Desarrollo de una aplicación web para la gestión del seguimiento del cumplimiento de tareas de los docentes de la CARRERA DE INGENIERÍA DE SISTEMAS DE LA UNIVERSIDAD POLITÉCNICA SALESIANA

Una Tesis Presentada Para Obtener El Título De

Ingeniero de Sistemas

Universidad Politécnica Salesiana, Quito

Pablo Efrén Guerra Mora

Julio 2017

**Dedicatoria**

ESTA PÁGINA ES OPCIONAL.

Dedicamos esta plantilla a los usuarios de nuestros sitios: normasapa.com y normasieee.com.**Agradecimientos**

Gracias por su preferencia de normasapa.com, no olviden recomendarnos con sus colegas y compañeros. ESTA PAGINA ES OPCIONAL

**Abstract**

Con el crecimiento de la carrera de ingeniería de sistemas de la Universidad Politécnica Salesiana han surgido necesidades de mejora de procesos que a su vez llevarán a la carrera y a sus directivos a tener información más exacta de cómo está funcionando la carrera y cómo se están cumpliendo todos los integrantes de la carrera las responsabilidades que les corresponde. La forma actual como se evalúa las responsabilidades de los docentes con la carrera como cronogramas, reuniones, evaluaciones, etc. se la realiza de una forma manual; subiendo archivos a repositorios virtuales o físicos y calificando estas tareas en una hoja digital. Es por eso que luego de un análisis y evaluación del proceso se ha llevado a cabo el desarrollo de un sistema de evaluación que está especialmente diseñado para los fines que la carrera necesita para evaluar a sus docentes. Este sistema permite la creación de tareas de forma automática así como la evaluación de estas tareas por parte de los jefes de área de la carrera. La forma de evaluación no es solo cuantitativa sino cualitativa por lo que existirá una perspectiva completa del rendimiento de los docentes y por lo tanto de la carrera para así encontrar puntos de mejora para que los resultados y procesos de la carrera estén siempre optimizados.

# Índice

[Índice v](#_Toc486936512)

[1 Introducción e información general 1](#_Toc486936513)

[Justificación del tema 1](#_Toc486936514)

[Objetivos 2](#_Toc486936515)

[Objetivo General 2](#_Toc486936516)

[Objetivos Específicos 2](#_Toc486936517)

[Marco metodológico 2](#_Toc486936518)

[Modelo vista controlador 2](#_Toc486936519)

[Desarrollo ágil 3](#_Toc486936520)

[SCRUM 3](#_Toc486936521)

[Taiga 4](#_Toc486936522)

[2 Marco teórico 5](#_Toc486936523)

[Base de datos 5](#_Toc486936524)

[MySQL 5](#_Toc486936525)

[Frameworks 5](#_Toc486936526)

[Spring 5](#_Toc486936527)

[Angular JS 6](#_Toc486936528)

[Hibernate 7](#_Toc486936529)

[Angular-material 7](#_Toc486936530)

[Ambientes de desarrollo 8](#_Toc486936531)

[JBoss 8](#_Toc486936532)

[Node.js 8](#_Toc486936533)

[Herramientas Javascript 9](#_Toc486936534)

[Gulp 9](#_Toc486936535)

[Babel 9](#_Toc486936536)

[BrowserSync 9](#_Toc486936537)

[Buenas prácticas 10](#_Toc486936538)

[Git 10](#_Toc486936539)

[3 Análisis y diseño 10](#_Toc486936540)

[Análisis del problema 10](#_Toc486936541)

[Recolección de información 10](#_Toc486936542)

[Requisitos 11](#_Toc486936543)

[UML 13](#_Toc486936544)

[Casos de uso 14](#_Toc486936545)

[Caso de uso principal 14](#_Toc486936546)

[Diagramas de clases 17](#_Toc486936547)

[Controladores 17](#_Toc486936548)

[Interfaz de usuario 21](#_Toc486936549)

[Angular Material 21](#_Toc486936550)

[4 Construcción y pruebas 21](#_Toc486936551)

[Modelo vista controlador 21](#_Toc486936552)

[Base de datos 22](#_Toc486936553)

[Diagrama de base de datos 24](#_Toc486936554)

[Arquitectura de servicios Web 25](#_Toc486936555)

[Casos de uso 26](#_Toc486936556)

[Administración de tareas 26](#_Toc486936557)

[Registro de Reuniones 30](#_Toc486936558)

[Envío de tareas 35](#_Toc486936559)

[Diagramas de clases 39](#_Toc486936560)

[Controladores 39](#_Toc486936561)

[Servicios 40](#_Toc486936562)

[Gestores 41](#_Toc486936563)

[Módulos 42](#_Toc486936564)

[Criterios 43](#_Toc486936565)

[Tareas 44](#_Toc486936566)

[Usuarios 45](#_Toc486936567)

[Utilitarios 46](#_Toc486936568)

[Interfaces de usuario 47](#_Toc486936569)

[Login 47](#_Toc486936570)

[Pantalla de inicio 48](#_Toc486936571)

[Categorías de tareas 49](#_Toc486936572)

[Lista de tareas 49](#_Toc486936573)

[Enviar tarea 50](#_Toc486936574)

[Calificar tarea 51](#_Toc486936575)

[Crear tarea 52](#_Toc486936576)

[Reporte de notas 53](#_Toc486936577)

[Reportes de docentes 54](#_Toc486936578)

[Desafíos 54](#_Toc486936579)

[Plan de pruebas 55](#_Toc486936580)

[Objetivo 55](#_Toc486936581)

[Alcance de las pruebas 55](#_Toc486936582)

[Enfoque de las pruebas 55](#_Toc486936583)

[Criterios de aceptación 56](#_Toc486936584)

[Entregables 56](#_Toc486936585)

[Recursos 56](#_Toc486936586)

[Ejecución de pruebas 57](#_Toc486936587)

[Pruebas de usabilidad 57](#_Toc486936588)

[Pruebas de rendimiento 59](#_Toc486936589)

[Conclusiones y Recomendaciones 61](#_Toc486936590)

[Conclusiones 61](#_Toc486936591)

[Lista de referencias 63](#_Toc486936592)

**Índice de tablas**

[Tabla 2. Escenario ingreso al sistema 16](#_Toc471991281)

[Tabla 3. Creación de tarea 16](#_Toc471991282)

[Tabla 4. Carga de tarea al sistema 16](#_Toc471991283)

[Tabla 5. Revisión y calificación de tarea 17](#_Toc471991284)

[Tabla 6. Visualización de nota en el sistema 17](#_Toc471991285)

[Tabla 7. Ingreso al sistema 27](#_Toc471991286)

[Tabla 9. Creación de tarea 27](#_Toc471991287)

[Tabla 10. Revisión de tareas 28](#_Toc471991288)

[Tabla 11. Calificación de tareas 29](#_Toc471991289)

[Tabla 12. Ingreso al sistema 31](#_Toc471991290)

[Tabla 13. Ingreso al módulo de tareas 31](#_Toc471991291)

[Tabla 14. Selección tipo de tarea 33](#_Toc471991292)

[Tabla 15. Ingreso detalle reunión 33](#_Toc471991293)

[Tabla 16. Registro de reunión 34](#_Toc471991294)

[Tabla 17. Ingreso al sistema 36](#_Toc471991295)

[Tabla 18. Ingreso al módulo de tareas 36](#_Toc471991296)

[Tabla 19. Selección de tarea 37](#_Toc471991297)

[Tabla 20. Envío de tarea 38](#_Toc471991298)

**Índice de figuras**

[Figura 1. Caso de uso principal 15](#_Toc471991299)

[Figura 3. Diagrama de clases controladores 18](#_Toc471991300)

[Figura 4. Diagrama de clases Utilitarios 19](#_Toc471991301)

[Figura 5. Diagrama de clases entidades 20](#_Toc471991302)

[Figura 7. Diagrama de clases entidades 21](#_Toc471991303)

[Figura 3. Diagrama de base de datos 24](#_Toc471991304)

[Figura 7. Arquitectura de servicios web 25](#_Toc471991305)

[Figura 8. Caso de uso administración de tareas. 26](#_Toc471991306)

[Figura 9. Registro de reuniones 30](#_Toc471991307)

[Figura 10. Caso de uso envío de tareas 35](#_Toc471991308)

[Figura 11. Diagrama de clases controladores 39](#_Toc471991309)

[Figura 12. Diagrama de clases Servicios 40](#_Toc471991310)

[Figura 13. Diagrama de clases gestores 41](#_Toc471991311)

[Figura 14. Diagrama de clases módulos 42](#_Toc471991312)

[Figura 15. Diagrama de clases criterios 43](#_Toc471991313)

[Figura 14. Diagrama de clases tareas 44](#_Toc471991314)

[Figura 15. Diagrama de clases usuarios 45](#_Toc471991315)

[Figura 16. Diagrama de clases utilitarios 46](#_Toc471991316)

[Figura 18. Interfaz de usuario – login 47](#_Toc471991317)

# 1 Introducción e información general

## Justificación del tema

En las materias de la Universidad Politécnica Salesiana, existen profesores que tienen como rol la docencia, otros, además de dictar materias su rol es el de organizar, coordinar y mantener la carrera funcionando de modo que todo lo que suceda en cada semestre esté perfectamente alineado a los objetivos de la universidad. Este rol lo tienen los jefes de área, que liderados por el director de carrera tienen la potestad de evaluar y diagnosticar a los docentes que dicten materias de la carrera. Lo que se evalúa en cada docente son los objetivos que deben cumplir cada semestre, estas son tareas que por lo general son documentos que deben ser entregados hasta una fecha determinada y contienen información sobre el avance de los objetivos de cada semestre.

En el período de tiempo en que la carrera ha existido se han utilizado diferentes medios para evaluar las tareas y el factor común de éstos ha sido la carga de tareas en un medio físico o uno en la nube para luego ser calificados por los jefes de área del mismo modo.

Aunque la utilidad para la evaluación a los docentes de la carrera es provechosa estos medios no fueron creados para este fin y no poseen todas las características necesarias para que la evaluación sea lo más eficiente y eficaz por lo que un sistema especialmente diseñado para la evaluación de docentes es lo más apropiado.

## Objetivos

### Objetivo General

Mejorar el proceso de evaluación de las tareas de los docentes de la carrera de ingeniería de sistemas de la Universidad Politécnica Salesiana.

### Objetivos Específicos

* Desarrollar un sistema Web que facilite el seguimiento de las tareas de los docentes con parámetros configurables de acuerdo a la situación de cada área de trabajo.
* Investigar y definir e
* .0l modelo de gestión de seguimiento de tareas utilizado actualmente
* Analizar y diseñar la solución (el nuevo sistema)
* Construir el software y probar su funcionalidad.
* Incrementar el número de tareas entregadas en comparación con el  sistema actual

## Marco metodológico

### Modelo vista controlador

El modelo vista controlador organiza una aplicación de manera que los datos, la lógica y la interfaz de usuario están separados y conectados por un módulo principal, en la aplicación se utilizó debido a la necesidad de dividir en capas las partes de la aplicación para una mejor mantenibilidad y perspectiva a futuro.

### Desarrollo ágil

Los proyectos que utilizan metodologías de desarrollo ágil difieren de las metodologías tradicionales las cuales están basadas en un desarrollo secuencial en que su desarrollo es incremental e iterativo, es decir el proyecto avanza en iteraciones y las funcionalidades pueden ir creciendo poco a poco según los requerimientos y el tiempo disponible. Las metodologías no ágiles o tradicionales tienen un plan definido que se debe seguir y se tiene un producto completo al final del plan. El objetivo de comprar metodologías ágiles con tradicionales no tiene como propósito concluir que el desarrollo ágil es mejor que el secuencial, ambas son herramientas de desarrolladores y existen circunstancias apropiadas para cada una (Kenneth, S. Rubin, 2012, pág. 29).

El uso de una metodología ágil en este proyecto fue resuelto debido a que el equipo de trabajo es pequeño (1 persona) y el tiempo de desarrollo es corto (menos de 1 año), además la metodología escogida es muy útil para organizar el tiempo y tener un producto presentable en un período de tiempo corto, así se puede tener retroalimentación de las funcionalidades del software y realizar mejoras para al final tener un producto de calidad y que cumpla con los objetivos establecidos.

### SCRUM

Es una metodología ágil, utilizada generalmente en proyectos cortos (menos de un año) conformados por equipos pequeños o medianos, se caracteriza por organizar y manejar el trabajo de los integrantes del proyecto así como el tiempo en el que se desarrollarán las iteraciones. Las prácticas que definen a SCRUM lo hacen flexible y permiten que cada proyecto que lo utilice tenga un enfoque diferente de la metodología. Está basado en los valores de la honestidad, franqueza, coraje, respeto, enfoque, confianza, y colaboración (Kenneth, S. Rubin, 2012, pág. 13). Las características con las que SCRUM organiza los elementos de un proyecto están alineadas al producto que se desarrolló y es por eso que se decidió usar esta metodología.

### Taiga

Taiga es una herramienta de gestión de proyectos SCRUM que soporta desde proyectos simples a complejos, tiene un enfoque de código abierto y es una alternativa a desarrolladores que están utilizando una herramienta de desarrollo por primera vez.

# 2 Marco teórico

## Base de datos

### MySQL

Uno de los motores de bases de datos más populares del mundo es MySQL en parte porque tiene una naturaleza open source por lo que cualquiera puede usarlo gratuitamente, además está incluido en los repositorios de software de la mayoría de plataformas.

La elección de MySQL sobre otros motores de base de datos se dio debido a que por su gran popularidad tienen una gran comunidad que continuamente está dando solución a problemas y bugs que presenta la plataforma, además que MySQL posee ciertas ventajas sobre otras plataformas (Tabla 1)

## Frameworks

### Spring

En el desarrollo de aplicaciones de software la complejidad es un problema que causa que una aplicación se vuelva inmantenible con el tiempo, Spring es un framework que permite que el desarrollo de aplicaciones complejas sea manejable, escalable, mantenible y estructurado, todo esto es posible debido a que incluye características como:

* Implementa el modelo vista controlador
* Inyección de dependencias
* Soporte nativo de JDBC, JPA, JMS
* Incluye librerías como hibernate, maven, etc.
* Soporte para proyectos web

Spring establece un entorno de desarrollo dentro del cual tareas comunes que llevarían tiempo en implementar como compilación, pruebas, inyección de dependencias, conexión con bases de datos, etc. ya se encuentran incluidas en el núcleo por lo que lo único que debe hacer el desarrollador es configurar y comenzar a tomar ventaja de esas características.

### Angular JS

Angular JS es un framework estructural para aplicaciones web dinámicas. Utiliza HTML como lenguaje de interfaz y permite extender la sintaxis de HTML para expresar los componentes de la aplicación clara y sucintamente (What is Angular? 2010-2017). Tiene características que lo hacen un framework potente y liviano como inyección de dependencias, que con el uso de javascript permite añadir y usar librerías ajenas a Angular fácilmente. Angular presenta una solución frontend completa, ya que puede manipular del el DOM y manejar todas las operaciones CRUD (Crear, leer, actualizar y eliminar) sin la necesidad de otras librerías. Una de las características que hacen que Angular JS sea un framework “superheroico” es la posibilidad de crear aplicaciones SPA (single page applications) que no siempre recargan ni cambian de página, en cambio obtiene secciones HTML las añade y las renderiza sin tener que recargar la página.

### Hibernate

El mapeo de bases de datos mediante el patrón objeto/relación en aplicaciones Java se lo realiza mediante archivos de configuración y clases. Hibernate es un framework que facilita el mapeo de las tablas representándolas como clases y por lo tanto objetos Java, estas clases son llamadas persistentes ya que en un proceso de la aplicación los datos obtenidos de la base son creados como objetos los cuales pueden ser manipulados en el proceso y devueltos a la base. Hibernate es el encargado de estas tareas, y presenta una solución confiable, eficiente, flexible y escalable

### Angular-material

Material es un framework de interfaz de usuario creado por Google que comprende componentes basados en objetos materiales como luces, sombras y similares. Su interacción con el usuario ofrece una experiencia limpia, agradable a la vista y simple. Es una alternativa poderosa que no solo ofrece interfaz de usuario sino que contribuye a la lógica de la aplicación como tal ofreciendo características como:

* Ruteo
* Testing
* Conexión entre el modelo y la vista en el patrón MVC
* Servicios y Factories que encapsulan funcionalidad

Además trabaja en conjunto con Angular JS utilizando directivas y dotando a la aplicación de la posibilidad de contener lógica dentro del DOM.

## Ambientes de desarrollo

### JBoss

Es un servidor de aplicaciones parte de la colección de productos Red Hat que permite ejecutar archivos compilados Java, esto mediante la creación de servicios que en un punto determinado serán requeridos por la aplicación como conexión a base de datos, manejador de transacciones, servicios de seguridad, etc. todos estos servicios son transparentes para el usuario que solo necesitará de archivos XML de configuración para atar los servicios a los componentes de la aplicación sin necesidad de programar. Esto hace de JBoss una eficaz herramienta ahorrando recursos y reduciendo el tiempo de integración y configuración que se requeriría para levantar una aplicación Java desde cero.

### Node.js

Es un software que ejecuta programas escritos en lenguaje javascript, está diseñado para el desarrollo de aplicaciones escalables, fue necesario en la construcción de ésta aplicación ya que para la ejecución del sistema frontend en Angular.js se necesitaba un servidor de aplicaciones javascript y Node.js es el sistema que cumple las necesidades de la aplicación y se acopla muy bien con Angular.js

## Herramientas Javascript

### Gulp

En el desarrollo de aplicaciones web existen tareas que son comunes y que son requeridas muchas veces en el proceso de desarrollo, compilación y ejecución de la aplicación como la minificación, compilación y validación de código, levantamiento y recarga del servidor de aplicaciones y otras que pueden ser requeridas según las circunstancias del desarrollador. Babel es un conjunto de herramientas que automatiza tareas y reduce el tiempo de desarrollo de una aplicación. Se configura desde archivo javascript que contiene todas las tareas y el momento en que se van a ejecutar.

### Babel

Con el avance de los lenguajes de programación, ofreciendo cada vez más versatilidad y específicamente la estandarización en un futuro de EcmaScript6 Babel permite adelantarse a la estandarización y escribir código en EcmaScript 6 y compilarlo a la versión estándar actual de EcmaScript (5) para que pueda ser usado en cualquiera de los navegadores modernos que aún no incluyen soporte de todas las características de EcmaScript 6.

### BrowserSync

Es una herramienta que ayuda en el proceso de pruebas de una aplicación, esto mediante la recarga automática de páginas cuando detecta que ha existido algún cambio en el

código; así el desarrollador no desperdicia tiempo y puede realizar pruebas a su software de una manera más eficiente.

## Buenas prácticas

### Git

Es una herramienta de versionamiento creada por Linus Torvalds en 2005, se usa principalmente para que en los equipos de trabajo no existan problemas ni confusión cuando varios desarrolladores suban o modifiquen partes del mismo código.

En este proyecto se decidió utilizar Git debido a su potente motor de versionamiento desde el cual se pueden subir cambios al repositorio, clonar proyectos, hacer comparaciones de código, revertir cambios y respaldar el código usando únicamente comandos.

# 3 Análisis y diseño

## Análisis del problema

### Recolección de información

Se tuvo una reunión con los docentes miembros del consejo de carrera de sistemas y se obtuvo detalles de cuál es el proceso actual de evaluación de docentes además se envió una encuesta a los docentes de la carrera para obtener más información. (Anexo 1).

Con los datos recolectados se obtuvo un panorama general y específico del proceso con lo que se pudo hacer un análisis de las fortalezas y debilidades del sistema actual y cómo se podría mejorar el proceso.

### Requisitos

#### Requisitos de usuarios

Verbalmente, en una reunión con el directorio de la carrera se definieron los siguientes requisitos del sistema:

* El sistema permitirá la evaluación de tareas de docentes por parte de los jefes de área.
* Las tareas serán guardadas en una base de datos
* Existirán tareas comunes para todos los semestres
* Se podrán asignar varios docentes a una tarea
* Existirán dos tipos de tareas:
  + Tareas: deberán contener un archivo adjunto y un comentario
  + Reuniones: contendrán los detalles de la reunión y podrán contener o no un archivo adjunto
* Los criterios de evaluación de las tareas serán:
  + Cumplimiento de la tarea
  + Calidad de la tarea
  + Tiempo de entrega

#### Requisitos del sistema

Los requisitos para que el sistema esté funcionando en un ambiente de producción son:

* Un servidor java
* Un servidor javascript (*Node.js*)
* Una base de datos MySQL

#### Requisitos funcionales

Las funcionalidades principales que deberá cumplir el sistema para cumplir sus objetivos son:

* Los usuarios se pueden dividir según sus roles.
* Debe dar acceso a diferentes partes del sistema según los roles.
* Los usuarios podrán ingresar al sistema con credenciales creadas por el administrador del sistema.
* El jefe de área debe ser capaz de crear tareas y asignar las tareas a los docentes.
* El jefe de área puede crear reuniones y cancelarlas.
* En la creación y envío de tareas se pueden subir archivos.

#### Requisitos no funcionales

Los requisitos del sistema no funcionales tratan de cumplir las buenas prácticas de desarrollo de software en temas de usabilidad y facilidad de uso, así mismo, tienen como objetivo presentar un sistema de la mejor calidad posible que cumpla con las expectativas de los docentes, jefes de área y la carrera en general para mejorar el proceso de evaluación:

* Flexibilidad
* Escalabilidad
* Bajo impacto en el traslado desde el sistema anterior al actual
* Seguridad
* Mantenibilidad

## UML

El lenguaje unificado de modelamiento (Unified Modeling Language) es un lenguaje de modelamiento utilizado para especificar, visualizar, construir y documentar los artefactos de un sistema de software. (The Unified Modeling Language 2005). Sirve para representar conceptos intangibles como decisiones, configuraciones, flujos, etc. de sistemas de software que ya existen o van a ser construidos. Se utiliza principalmente para planificar y entender cómo va a funcionar un sistema, contiene varios modelos que pueden representar diferentes conceptos como diagramas de flujo, diagramas de casos de uso, diagramas de clases, diagramas de bases de datos.

UML es un recurso muy valioso para la construcción de software ya que los programadores y arquitectos de software pueden expresar sus ideas y cada participante de la construcción puede entender claramente lo que el diseñador quiere y desea obtener.

## Casos de uso

### Caso de uso principal

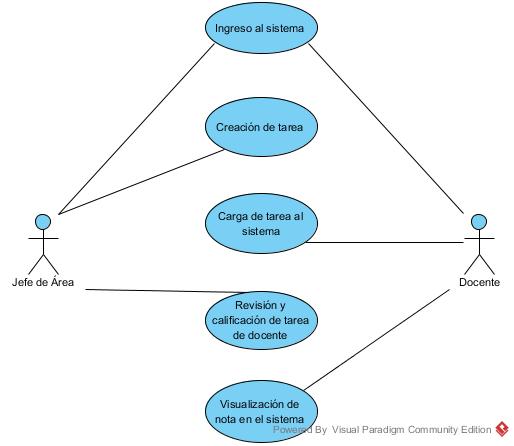


Figura 1. Caso de uso principal

En la figura de caso de uso el jefe de área crea las tareas que estarán disponibles al docente, luego de creadas el docente carga la tarea al sistema para que el jefe de área pueda visualizar y calificar, finalmente el docente tiene disponible la nota de su tarea en el sistema.

Tabla 2. Escenario ingreso al sistema

|  |  |
| --- | --- |
| **Caso de uso** | Ingreso al sistema |
| **Descripción** | El jefe de área y el docente ingresan a la aplicación |
| **Actores** | Jefe de área  Docente |
| **Precondiciones** | Tener una cuenta registrada en la aplicación |
| **Postcondiciones** | Se mostrará la pantalla principal de la aplicación |
| **Escenario principal** | El usuario ingresa su nombre de usuario y contraseña  El sistema redirige a la página principal de la aplicación  Se muestra el resumen de las tareas y reuniones del usuario |
| **Flujo alternativo** | El usuario ingresa su nombre de usuario y contraseña  Se muestra el mensaje que sus credenciales no son válidas  El sistema no redirige a ninguna otra página |

Tabla 3. Creación de tarea

|  |  |
| --- | --- |
| **Caso de uso** | Creación de tarea |
| **Descripción** | El jefe de área crea la tarea a un docente o grupo de docentes. |
| **Actores** | Jefe de área |
| **Precondiciones** | Haber ingresado al sistema exitosamente. |
| **Postcondiciones** | Estará registrada una tarea en la base de datos y podrá ser visualizada por los docentes asignados. |
| **Escenario principal** | El jefe de área selecciona desde el menú tarea crear nueva tarea.  Ingresa los detalles de la tarea.  Selecciona los docentes a los que se va a asignar la tarea.  Registra la tarea desde el botón guardar. |
| **Flujo alternativo** | Se perdió la sesión del usuario  La tarea no se puede crear |

Tabla 4. Carga de tarea al sistema

|  |  |
| --- | --- |
| **Caso de uso** | Carga de tarea al sistema |
| **Descripción** | El docente carga la tarea que el jefe de área le ha asignado |
| **Actores** | Docente |
| **Precondiciones** | Haber iniciado sesión en el sistema  Seleccionar la tarea |
| **Postcondiciones** | La tarea está registrada como enviada y el jefe de área la puede visualizar en su lista de tareas enviadas. |
| **Escenario principal** | El docente selecciona la tarea desde la lista de tareas pendientes  Selecciona y sube un archivo con un comentario de la tarea  La tarea se registra como enviada |
| **Flujo alternativo** | El docente selecciona la tarea  Selecciona y sube un archivo con un comentario de la tarea  No selecciona el botón enviar  La tarea sigue registrada como pendiente |

Tabla 5. Revisión y calificación de tarea

|  |  |
| --- | --- |
| **Caso de uso** | Revisión y calificación de tarea |
| **Descripción** | El jefe de área revisa y selecciona una calificación de la tarea enviada por el docente |
| **Actores** | Jefe de área |
| **Precondiciones** | Que el docente haya enviado la tarea y esté registrada como enviada |
| **Postcondiciones** | La tarea estará marcada como calificada  No se mostrará en la lista de tareas por calificar del jefe de área  La tarea no estará en la lista de tareas enviadas del docente |
| **Escenario principal** | El jefe de área selecciona una tarea enviada  Descarga el archivo  Selecciona una calificación  Registra la calificación |
| **Flujo alternativo** |  |

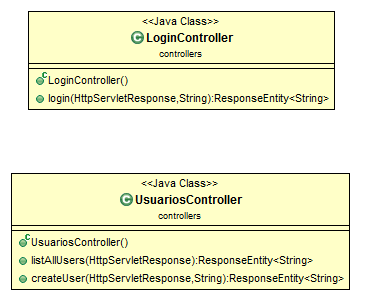
Tabla 6. Visualización de nota en el sistema

|  |  |
| --- | --- |
| **Caso de uso** | Visualización de nota en el sistema |
| **Descripción** | El docente visualiza la nota asignada por el jefe de área en el sistema |
| **Actores** | Docente |
| **Precondiciones** | Que el jefe de área haya enviado la calificación de la tarea  Que la tarea esté registrada como calificada |
| **Postcondiciones** |  |
| **Escenario principal** | Docente ingresa a la lista de tareas calificadas.  Selecciona la tarea.  Visualiza la nota y observaciones de la tarea. |
| **Flujo alternativo** |  |

## Diagramas de clases

### Controladores

Los controladores son las clases que mapean los servicios web para que puedan ser consumidos por la aplicación del cliente



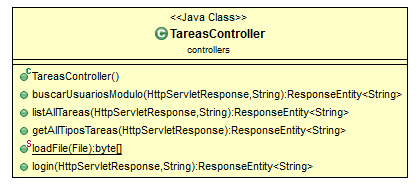


Figura 3. Diagrama de clases controladores

Utilitarios

Son clases que cumplen tareas específicas como la inserción de datos, consultas, procesamiento de datos, etc.

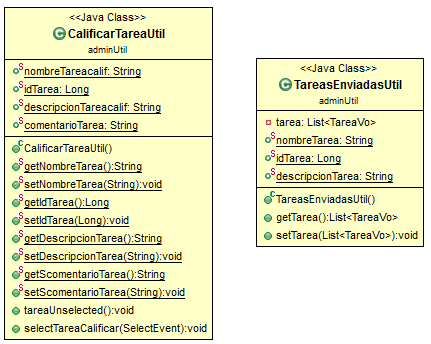


Figura 4. Diagrama de clases Utilitarios

**Entidades**

Son clases que representan tablas en la base de datos, se crean objetos de las tablas a los que se puede modificar y llamar a un método de inserción para que los registre como una entrada en la base de datos.

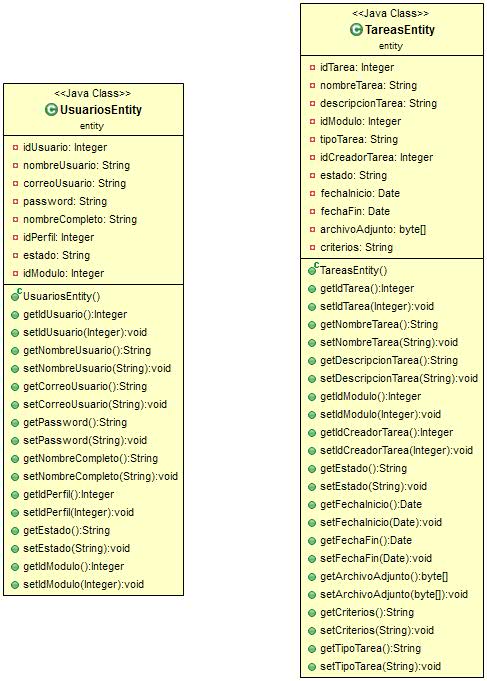


Figura 5. Diagrama de clases entidades

View objects

Los view objects representan una estructura de petición o respuesta que viene en formato JSON y es lo que se devuelve y se requiere como parámetro en los servicios web.

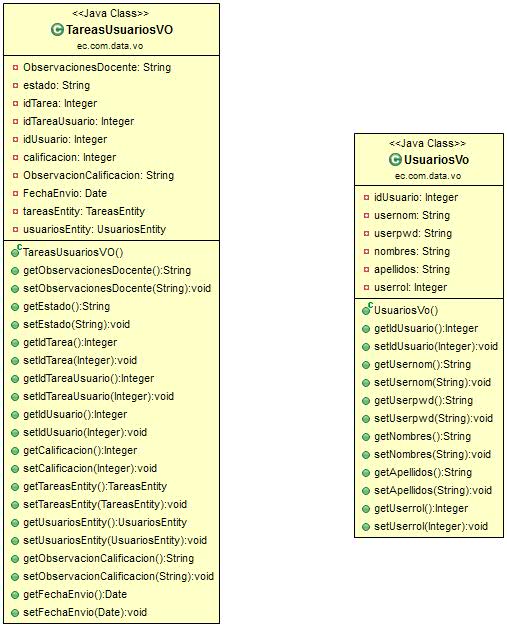


Figura 7. Diagrama de clases entidades

## Interfaz de usuario

### Angular Material

En la fase de análisis y diseño se decidió usar el framework Angular.JS por lo que un framework de interfaz de usuario que se acopla perfectamente a Angular es Angular Material, un framework perteneciente a Google que posee propiedades y directivas propias de Angular por lo que su integración no tuvo mayor impacto en el desarrollo del sistema.

# 4 Construcción y pruebas

### Modelo vista controlador

El modelo vista controlador organiza una aplicación de manera que los datos, la lógica y la interfaz de usuario están separados y conectados por un módulo principal, en la aplicación se utiliza este modelo en el backend con el framework Spring y en el frontend con AngularJS, el objetivo principal de este patrón de diseño de software es la mantenibilidad, añadiendo modularidad y organizando el código de forma tal que si existen cambios en la lógica, datos o interfaz éstos sean realizados en el módulo correspondiente.

Los objetos que conforman el patrón de diseño son:

#### Modelo

Los objetos modelo representan la información de la aplicación y la forma cómo se van a manipular y procesar esta información, si existen cambios en la lógica del negocio, el modelo no va a cambiar ya que el controlador utiliza únicamente los métodos accesores a los objetos del modelo

#### Vista

Los objetos que representan la vista son los que el usuario final puede ver, interactuar y con los que se manipulará los datos del modelo. La vista se comunica con el modelo mediante el controlador.

#### Controlador

Los objetos controlador son los encargados de conectar la lógica del negocio con los datos (modelo) y la interfaz (vista), el usuario final interactúa con la aplicación en la vista, sus acciones son interpretadas en el controlador y modificadas en el modelo.

## Base de datos

La base de datos utilizada utiliza un modelo entidad relación donde las tareas son creadas como un registro único y se unen mediante clave foránea con la tabla TareasUsuarios que contiene un id de tarea y un registro por cada docente que se le asignó la tarea. Los usuarios (docentes o jefes de área) tienen asignados perfiles o roles y cada uno tiene acceso o no a módulos del sistema.

La tabla de criterios contiene los parámetros de evaluación de tareas con lo que cada uno mostrará diferentes componentes al momento de evaluar una tarea que ha sido enviada.

El tipo de tareas discrimina entre tareas y reuniones, cuando un jefe de área decide crear una tarea selecciona si va a ser tarea o reunión y los detalles de registro varían según el caso.

### Diagrama de base de datos

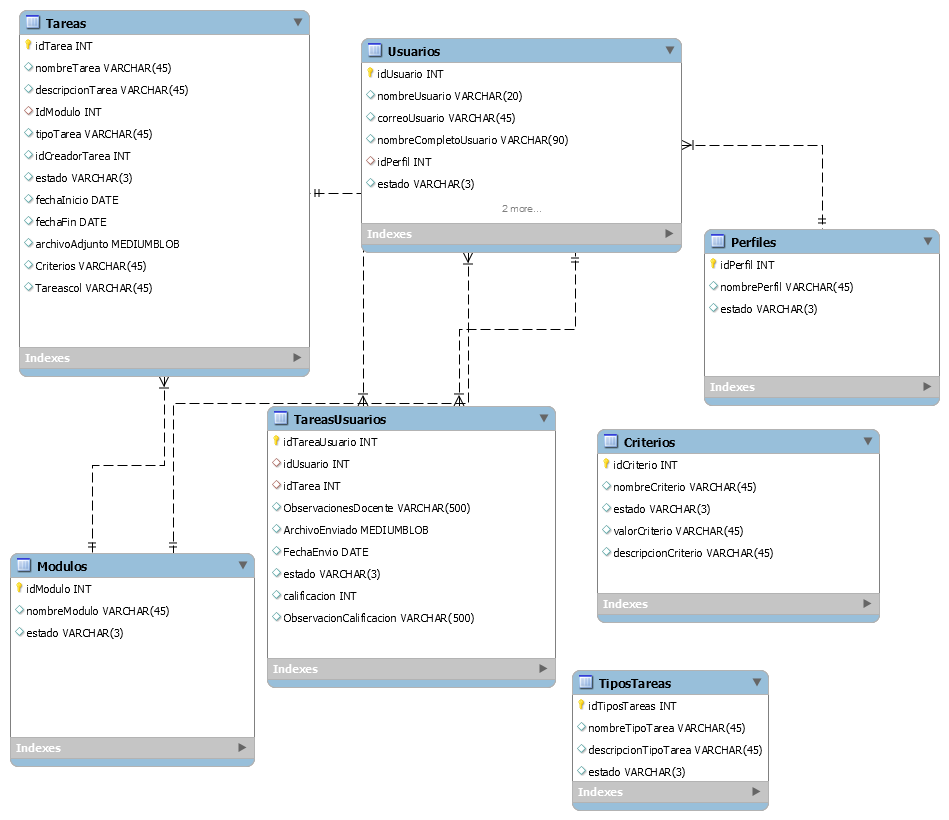


Figura 3. Diagrama de base de datos

## Arquitectura de servicios Web

El usuario final interactúa con los datos mediante el servidor de aplicaciones javascript (Node.js) que llama a los servicios web publicados de la aplicación java. La forma en que se envía y recibe respuestas desde y hacia los servicios web es en texto en formato JSON en el que se puede representar estructuras de datos básicas como objetos y arreglos.

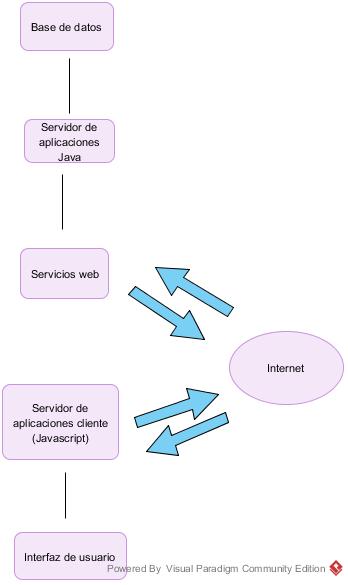


Figura 7. Arquitectura de servicios web

## Casos de uso

### Administración de tareas

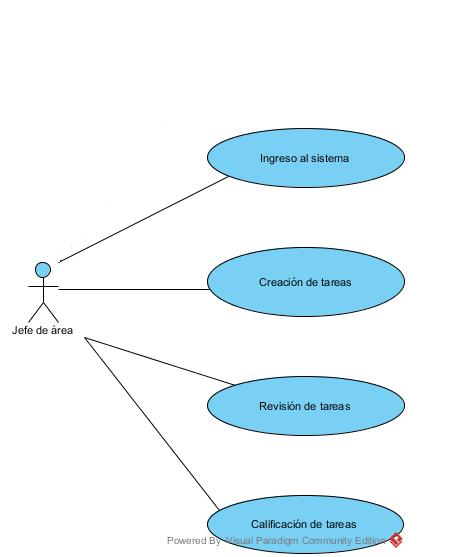


Figura 8. Caso de uso administración de tareas.

Tabla 7. Ingreso al sistema

|  |  |
| --- | --- |
| **Caso de uso** | Ingreso al sistema |
| **Descripción** | Ingreso y verificación de nombre de usuario y contraseña |
| **Actores** | Jefe de área |
| **Precondiciones** | Tener registrado un usuario y contraseña en el sistema |
| **Postcondiciones** | Ingreso al sistema exitoso |
| **Escenario principal** | * El jefe de área ingresa su nombre de usuario y contraseña * El sistema verifica las credenciales e ingresa al sistema |
| **Flujo alternativo** | * Las credenciales no son correctas * Se muestra un mensaje de credenciales incorrectas |

Tabla 9. Creación de tarea

|  |  |
| --- | --- |
| **Caso de uso** | Creación de tarea |
| **Descripción** | El jefe de área registra una tarea en el sistema |
| **Actores** | Jefe de área |
| **Precondiciones** | Haber ingresado al sistema |
| **Postcondiciones** | Tarea registrada y asignada a docentes |
| **Escenario principal** | * Ingreso al módulo de tareas * Selecciona opción nueva tarea * Selecciona el tipo de tarea (Tarea) * Ingresa los detalles de la tarea * Presiona el botón guardar |
| **Flujo alternativo** | * Ingreso al módulo de tareas * Ingreso al módulo de tareas * Selecciona opción nueva tarea * Selecciona el tipo de tarea (Tarea) * Ingresa los detalles de la tarea incompletos * Mensaje de datos de tarea incompletos |

Tabla 10. Revisión de tareas

|  |  |
| --- | --- |
| **Caso de uso** | Revisión de tareas |
| **Descripción** | Ingreso y revisión de tarea de docente |
| **Actores** | Jefe de área |
| **Precondiciones** | Seleccionar una tarea enviada |
| **Postcondiciones** | Tarea abierta y marcada como leída |
| **Escenario principal** | * Ingreso al módulo de tareas * Selecciona tarea en lista de tareas enviadas * Observa los detalles de la tarea |
| **Flujo alternativo** |  |

Tabla 11. Calificación de tareas

|  |  |
| --- | --- |
| **Caso de uso** | Calificación de tareas |
| **Descripción** | El jefe de área selecciona y registra calificación de tarea |
| **Actores** | Jefe de área |
| **Precondiciones** | Haber ingresado a los detalles de la tarea |
| **Postcondiciones** | Tarea registrada como calificada |
| **Escenario principal** | * Selecciona una tarea de la lista * Observa los detalles y selecciona calificar * Selecciona los detalles de calificación * Envía la calificación |
| **Flujo alternativo** | * Selecciona una tarea de la lista * Observa los detalles y selecciona calificar * Selecciona los parámetros de calificación incompletos * Se muestra un mensaje que no ha seleccionado todos los parámetros de calificación |

### Registro de Reuniones

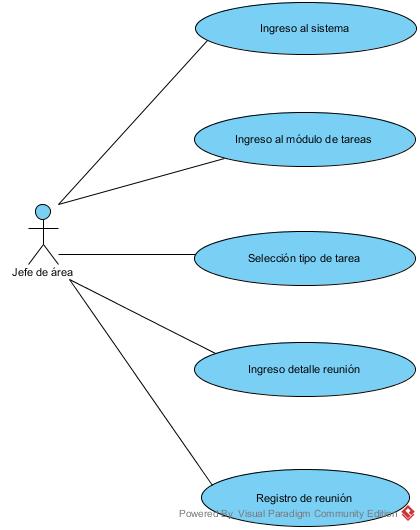


Figura 9. Registro de reuniones

Tabla 12. Ingreso al sistema

|  |  |
| --- | --- |
| **Caso de uso** | Ingreso al sistema |
| **Descripción** | Autenticación en el sistema |
| **Actores** | Jefe de área |
| **Precondiciones** | Tener una cuenta registrada en el sistema |
| **Postcondiciones** | Ingreso exitoso al sistema |
| **Escenario principal** | * Ingresa nombre de usuario * Ingresa contraseña * Selecciona ingresar * El sistema verifica los datos de autenticación * Ingresa a la pantalla principal del sistema |
| **Flujo alternativo** | * Ingresa nombre de usuario * Ingresa contraseña * Selecciona ingresar * El sistema verifica los datos de autenticación * Se muestra un mensaje de credenciales inválidas |

Tabla 13. Ingreso al módulo de tareas

|  |  |
| --- | --- |
| **Caso de uso** | Ingreso al módulo de tareas |
| **Descripción** | Ingresa al módulo de administración de tareas |
| **Actores** | Jefe de área |
| **Precondiciones** | Haberse autenticado en el sistema |
| **Postcondiciones** | Se encuentra en la pantalla principal del módulo de tareas |
| **Escenario principal** | * Se autentica en el sistema * Selecciona tareas en el menú principal * Ingresa al módulo de tareas |
| **Flujo alternativo** |  |

Tabla 14. Selección tipo de tarea

|  |  |
| --- | --- |
| **Caso de uso** | Selección tipo de tarea |
| **Descripción** | Selecciona reunión como tipo de tarea |
| **Actores** | Jefe de área |
| **Precondiciones** | Haber ingresado en el módulo de tareas |
| **Postcondiciones** | La tarea requerirá los detalles específicos de tipo reunión |
| **Escenario principal** | * Ingreso al módulo de tareas * Selecciona nueva tarea * En tipo de tarea selecciona Reunión |
| **Flujo alternativo** | * Ingreso al módulo de tareas * Selecciona nueva tarea * En tipo de tarea selecciona Tarea |

Tabla 15. Ingreso detalle reunión

|  |  |
| --- | --- |
| **Caso de uso** | Ingreso detalle reunión |
| **Descripción** | Ingresa los detalles y el archivo (si existe) de la tarea tipo reunión |
| **Actores** | Jefe de área |
| **Precondiciones** | Haber seleccionado Reunión como tipo de tarea |
| **Postcondiciones** | El formulario de reunión lleno y listo para el registro |
| **Escenario principal** | * Selecciona tarea tipo reunión * Llena los campos de los detalles de reunión * Selecciona archivo adjunto |
| **Flujo alternativo** | * Selecciona tarea tipo reunión * Llena los campos de los detalles de reunión |

Tabla 16. Registro de reunión

|  |  |
| --- | --- |
| **Caso de uso** | Registro de reunión |
| **Descripción** | Registra la reunión en la base de datos |
| **Actores** | Jefe de área |
| **Precondiciones** | Haber llenado los campos obligatorios para el tipo de tarea Reunión |
| **Postcondiciones** | Reunión registrada en la base de datos |
| **Escenario principal** | * Ingresa al módulo de tareas * Selecciona nueva tarea * Selecciona tipo de tarea Reunión * Llena los campos del detalle de la reunión * Selecciona los docentes asignados a la reunión * Presiona “Guardar” * La reunión se registra en la base de datos |
| **Flujo alternativo** | * Ingresa al módulo de tareas * Selecciona nueva tarea * Selecciona tipo de tarea Reunión * No llena los campos obligatorios del detalle de la reunión * Presiona “Guardar” * Se muestra un mensaje que no se han llenado todos los campos obligatorios de los detalles de la reunión |

### Envío de tareas

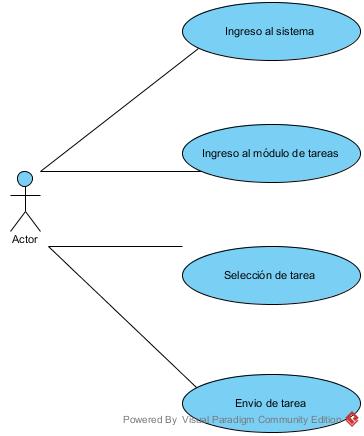


Figura 10. Caso de uso envío de tareas

Tabla 17. Ingreso al sistema

|  |  |
| --- | --- |
| **Caso de uso** | Ingreso al sistema |
| **Descripción** | Autenticación con nombre de usuario y contraseña en el sistema |
| **Actores** | Docente |
| **Precondiciones** | Tener una cuenta registrada en el sistema |
| **Postcondiciones** | Ingreso a la página principal del sistema |
| **Escenario principal** | * Ingresa a la aplicación * Ingresa nombre de usuario * Ingresa contraseña * Selecciona ingresar * Accede a la pantalla principal del sistema |
| **Flujo alternativo** | * Ingresa a la aplicación * Ingresa nombre de usuario * Ingresa contraseña * Selecciona ingresar * Se muestra un mensaje de alerta de credenciales inválidas |

Tabla 18. Ingreso al módulo de tareas

|  |  |
| --- | --- |
| **Caso de uso** | Ingreso al módulo de tareas |
| **Descripción** | Ingresa a la pantalla principal del módulo de tareas |
| **Actores** | Docente |
| **Precondiciones** | Haberse autenticado en el sistema |
| **Postcondiciones** | Ingresar al módulo principal de tareas |
| **Escenario principal** | * Se autentica en el sistema * Selecciona tareas desde el menú principal * Ingresa al módulo de tareas |
| **Flujo alternativo** |  |

Tabla 19. Selección de tarea

|  |  |
| --- | --- |
| **Caso de uso** | Selección de tarea |
| **Descripción** | Selecciona una tarea registrada por el jefe de área |
| **Actores** | Docente |
| **Precondiciones** | Haber ingresado al módulo de tareas |
| **Postcondiciones** | Entrar a los detalles de la tarea |
| **Escenario principal** | * Ingresa al módulo de tareas * Selecciona de la lista de tareas registradas la tarea a enviar * Ingresa a los detalles de la tarea |
| **Flujo alternativo** |  |

Tabla 20. Envío de tarea

|  |  |
| --- | --- |
| **Caso de uso** | Envío de tarea |
| **Descripción** | Envía la tarea y se registra como enviada en la base de datos |
| **Actores** | Docente |
| **Precondiciones** | Haber ingresado a los detalles de la tarea |
| **Postcondiciones** | La tarea se registra como enviada y no aparece en la lista de tareas pendientes del docente |
| **Escenario principal** | * Selecciona una tarea de la lista de tareas pendientes * Ingresa a los detalles de la tarea * Llena el campo comentarios * Selecciona y adjunta un archivo * Presiona “Enviar” * La tarea se encuentra registrada como enviada |
| **Flujo alternativo** | * Selecciona una tarea de la lista de tareas pendientes * Ingresa a los detalles de la tarea * Llena el campo comentarios * No selecciona un archivo * Presiona “Enviar” * Se muestra un mensaje de alerta que debe seleccionar un archivo adjunto |

## Diagramas de clases

### Controladores

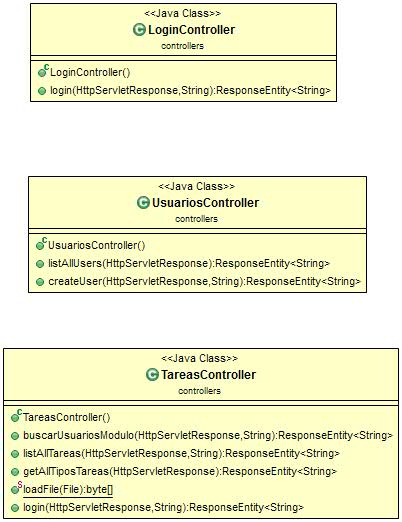


Figura 11. Diagrama de clases controladores

### Servicios

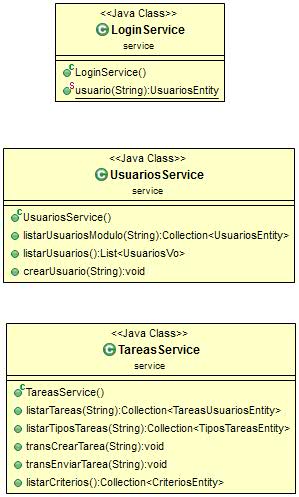


Figura 12. Diagrama de clases Servicios

### Gestores

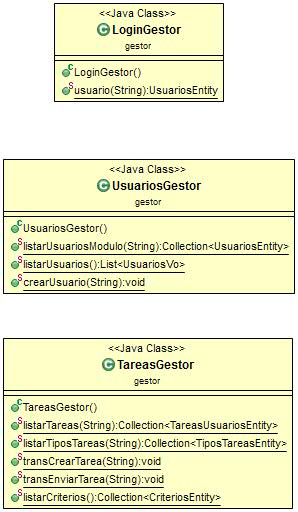


Figura 13. Diagrama de clases gestores

### Módulos

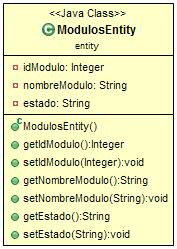


Figura 14. Diagrama de clases módulos

### Criterios

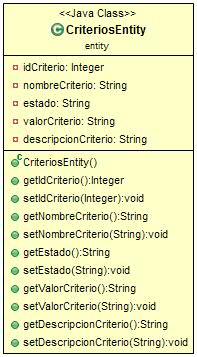


Figura 15. Diagrama de clases criterios

### Tareas

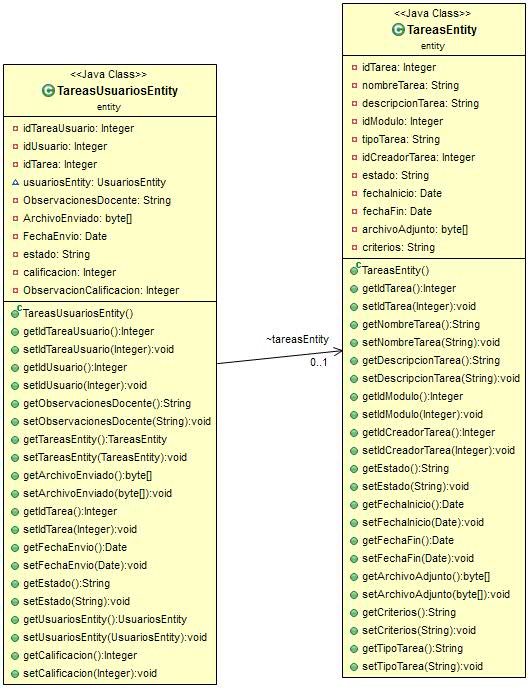


Figura 14. Diagrama de clases tareas

### Usuarios

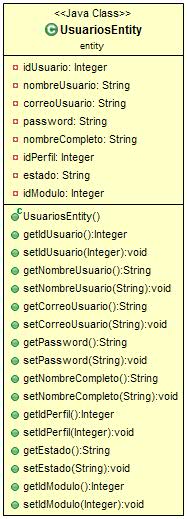


Figura 15. Diagrama de clases usuarios

### Utilitarios

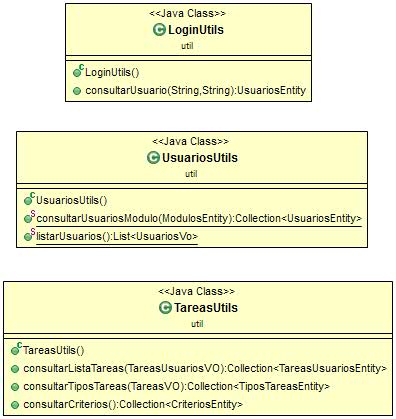


Figura 16. Diagrama de clases utilitarios

## Interfaces de usuario

A continuación se muestran las pantallas de las principales interfaces de usuario del sistema:

### Login



Figura 18. Interfaz de usuario – login

### Pantalla de inicio

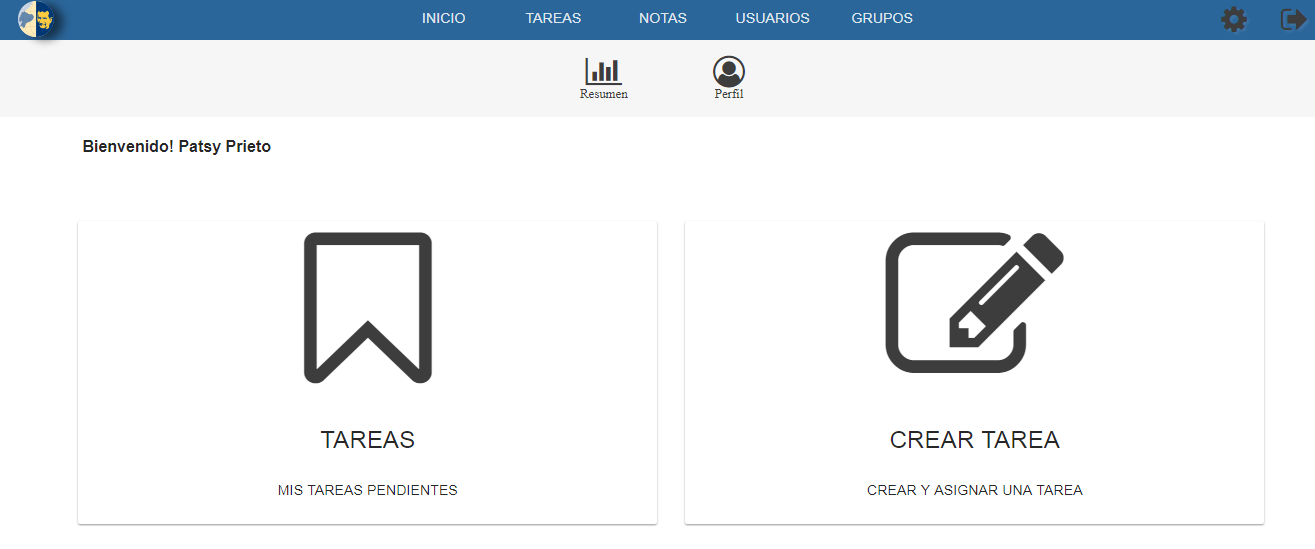


Figura 19. Pantalla de inicio

### Categorías de tareas



Figura 20. Categorías de tareas

### Lista de tareas

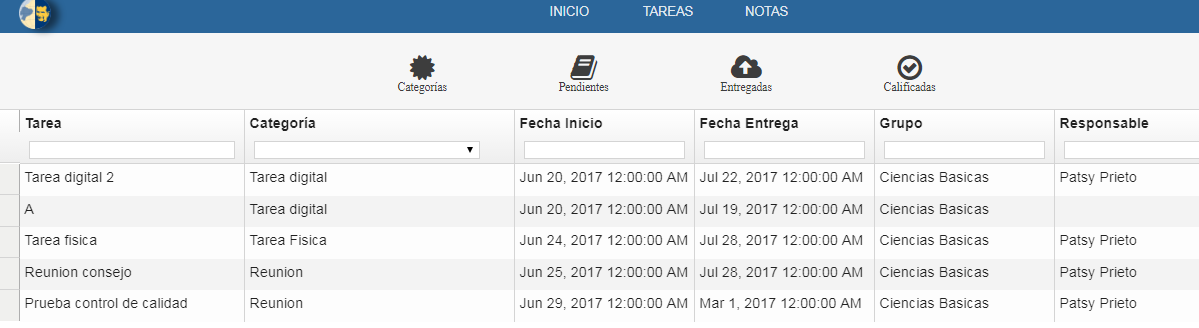


Figura 21. Lista de tareas.

### Enviar tarea



Figura 24.Enviar tarea.

### Calificar tarea

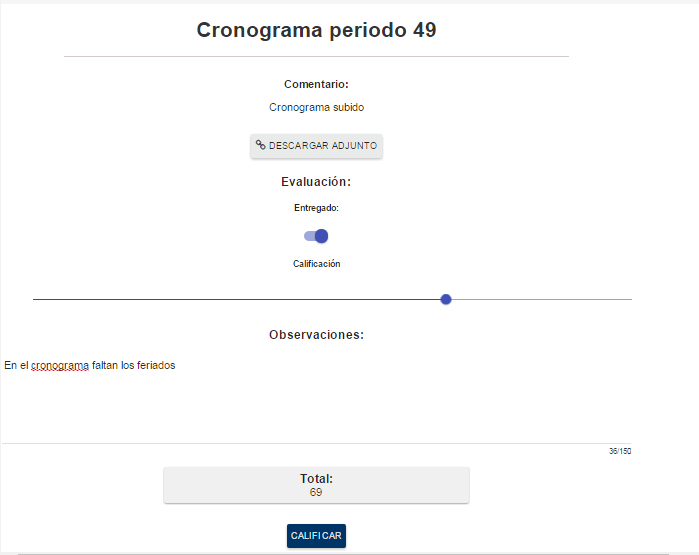


Figura 22. Calificar tarea.

### Crear tarea

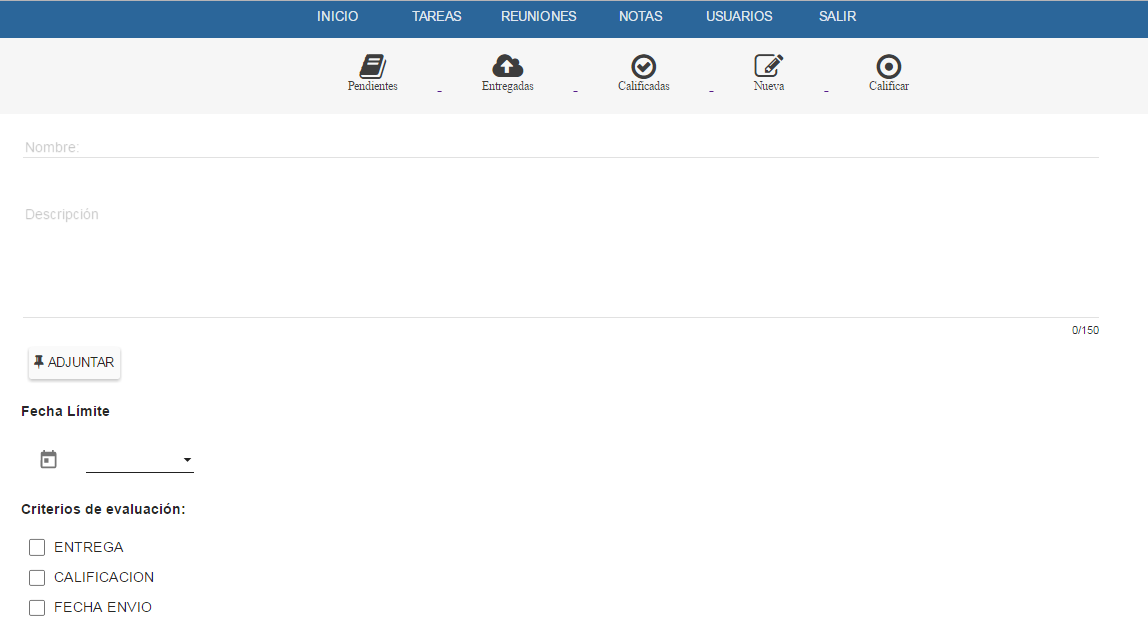


Figura 22. Crear tarea.

### Reporte de notas

Figura 23. Reporte de notas.

### Reportes de docentes

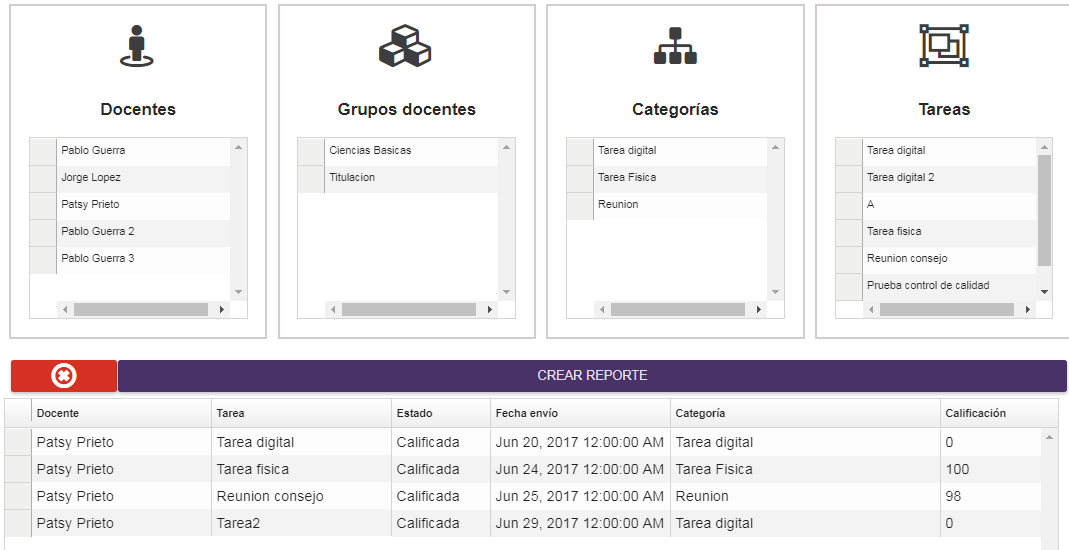


Figura 24. Reporte de docentes.

## Desafíos

Debido a que la metodología usada era incremental e iterativa en el desarrollo podían suscitarse cambios de lógica del sistema y uno de los cambios con más impacto al sistema fue que los docentes puedan pertenecer a varios grupos (ej. titulación, ciencias básicas) por lo que para este cambio se tuvo que crear una nueva tabla en la base de datos que guarde los diferentes grupos al que pertenece el usuario. En la parte de programación

el objeto usuario contiene una colección de objetos gruposUsuario entonces cuando se obtiene la lista de usuarios también se consulta la tabla gruposUsuario y así es posible

editar los grupos a los que pertenece un usuario y el funcionamiento de la aplicación no sufrió un gran impacto.

## Plan de pruebas

### Objetivo

Especificar qué partes del sistema van a ser sometidas a pruebas, señalando la razón por la cual estas partes son cruciales para el flujo normal del proceso.

### Alcance de las pruebas

Los módulos del sistema que se van a probar incluyen:

* Módulo Tareas
* Módulo Usuarios
* Módulo Reportes

### Enfoque de las pruebas

Las pruebas se enfocarán principalmente en el flujo del sistema, debido a que no existirá una afluencia considerable del sistema el enfoque principal no es al rendimiento sino al uso que le darán los docentes y jefes de área.

### Criterios de aceptación

* Existe una fluidez normal en el uso del sistema
* No existen demoras considerables en las llamadas a servicios del sistema
* El usuario sabe qué hacer para completar una tarea

### Entregables

Para la realización de pruebas se utilizarán cuadros con casos de pruebas que contienen métricas, resultados y criterios de las pruebas a partir de un promedio de los usuarios que realizaron las pruebas.

### Recursos

#### Herramientas de pruebas

Para las pruebas de rendimiento se utilizara la herramienta JMeter que sirve para probar el comportamiento funcional y medir el rendimiento de aplicaciones web con servicios Java.

#### Personal

El personal necesario para la realización de las pruebas fue:

* Administrador del sistema para pruebas de rendimiento
* Un equipo con conocimientos de sistemas y experiencia en control de calidad de software.

## Ejecución de pruebas

### Pruebas de usabilidad

#### Módulo Tareas

Tabla 21. Pruebas de usabilidad módulo tareas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tarea** | **Valor a medir** | **Valor actual** | **Peor valor posible** | **Valor objetivo** | **Mejor valor posible** |
| Creación de tarea | Tiempo para creación de tarea exitosa | 3 minutos | 6 minutos | 2 minuto | 1 minuto |
| Envío de tarea | Envío de tarea exitoso | 2 minutos | 5 minutos | 1 minuto | 1 minuto |
| Calificación de tarea | Número de errores | 1 error | 3 errores | 0 errores | 0 errores |

#### Módulo Usuarios

Tabla 22. Pruebas de usabilidad módulo usuarios

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tarea** | **Valor a medir** | **Valor actual** | **Peor valor posible** | **Valor objetivo** | **Mejor valor posible** |
| Creación de usuario | Número de errores | 1 error | 3 errores | 1 error | 0 errores |
| Edición de usuario | Tiempo del primer intento | 1 minuto | 2 minutos | 30 segundos | 1 minuto |

#### Módulo Reportes

Tabla 23. Pruebas de usabilidad módulo reportes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tarea** | **Valor a medir** | **Valor actual** | **Peor valor posible** | **Valor objetivo** | **Mejor valor posible** |
| Ingreso al módulo reportes | Tiempo para ingreso al módulo de reportes desde la pantalla de inicio | 30 segundos | 120 segundos | 10 segundos | 20 segundos |
| Creación de reporte sin filtros | Tasa de completación | 2 intentos | 5 intentos | 1 intento | 1 intento |
| Creación de reporte con filtros | Número de errores | 1 error | 3 errores | 0 errores | 0 errores |

#### Conclusión pruebas usabilidad

Luego de una introducción y capacitación al sistema los usuarios de prueba tuvieron pocos problemas con el uso del sistema, algunos tuvieron problemas con los datos necesarios para la completación de ciertas tareas por lo que se encontraron validaciones faltantes y fueron corregidas después de las pruebas.

### Pruebas de rendimiento

Para las pruebas de rendimiento se utilizó el escenario de la consulta de tareas que es una de las que más carga lleva al servidor.

**Tiempo de respuesta**

#### 1 Usuario

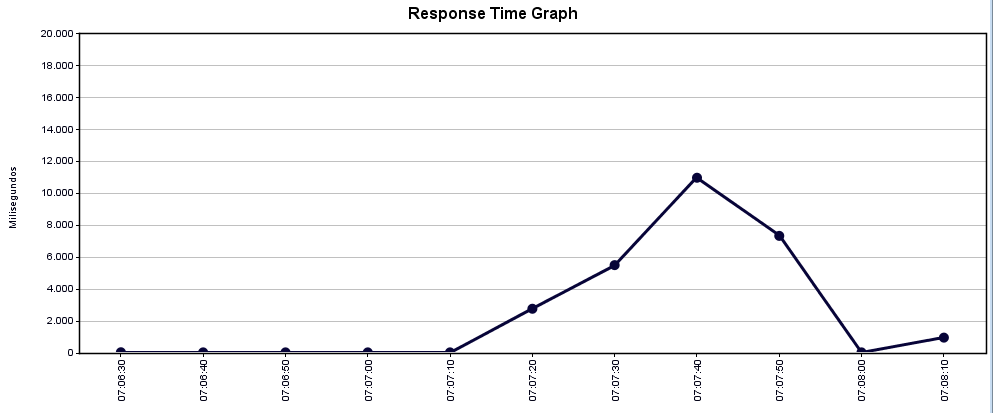


Figura 25. Pruebas de rendimiento consulta de tareas 1 usuario

Con 1 usuario el tiempo de respuesta es de 12ms por lo que el rendimiento para 1 usuario es aceptable.

**5 Usuarios**

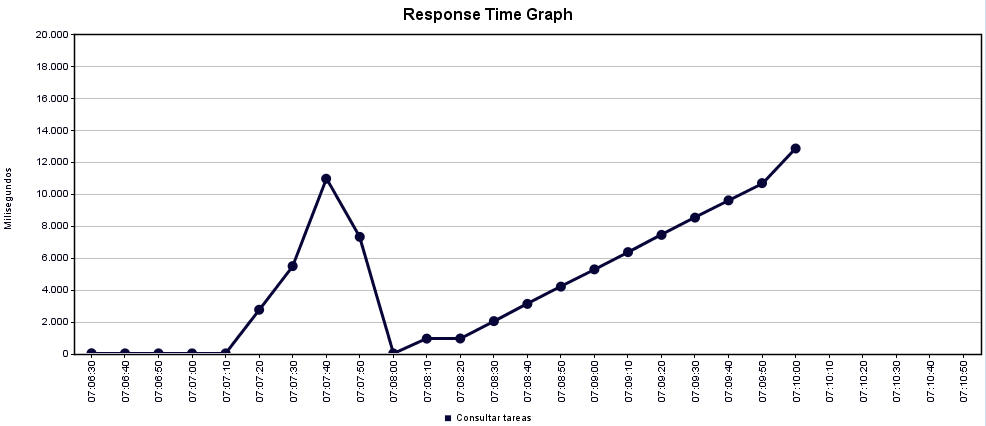


Figura 26. Pruebas de rendimiento con 5 usuarios

Con 5 usuarios al mismo tiempo la repuesta sube a aproximadamente 14ms lo cual es aceptable para una aplicación java.

**100 Usuarios**

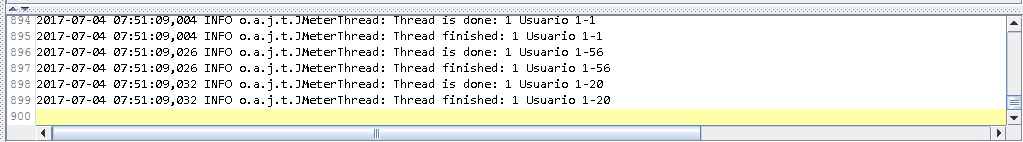


Figura 27. Pruebas de rendimiento con 100 usuarios

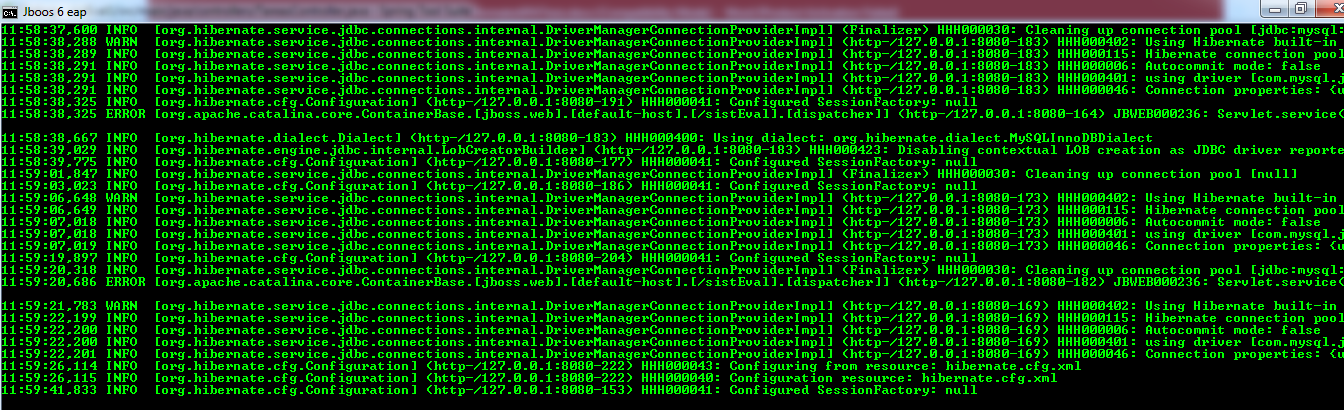


Figura 28. Pruebas de rendimiento con 100 usuarios

Con 100 usuarios la máquina virtual de Java alcanzó su capacidad máxima de memoria y dejó de responder.

# Conclusiones y Recomendaciones

## Conclusiones

Recibiendo retroalimentación de los docentes y jefes de área de la carrera de ingeniería de sistemas se llegó a la conclusión de que el sistema va a mejorar el proceso de evaluación de los docentes de la carrera y cuando esté implementado se van a optimizar los procesos y se van a emitir mejores juicios y se podrá observar un mejor panorama sobre el rendimiento de los docentes en cuanto a las responsabilidades extracurriculares.

# 

# Lista de referencias

Kenneth, S. Rubin, (2012). Essential SCRUM: A practical guide to the most popular agile process

Jamae Javid, Johnson Peter (2009). JBoss in Action: Installation, configuration and deployment

Fisher Mark, Partner Jonas, Bogoevici Marius, Fuld Iwein (2012) Spring integration in action

Teixeira Pedro (2013) Professional Node.js: Building Javascript based scalable software

Rauch Guillermo (2012) Smashing Node.js: Javascript Everywhere

Bauer Christian, King Gavin () Hibernate in action: A guide to the concepts and practice of object/relational mapping

Kotaru V. Keerti (2016) Material Design implementation with AngularJS

Bell Charles (2016) MySQL for the internet of things: Data management for sensors and connected devices

Seddighi Reza Ahmad (2009) Spring Persistence with Hibernate

Naik Kshirasagar, Tripathy Priyadarshi (2008) Software testing and quality assurance: Theory and practice.

McCullough Matthew, Loeliger Jon (2012) Version control with Git: Powerful tools and techniques for collaborative software development.

James Rumbaugh, Ivar Jacobson, Grady Booch (2005) The Unified Modeling Language Reference Manual

Eckel Bruce (2006) Thinking in Java: The definitive introduction to objetct oriented programming in the language of the world wide web.

Arana, J. & Parreño, T. (2016). *CREACIÓN DE UN FRONT END USANDO HTML5, PARA EXPLOTAR LAS FUNCIONALIDADES DE LA PLATAFORMA MOODLE, ADEMÁS DE UNA APLICACIÓN MÓVIL ANDROID PARA EL MANEJO OPORTUNO DE NOTIFICACIONES CON SINCRONIZACIÓN A GOOGLE CALENDAR* (Ingeniería). Universidad Politécnica Salesiana.

Apple Inc. (2015). Model-View-Controller. 2017, de Apple Inc. Sitio web: <https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html>

Google. (2010). What is Angular. 2017, de Google Sitio web: <https://docs.angularjs.org/guide/introduction>

# 