



**Tabla de Contenido**

[Introducción 2](#_heading=h.1fob9te)

[Objetivos 3](#_heading=h.3znysh7)

[Justificación 4](#_heading=h.2et92p0)

[Análisis de requerimientos de software 5](#_heading=h.tyjcwt)

[Arquitectura del sistema 6](#_heading=h.3dy6vkm)

[Especificación de casos de uso 7](#_heading=h.1t3h5sf)

[Diagrama de clases 8](#_heading=h.4d34og8)

[Diccionario de datos 9](#_heading=h.2s8eyo1)

[Mockups 10](#_heading=h.17dp8vu)

[Conclusión 19](#_heading=h.2xcytpi)

[Bibliografía 20](#_heading=h.1ci93xb)

# Introducción

El presente documento especificará los requisitos en el cual se pretende desarrollar y dar una solución acerca de la necesidad presentada por la empresa Siigo, para la gestión de un servicio de facturación y gestión de inventarios la cual se pueda actualizar en tiempo real el stock y esta no debe permitir chequear productos que no se encuentran disponibles, y por último una notificación donde se le informe al proveedor que un producto está llegando a tope.

# Objetivos

Se pretende a través del presente documento dar a conocer el proceso e implantación del software contable el cual ayudará a dar solución al problema planteado por la empresa Siggo. Pretendemos transmitir esta información de manera práctica y sencilla utilizando contenido multimedia para facilitar la fácil comprensión del mismo.

Se pretende poder desarrollar una solución óptima y eficaz que cumpla las expectativas esperadas por parte de la empresa Siigo patrocinadora del proyecto y también al Sena y todos sus instructores quienes nos han formado para obtener el conocimiento óptimo para desarrollar este proyecto.

# Justificación

Teniendo en cuenta el presente proyecto, sobre la realización de un software que va a permitir realizar una gestión de un servicio de facturación y gestión de inventarios, en la cual se va a hacer uso de un Framework llamado Laravel, en la cual permite crear bases de datos escalables, ya que permite exportar a diferentes motores de bases de datos y se va a usar para construir el Back-end del software. Por otro lado, esta propuesta se hace para ofrecer una solución aplicada a la necesidad de una política sólida, en temas de mejoramiento de interacción con las áreas y el mejoramiento de identificación del campo de la empresa.

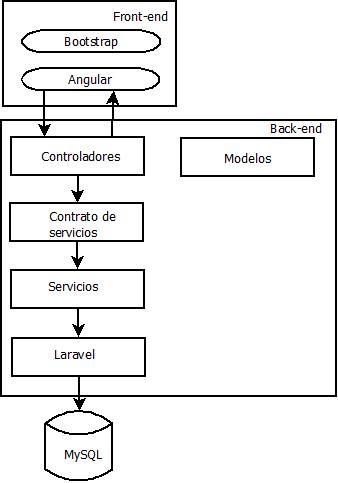
# Análisis de requerimientos de software

Se tiene previsto que HACKATHON sea un aplicativo web construido con tecnologías de vanguardia, lo que garantice un rendimiento más que bueno en los navegadores actuales, además de responder de manera responsiva en los dispositivos móviles. Siendo importante también, la implementación de interfaces claras y amigables e intuitivas con el usuario final. Así mismo también la importancia de una funcionalidad que cumpla con los requerimientos deseados y solucione los problemas propuestos.

HACKATHON se plantea como un software construido bajo los estándares de desarrollo web actuales, tomando como punto de partida, la implementación de su base de datos de datos sobre un motor gratuito, tal como MySQL. En el apartado del Front-end, todas las interfaces de usuario serán diseñadas e implementadas bajo la tecnología de Angular, garantizando así una aplicativo SPA con fluidez de navegación y una notable delegación de responsabilidades sobre el navegador, procurando que el aplicativo responda con eficiencia ante las peticiones del usuario. Por supuesto, todo esto acompañado de una apariencia amigable, moderna y estética.

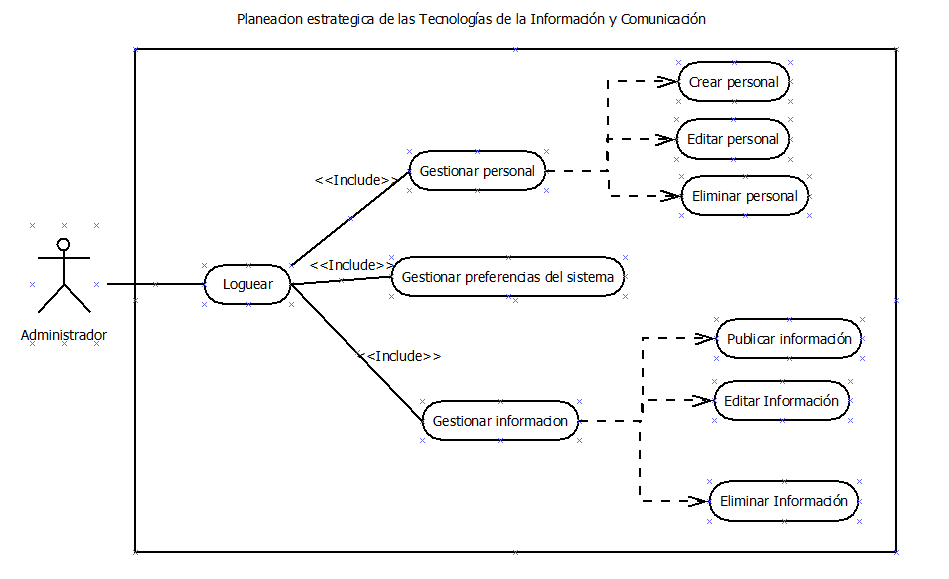
Para la composición del API (servidor) se recurre a Laravel como herramienta de uso gratuito, la cual provee facilidades para la creación de los servicios REST, encargados de recibir las peticiones desde el Front-end. Gracias a la conjunción de estas tecnologías es posible construir una aplicación web confiable, moderna y de gran rendimiento, ya que estas interactúan con sinergia y son de moderado entendimiento, lo cual permite a los desarrolladores ascender en la curva de aprendizaje con mayor facilidad.

# Arquitectura del sistema

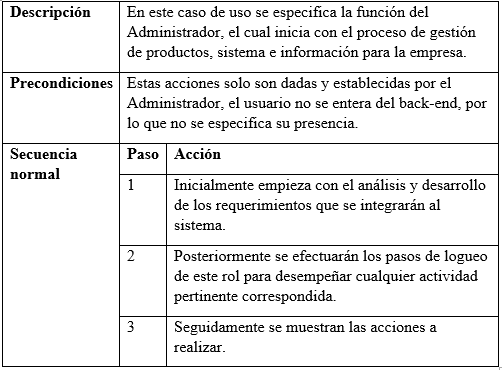


# Casos de uso:

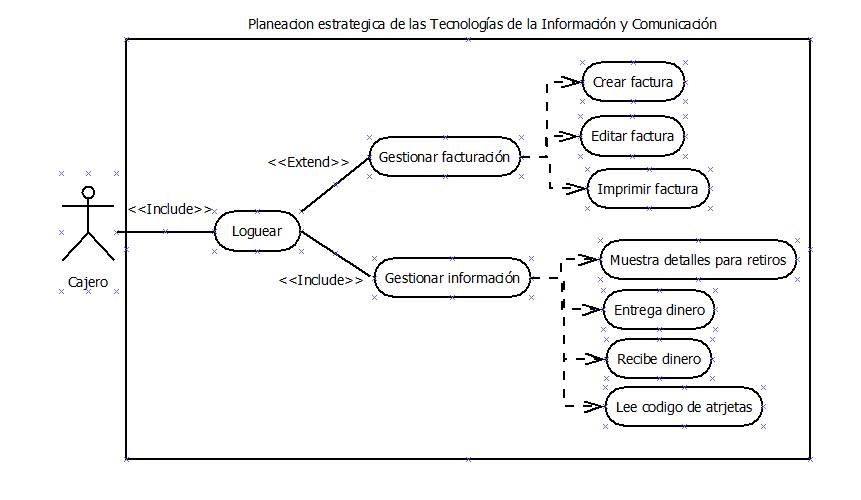
Caso de uso #1: Administrador



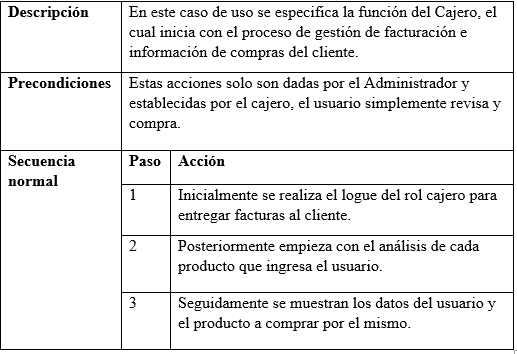
**Explicación:**



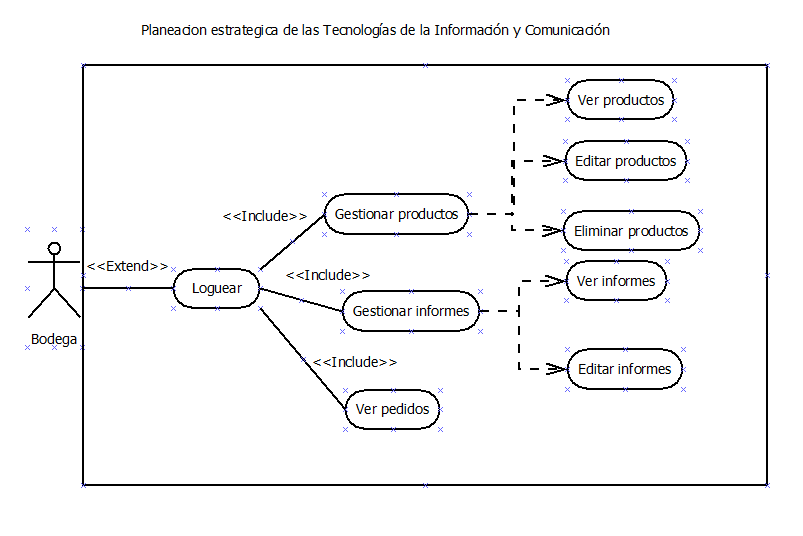
Caso de uso #2: Cajero



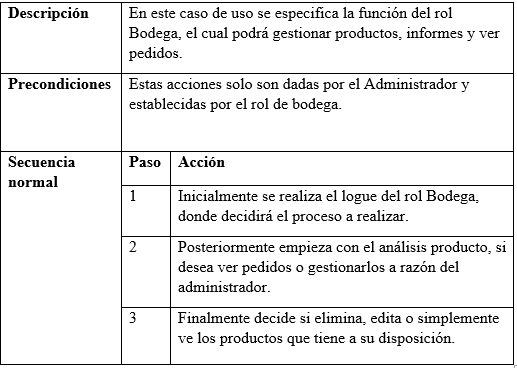
**Explicación:**

****

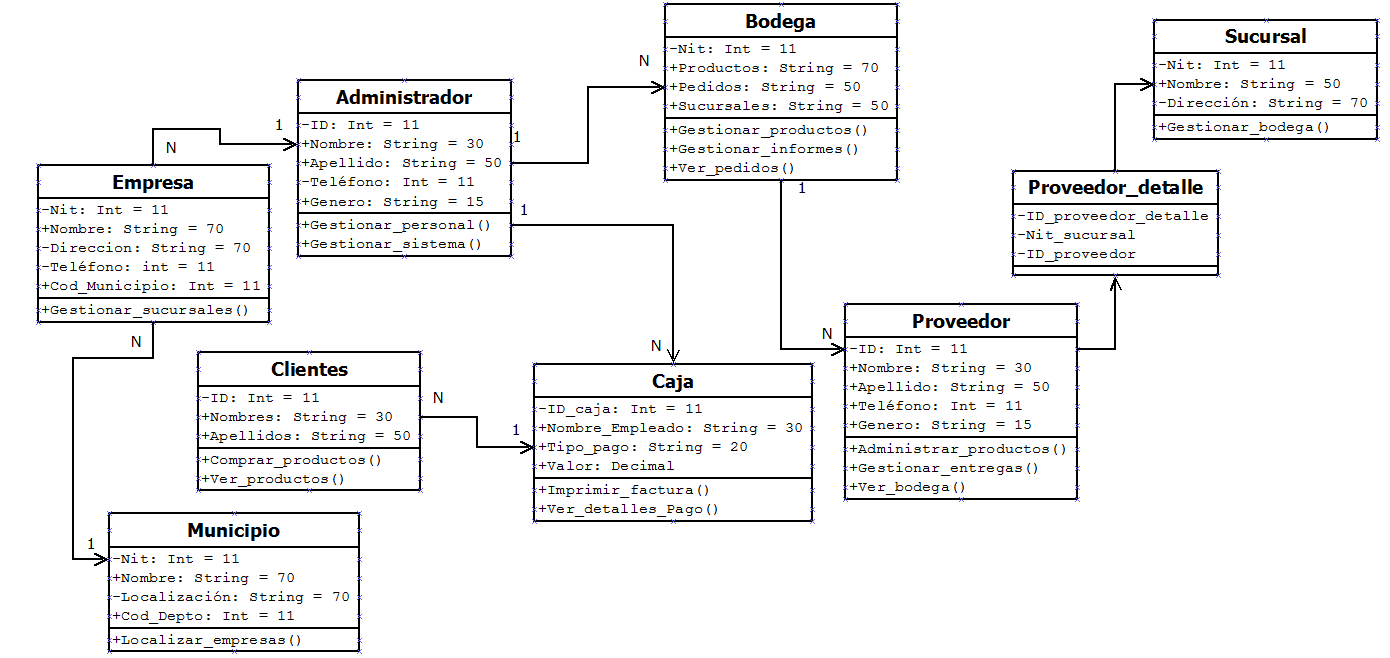
Caso de uso #3: Bodega:



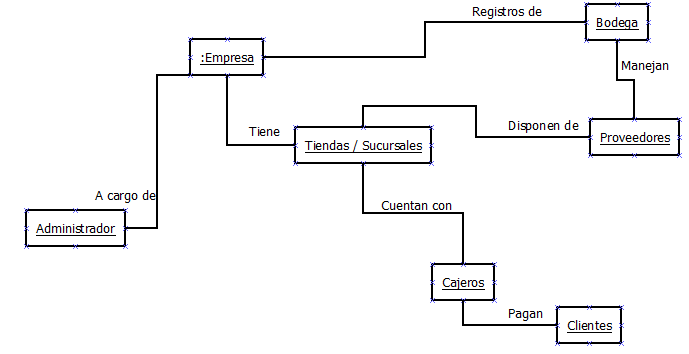
**Explicación:**

****

# Diagrama de clases



# Diagrama de objetos



# Mockups

|  |
| --- |
| Hackathon SIIGO: Login |
|  |
| Mockup Elaborada en Balsamik Wireframes |

|  |
| --- |
| Hackathon SIIGO: Rol –> Caja: Nueva Factura |
|  |
| Mockup Elaborada en Balsamik Wireframes |

|  |
| --- |
| Hackathon SIIGO: Rol –> Caja: Borradores |
|  |
| Mockup Elaborada en Balsamik Wireframes |

|  |
| --- |
| Hackathon SIIGO: Rol –> Caja: Historial Facturas |
|  |
| Mockup Elaborada en Balsamik Wireframes |

|  |
| --- |
| Hackathon SIIGO: Rol –> Bodega: Inventario |
|  |
| Mockup Elaborada en Balsamik Wireframes |

|  |
| --- |
| Hackathon SIIGO: Rol –> Bodega: Inventario - Modificar Producto |
|  |
| Mockup Elaborada en Balsamik Wireframes |

|  |
| --- |
| Hackathon SIIGO: Rol –> Bodega: Agregar Productos |
|  |
| Mockup Elaborada en Balsamik Wireframes |

|  |
| --- |
| Hackathon SIIGO: Rol –> Bodega: Crear Nuevos Productos |
|  |
| Mockup Elaborada en Balsamik Wireframes |

|  |
| --- |
| Hackathon SIIGO: Rol –> Administrador |
|  |
| Mockup Elaborada en Balsamik Wireframes |

**Nota:** El perfil de administrador podrá acceder a todas las secciones mostradas anteriormente, y además será el único que pueda registrar nuevos usuarios con sus respectivos roles.

# Conclusión

Se puede concluir que con este software se usó algunos frameworks para que fuera fácil de cumplir las expectativas sobre los requistos, una de las dificultades va a hacer la estimación del tiempo para poder terminar el programa e intentar de una manera factible que sea funcional y por medio de los diagramas, mockups y una pequeña información en la cual se explica y da una solución al problema que se plantea desde el inicio del documento.

# 