AVANCE DEL PROYECTO MERAKI

Meraki: herramienta para gestión de eventos de socialización

**Álvaro Javier Hernández Rueda**

**Hernán David Álvarez Caballero**

**Desarrollo de Software Orientado a Web**

**Seguridad Informática**

**Redes de Datos**

**Universidad Pontificia Bolivariana**

**Facultad de Ingeniería de Sistemas e Informática**

**Floridablanca**

**2017**

**UNIVERSIDAD PONTIFICIA BOLIVARIANA**

**FACULTAD DE INGENIERÍA DE SISTEMAS E INFORMÁTICA**

**GUÍA PARA LA PRESENTACION DE PROPUESTAS DE PROYECTOS INTEGRADORES O DE AULA**

1. **Introducción**

El presente documento tiene por objeto orientar la presentación de propuestas de proyectos integradores o de aula para estudiantes en la Facultad de Ingeniería de Sistemas e Informática.

El formato adoptado se elabora a partir de los lineamientos para la presentación de proyectos a COLCIENCIAS y las políticas de investigación de la Universidad Pontificia Bolivariana Bucaramanga.

**El documento consta de las siguientes partes:**

**1. INFORMACIÓN GENERAL:**

**1.1 Título:** Meraki: herramienta para gestión de eventos de socialización. [1]

**1.2 Grupo y línea de Investigación.** Redes y Seguridad Informática, Ingeniería de Software, Bases de Datos.

**1.3 Duración del proyecto.** 14 semanas.

**2. PLANTEAMIENTO DEL PROBLEMA O SITUACIÓN PROBLEMATICA**:

Actualmente en la Universidad Pontificia Bolivariana Seccional Bucaramanga se generan diferentes proyectos con distintos alcances y aplicaciones para un entorno laboral o académico, sin embargo, aunque existe el contenido y los ponentes no se cuenta con una herramienta que facilite la información correspondiente a eventos de socialización de proyectos de los estudiantes de la institución. Se cuenta con gran conocimiento en las disciplinas de cada carrera por ello se busca generar una cultura de acercamiento a la socialización de proyectos, innovación y trabajo en equipo para la mejora continua, para así ayudar a la formación de profesionales más íntegros y multidisciplinares.

Por lo tanto, se plantea la siguiente pregunta problema, ¿Cómo centralizar la información de los eventos de socialización de proyectos de los estudiantes de la Universidad Pontificia Bolivariana Seccional Bucaramanga, haciendo uso de tecnologías web que gestionen las actividades, exposiciones, preguntas y horarios en tiempo real?

**3. OBJETIVOS**:

**Objetivo General:** Implementar una herramienta web desarrollada en NodeJs aplicando técnicas de participación en tiempo real que centralice la información correspondiente a los eventos de socialización de proyectos mejorando la experiencia de los participantes.

**Objetivos Específicos:**

1. Identificar las falencias y la información relevante de los eventos de socialización de los estudiantes de la Universidad Pontifica Bolivariana Seccional Bucaramanga.
2. Especificar los requerimientos de la herramienta web.
3. Implementación de la herramienta web en NodeJs aplicando técnicas de tiempo real según los requerimientos establecidos. [2]
4. Calificar el impacto de la plataforma en los participantes del evento de socialización de proyectos de ingeniería de sistemas de la Universidad Pontificia Bolivariana seccional Bucaramanga a través de los requerimientos de testing especificados anteriormente.

**4. JUSTIFICACIÓN:** Este proyecto de Investigación surge dada la necesidad de centralizar la información correspondiente a los eventos de socialización de proyectos con el fin de mejorar la experiencia y generar una cultura de acercamiento a la socialización de proyectos, innovación y trabajo en equipo en la Universidad Pontificia Bolivariana seccional Bucaramanga, donde las personas que integran la institución puedan llegar a conocer más acerca de estos eventos y los temas de interés que se tratan en ellos. A su vez, gracias a la herramienta web ayudar a los docentes en la gestión y evaluación de los eventos; creando la expectativa y la iniciativa de contribuir a un ambiente de apoyo mutuo.

**5. METODOLOGÍA:**

Desarrollo iterativo e incremental de tipo scrum dado que: [3]

* Sistema modular: Las características de la herramienta permiten desarrollar una base funcional mínima y sobre ella ir incrementando las funcionalidades o modificando el comportamiento o apariencia de las ya implementadas.
* Entregas frecuentes y continuas al docente de los módulos terminados, de forma que puede disponer de una funcionalidad básica en un tiempo mínimo y a partir de ahí un incremento y mejora continua del sistema.
* Previsible inestabilidad de requisitos: Es posible que el sistema incorpore más funcionalidades de las inicialmente identificadas. Es posible que durante la ejecución del proyecto se alteren ciertas características. Para el cliente resulta difícil precisar cuál será la dimensión completa del sistema, y su crecimiento puede continuar en el tiempo suspenderse o detenerse.

Personas y roles:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Correo de contacto** | **Rol** |
| Alvaro Hernandez R | Alvaro.hernandez.2014@upb.edu.co | Scrum master |
| Hernan Alvarez Caballero | hernan.alvarez@fryosstudios.com | Scrum master |
| Docentes |  | Product Owner |

**6. RESULTADOS ESPERADOS**: Implementación de la herramienta web en el servidor asignado del Centro de Computación Avanzada (CCA), accesible a toda la comunidad educativa, cumpliendo con los requerimientos levantados a lo largo del desarrollo y especificados en el acta de requerimientos, a su vez la documentación generada bajo los estándares propuestos por la metodología plasmada en las bitácoras de desarrollo junto con los *mockups* y diseños de las vistas presentadas al usuario.

**7. CRONOGRAMA:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Actividades** | **Duración** | **Fecha inicio** | **Fecha Fin** |
| Actividad 1.1. : Diseñar una herramienta para recolectar la información de los puntos a mejorar | 1 | 31 julio | 6 agosto |
| Actividad 1.2. : Implementar la herramienta de recoleccion de información | 1 | 31 julio | 6 agosto |
| Actividad 1.3. : Analisis de los resultados optenidos por la herramienta de recolección de información | 1 | 31 julio | 6 agosto |
| Actividad 2.1. : Reconocer y seleccionar las funcionalidades con base a las necesidades encontradas | 1 | 7 agosto | 13 agosto |
| Actividad 2.2. : Consenso del acta de requerimientos con el cliente | 1 | 7 agosto | 13 agosto |
| Actividad 2.3. : Redactar el acta de requerimientos | 1 | 7 agosto | 13 agosto |
| Actividad 3.1. : Modularizar la herramientas web con base en el acta de requerimientos | 1 | 14 agosto | 20 agosto |
| Actividad 3.2. : Documentarse en la tecnologia y los frameworks a utilizar en el desarrollo | 10 | 14 agosto | 22 octubre |
| Actividad 3.3. : Implementar los modulos necesarios al lado del servidor | 10 | 14 agosto | 22 octubre |
| Actividad 3.4. : Diseñar y maquetar el frontend de la herramienta web | 7 | 4 septiembre | 22 octubre |
| Actividad 4.1. : Evaluar la herramienta web mediante los parametros establecidos en el acta de requerimientos | 1 | 23 octubre | 29 octubre |
| Actividad 4.2. : Analizar los resultados obtenidos de la evaluación | 1 | 23 octubre | 29 octubre |
| Actividad 4.3. : Realizar informe final del proyecto | 4 | 2 octubre | 29 octubre |

1. **Bibliografía**

|  |  |
| --- | --- |
| **[1]** | **Anonimo, «Diccionario Libre,» 16 enero 2016. [En línea]. Available: http://diccionariolibre.com/definicion/meraki/108892.** |
| **[2]** | **L. Foundation, «Node JS,» [En línea]. Available: https://nodejs.org/en/.** |
| **[3]** | **W. Lara, «Platzi,» 2015. [En línea]. Available: https://platzi.com/blog/guia-scrum/.** |

# Requerimientos Software

**Prefacio**

|  |  |
| --- | --- |
|  |  |
|  | Este documento describe los requerimientos de software de Meraki el cual tiene como objetivo centralizar la información de los eventos de socialización de proyectos. |
| **Alcance** | Esta acta de requerimientos de software es la base del desarrollo de software del proyecto. Describe los siguientes tópicos:   * Se busca Implementar una herramienta web aplicando técnicas de participación en tiempo real que centralice la información correspondiente a los eventos de socialización de proyectos.   Este documento no describe:   * Las especificaciones de diseño del aplicativo, la infraestructura de red a implementar. |

**Historia del Documento**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Comentarios** | **Autor** |
| 14/08/2017 | 0.1 | Versión inicial |  |
| 31/08/2017 | 0.2 | Revisión del cliente |  |
|  |  |  |  |

Tabla 1. Historia del documento de requerimientos

## 1.1 INTRODUCCION

### Entorno

En la Universidad Pontificia Bolivariana Seccional Bucaramanga se generan diferentes proyectos con distintos alcances y aplicaciones para un entorno laboral o académico, actualmente no se cuenta con una herramienta en funcionamiento para la gestión del momento de socialización de los proyectos.

### Audiencia

**Usuarios**: Este documento se dirige al usuario con el objetivo de que analice las características y funcionalidades que tendrá el sistema, así mismo de que cumpla sus expectativas, apruebe y de paso a la construcción del software.

**Programador**: el acta de requerimientos de software se dirige al programador con el fin de que entienda la estructura del software (características, funcionalidades, arquitectura, desempeño, entre otros) con lo cual inicie el proceso de diseño del proyecto.

### Documentación

* Documento de especificación de requerimientos
* Diseño de la red
* Configuración de dispositivos de red
* Diccionario de datos
* Manual de usuarios
* Diagrama entidad-relación
* Caso de uso
* Diagrama de actividades
* Diagrama de clase
* Análisis de riesgo

## Requerimientos Software

### Requerimientos Funcionales

|  |  |
| --- | --- |
| FSR1 | Codificación: se contempla la maquetación de la herramienta en HTML5 y CSS3; para el desarrollo backend se contempla NodeJs con el framework SailsJs. |
| FSR2 | Implementar el servicio web en el CCA (Centro de Computación Avanzada) de la UPB seccional Bucaramanga. |
| FSR3 | Diseño de la base de datos. |
| FSR4 | El sistema debe poder verificar la autenticación de ingreso al aplicativo por parte del(los) usuario(s) autorizado(s). |
| FSR5 | La aplicación web debe permitir gestionar (agregar, eliminar, modificar) los horarios de socialización. |
| FSR6 | La aplicación debe tener una vista de visualización de horario para cada evento. |
| FSR7 | La aplicación web debe mostrar qué ponencia se está realizando en el instante de la consulta de acuerdo al horario. |
| FSR8 | Diseño del sitio de la herramienta web. |
| FSR9 | Implementación del aplicativo en el servidor web. |
| FSR10 | El aplicativo web debe permitir a los usuarios participar en tiempo real con preguntas y valoraciones por sustentación. |
| FSR11 | Contemplar esquemas de seguridad sobre el servidor web. |
| FSR12 | El aplicativo debe tener dos tipos de usuario (administrador y participante) los cuales tienen las características de:   * Administrador: Registrar, visualizar, modificar y eliminar eventos, ponencias y participaciones. * Participante: revisar horarios, valorar las ponencias, generar preguntas a los ponentes. |
|  |  |

### Requerimientos de Testing

|  |  |
| --- | --- |
| ST1 | Verificar la conectividad desde dispositivos móviles y de escritorio. |
| ST2 | Presentación de los mockup reflejando el orden de la información. |
| ST3 | Test de funcionalidad a la aplicación web, realizando pruebas de integración para verificar la correcta visualización de la información. |
| ST4 | Verificación dinámica de la implementación de los requerimientos. |
| ST5 | Navegabilidad del sitio web con diferentes navegadores. |
| ST6 | Usabilidad de los formularios en diferentes navegadores. |
| ST7 | Test de seguridad, validando disponibilidad, integridad y confidencialidad de los datos y servicios. |
| ST8 | Prueba de seguridad a la estructura interna del aplicativo web. |

### Requerimientos no funcionales

|  |  |
| --- | --- |
| RNF01 | La interfaz de la web debe ser *responsive design* con el fin de que pueda ser visualizada por cualquier dispositivo. |
| RNF02 | El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final. |
| RNF03 | El sistema debe poseer interfaces gráficas bien formadas. |
| RNF04 | La respuesta que debe tener el usuario por el sistema al consultar la información debe ser eficaz. |
| RNF05 | El aplicativo web debe ser capaz de operar adecuadamente desde cualquier dispositivo. |

### Matriz Requerimientos Funcionales vs. Requerimientos de Testing

| Requerimiento funcional | ST1 | ST2 | ST3 | ST4 | ST5 | ST6 | ST7 | ST8 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| FSR1 |  |  | X |  | X | X |  | X |
| FSR2 |  |  |  |  |  |  | X | X |
| FSR3 |  |  | X |  |  |  | X |  |
| FSR4 |  |  |  |  |  |  |  | X |
| FSR5 |  |  |  | X |  |  | X |  |
| FSR6 |  |  | X |  | X | X | X |  |
| FSR7 | X |  | X | X | X |  |  |  |
| FSR8 |  | X |  | X |  |  |  |  |
| FSR9 | X |  | X | X | X | X | X | X |
| FSR10 |  |  |  | X | X | X |  |  |
| FSR11 |  |  |  | X |  |  | X | X |
| FSR12 | X |  |  | X |  |  |  |  |

Tabla 2. Matriz Requerimientos Funcionales vs. Requerimientos de Testing

# Diseño de arquitectura

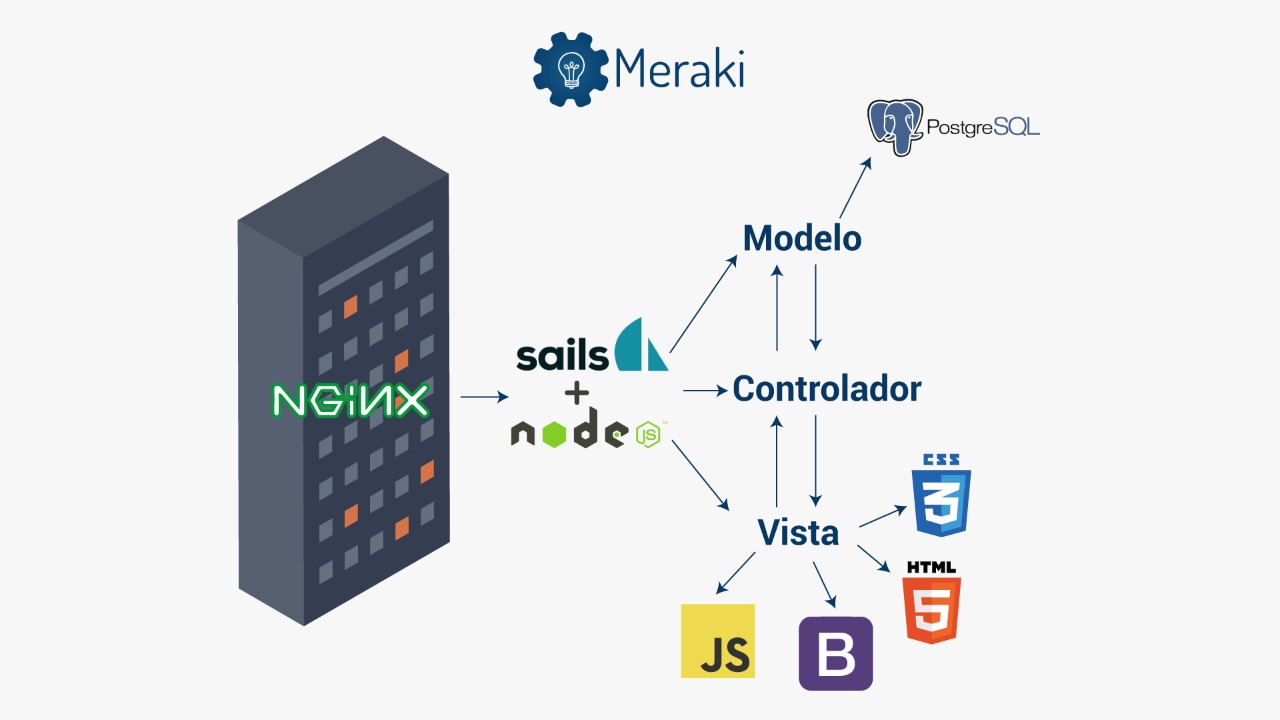


Imagen 1. Diseño de la arquitectura del software

Para el desarrollo del proyecto integrador Meraki se plantea utilizar un servidor web Nginx el cual corre una aplicación desarrollada en nodeJs con el framework Sails.js dado que es un lenguaje que es altamente compatible con diferentes herramientas front-end permitiendo manejar el mismo lenguaje (JS - JavaScript) tanto del lado del cliente como del servidor; para la base de datos se plantea utilizar PostgresSQL debido a que es multiplataforma y utiliza una estrategia de almacenamiento de filas llamada MVCC (control de concurrencia mediante versiones múltiples, *Multiversion concurrency control*, es un método para control de acceso generalmente usado por SGBDs (sistema gestor de base de datos) para proporcionar acceso concurrente a los datos, y en lenguajes de programación para implementar concurrencia), consigue mejor respuesta en grandes volúmenes, MVCC permite a los accesos de solo lectura continuar leyendo datos consistentes durante la actualización de registros, permitiendo copias de seguridad en caliente.

Los controladores, que son los encargados de trabajar los datos y decidir que se muestra, serán manejados en JS conforme se menciona inicialmente. Las vistas, la cuales son encargadas de mostrarme los datos, estarán maquetadas con HTML5, CSS3, JS y el framework Bootstrap.

## Modelo ER:

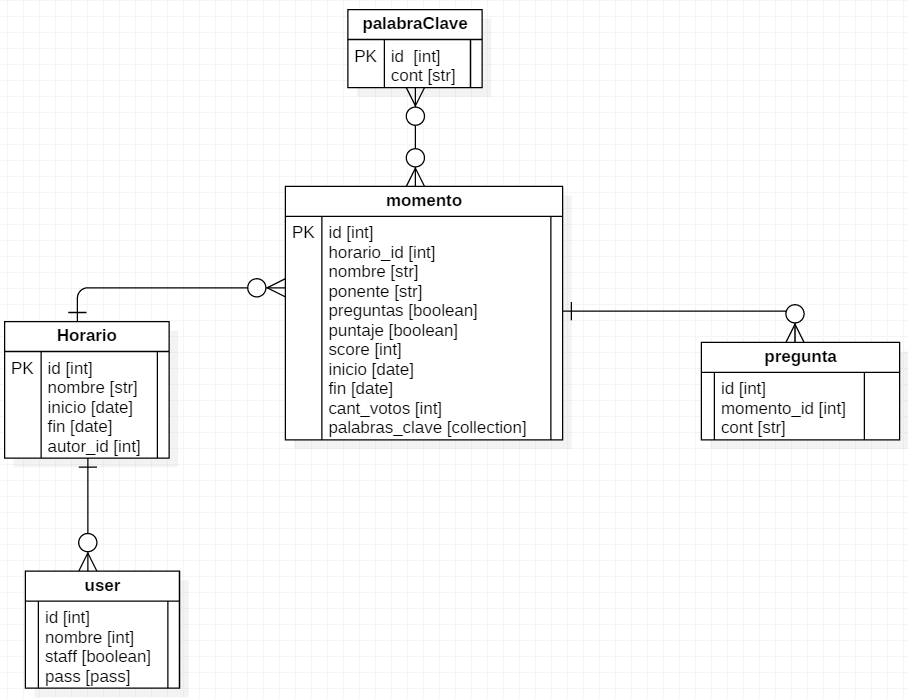


Imagen 2. Modelo entidad relación.

En este modelo se puede ver reflejada la base de datos que requiere inicialmente el aplicativo web, se cuenta con cinco tablas:

* PalabraClave: cuenta con los campos Id y cont, se guardan las palabras claves que puede ser resaltadas en una momento (o ponencia).
* Pregunta: guarda las preguntas que crean los asistentes al ponente, cuenta con id, el id del “momento” en el cual se está preguntando y el contenido de la pregunta.
* User: guardara los usuarios que puedes moderar o administrar los eventos, cuenta con id, nombre, si puede administrar o no (staff) y una contraseña.
* Horario: esta tabla guardara la información general de los eventos, cuenta con id, nombre del evento, fecha de inicio, fecha de finalización y su respectivo creador (referenciado por el id).
* Momento: es la tabla con más columnas y guarda la información de cada momento que se realiza en un evento (ejemplo: una ponencia, un conversatorio o un descanso) cuanta con id, el respectivo evento al que pertenece (referenciado por el id), nombre, nombre del ponente o encargado, si habilitada las preguntas y puntaje, el score (que por defecto es 0), hora de inicio, hora de finalización, cantidad de votos y las palabras clave que puede manejar.

## Diagrama de casos de uso:

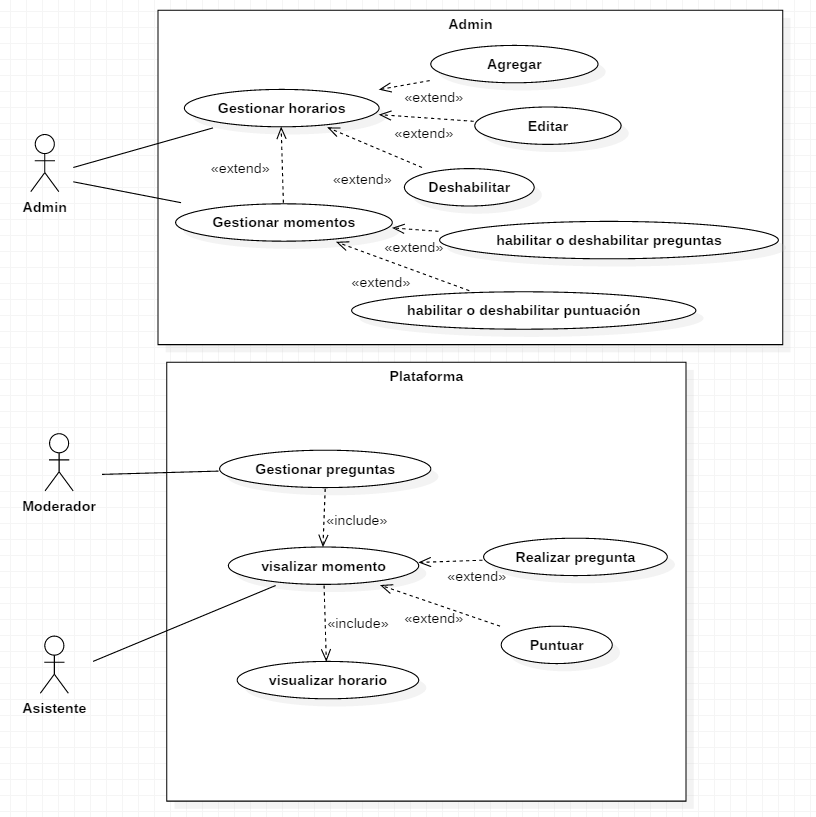


Imagen 3. Casos de uno Meraki

# Tabla api

|  |  |  |
| --- | --- | --- |
| VERB | URI (meraki.upb.edu.co) | CONTROLLER |
| GET | / | HorarioController#index |
| GET | /eventos/ | HorarioController#listarEventos |
| GET | /evento/:horarioId/ | HorarioController#detalle |
| GET | /momento/:momentoId/ | MomentController#detalleMomento |
| GET | /momento/:momentoId/preguntar/ | MomentController# pregMomento |
| GET | /PalabrasClave/ | PalabraController#listar |
| GET | /PalabrasClave/:palabraId/ | PalabraController#detalle |
| GET/POST | /login/ | UserController#login |
| GET/POST | /logout/ | UserController#logout |
| GET/POST | /agregar-evento/ | UserController#agregarEvento |
| GET/POST | /editar-evento/:eventoId/ | UserController#editarEvento |
| GET/POST | /agregar-momento/ | UserController#agregarMomento |
| GET/POST | /agregar-momento/:momentoId/ | UserController#editarMomento |
| GET | /moderador/:momentoId/ | MomentController#moderaMoment |

Tabla 3. Tabla de direcciones URL

En la tabla se presenta las tentativas de URLs que manejaría la herramienta, el nombre de la función del controlador busca retratar al máximo su objetivo, para el index principal de Meraki la ruta es la raíz (/), en las direcciones: /eventos/, /evento/:horarioId/, /PalabrasClave/, /PalabrasClave/:palabraId/ se lista los horarios disponibles, los respectivos momentos, las palabras calves registradas, los momentos activos para dicha palabra clave respectivamente; para realizar una pregunta a un momento determinado la persona debe acceder a la dirección /momento/:momentoId/preguntar/ donde :momentoId es igual al identificador del momento al cual se quiere realizar una pregunta, para el control de usuario de tiene las rutas /login/ y /logout/, se puede gestionar los eventons, momentos y preguntas en: /agregar-evento/, /agregar-momento/, /editar-evento/, /editar-momento/ y /moderador/:momentoId/.

1. Se está buscando la posibilidad de que sean URL amigables o semánticas (las URL semánticas están formadas con palabras relacionadas con el contenido de la página y fáciles de recordar).

**5 Aspectos de seguridad**

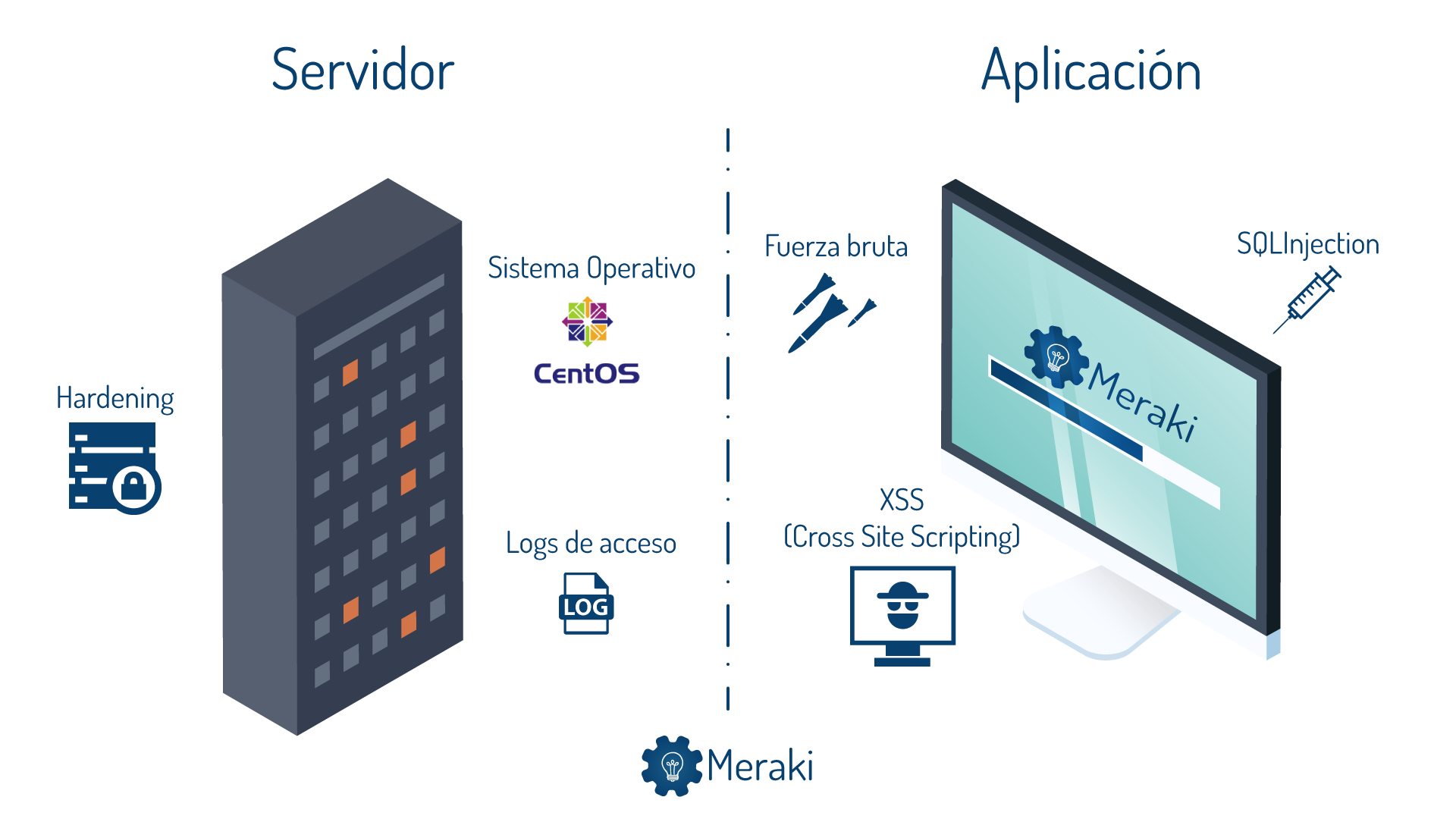


Imagen 4. Aspectos de seguridad

Garantizar que Meraki es una aplicación web robusta y segura es muy importante dado que soluciona un problema de la academia lo que implica que sea confiable y que sus usuarios tengan la certeza de que sus datos son verídicos. Por lo tanto, se plantean los ataques más comunes a aplicaciones y servidores web sobre los que se encuentra blindado Meraki, para ello se divide entre la seguridad implementada en el servidor y la seguridad implementada en la aplicación, para el servidor se utilizan técnicas de hardening buscando evitar la intrusión del mismo por medio de puertos sin utilizar, aplicar contraseñas alfanuméricas para acceder al servidor y bloqueo de pantalla periódico; así mismo el servidor trabaja sobre el sistema operativo CentOS dadas las especificaciones de seguridad recomendadas por el personal del CTIC de la Universidad y se tiene un registro de acceso a la aplicación permitiendo conocer quien, cuando y desde donde ha accedido al servidor lo que permite conocer si alguien ha entrado sin autorización.

Por otro lado, la seguridad de la aplicación contempla ataques de SQLInjection que son comandos de tipo SQL que se ejecutan en las entradas de datos a la aplicación, por lo que tal validación se realiza en el frontend de la aplicación y en el backend para evitar que se extraigan datos de la base de datos desde la vista de usuario. También, se contempla la protección a intrusiones de tipo Cross Site Scripting que es la ejecución de programas de ordenador que se ejecutan en la aplicación y buscan romper esquemas de seguridad para acceder a sus datos; por último, el sitio cuenta con protección a ataques de fuerza bruta en la autenticación de usuarios permitiendo solo 3 intentos a la hora de acceder, si se falla en el tercer intento, se bloquea la cuenta del usuario hasta nueva orden.