratuitamente.
TortoiseGit facilita tareas regulares, como confirmar, mostrar registros, diferenciar dos versiones, crear ramas y etiquetas, crear parches, etc. ("TortoiseGit – Windows Shell Interface to Git", 2021)

Características de TortoiseGit

- Fácil uso
- Potente diálogo de confirmación (ver capturas de pantalla)
- Por configuración de proyecto
- Integración con sistemas de seguimiento de problemas
- Herramientas útiles
- Disponible en variedad de idiomas
- Estabilidad

3.4.3 Github

GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador ("GitHub: Where the world builds software", 2021).

Características:

- Wiki para cada proyecto
- Página web para cada proyecto
- Gráfico para ver cómo los desarrolladores trabajan en sus repositorios y bifurcaciones del proyecto
- Funcionalidades como si se tratase de una red social
- Herramienta para trabajo colaborativo entre programadores
- Gestor de proyectos de estilo Kanban
- Actions herramientas de CI
- Codespaces un IDE en la nube para los repositorios

3.5 Base de datos

MySQL es un sistema de gestión de bases de datos relacional, ("MySQL", 2021).
Cuenta con una doble licencia. Por una parte es de código abierto, pero por otra, cuenta con una versión comercial gestionada por la compañía Oracle.

Entre las características disponibles en las últimas versiones se puede destacar:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente
- Disponibilidad en gran cantidad de plataformas y sistemas
- Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferentes velocidades de operación, soporte físico, capacidad, distribución geográfica, transacciones etc.
- Transacciones y claves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda e indexación de campos de texto.

3.5.1 Tipos de compilación del servidor

Hay tres tipos de compilación del servidor MySQL:

- Estándar: Los binarios estándar de MySQL son los recomendados para la mayoría de los usuarios, e incluyen el motor de almacenamiento InnoDB.
- Max (No se trata de MaxDB, que es una cooperación con SAP): Los binarios incluyen características adicionales que no han sido lo bastante probadas o que normalmente no son necesarias.
- MySQL-Debug: Son binarios que han sido compilados con información de depuración extra. No debe ser usada en sistemas en producción porque el código de depuración puede reducir el rendimiento.

4 Referencias

GitHub: Where the world builds software. (2021). Retrieved 17 March 2021, from https://github.com/

MySQL. (2021). Retrieved 17 March 2021, from https://www.mysql.com/

Modelo vista controlador (MVC). Servicio de Informática ASP.NET MVC 3 Framework. (2021). Retrieved 17 March 2021, from https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html

PHP: Hypertext Preprocessor. (2021). Retrieved 17 March 2021, from https://www.php.net/index.php#id2021-03-04-2

Pabón, I. (2021). SMARTSOFT - Metodología de Desarrollo Tradicional RUP. Retrieved 17 March 2021, from https://smartsoftcolombia.com/

Qué es la programación orientada a objetos. (2021). Retrieved 17 March 2021, from https://desarrolloweb.com/articulos

TortoiseGit – Windows Shell Interface to Git. (2021). Retrieved 17 March 2021, from https://tortoisegit.org/

Visual Studio Code - Code Editing. Redefined. (2021). Retrieved 17 March 2021, from https://code.visualstudio.com/