Introducción

El presente documento tiene como objetivo presentar el caso de estudio a emplearse como proyecto final de la carrera de Técnico en Redes y Software para el grupo TA3
Se buscará durante este tercer semestre desarrollar e implementar la solución que responda a las necesidades planteadas, cumpliendo con estándares de calidad, performance y seguridad teniendo aspectos fundamentales como alta disponibilidad, balance de carga, tolerancia a fallos y distribución geográfica de los servicios esenciales. El proyecto se realizará en supervisión de los tutores englobando las destrezas y conocimientos adquiridos a lo largo de toda la carrera.

Caso de Estudio

Se desea desarrollar un sistema de visualización de resultados deportivos.

El sistema posibilitará ver resultado de diferentes eventos deportivos (que se hayan realizado, o que se estén realizando en el momento), así como detalles de los mismos (fecha y hora, planteles, árbitros, etc).

A su vez, a los usuarios suscriptos, se les enviaran notificaciones de los eventos que deseen.

El sistema debe estar compuesto por varias aplicaciones interactuando entre sí, siguiendo los lineamientos de la Arquitectura de Microservicios y la arquitectura REST.

Por otro lado, se debe diseñar una solucion de infraestructura que de soporte tanto al sistema, como a las oficinas de la empresa, y que permita la fluida operativa de los empleados para el trabajo diario.

Generalidades del caso

Aplicación de Usuario

Esta aplicación permitirá visualizar los resultados de los eventos, así como ver los detalles de los mismos. Por ejemplo, en un partido de fútbol ver las alineaciones de los equipos participantes, y quienes anotaron, y en un partido de tenis ver quiénes fueron los jugadores y el resultado de los sets.

Se debe reservar un espacio en toda la aplicación para que se muestren banners de publicidad. Esta publicidad será estática o rotativa por sección, dependiendo de los parámetros del banner.

La visualización de los resultados es pública (es decir, no es necesario registrarse para poder visualizar resultados).

Esta aplicación debe permitir el registro de usuarios, el cual permitirá, mediante pago, eliminar los banners de publicidad, así como suscribirse a eventos (ej. Campeonatos, o partidos programados a futuro), y enviar notificaciones del resultado, tanto por correo electrónico, como en el navegador.

Se debe implementar utilizando un framework de Frontend, de forma que acceda a las demás aplicaciones para obtener información.

API de Resultados

Esta aplicación es una simple API pública que brinda los resultados y estadísticas de los eventos, en formato JSON. Debe utilizarse como fuente de datos para la aplicación de usuario, así como ser libre para que cualquiera la consulte.

La API debe ser RESTFul.

API de Autenticación

Esta aplicación es una simple API pública que se encarga de gestionar la autenticación. Se le brindan credenciales y retorna una respuesta JSON indicando si las credenciales son correctas o no. Es la APP que utiliza la app de usuario para autenticarse.

La API debe ser RESTFul, y debe desarrollarse de forma que funcione como un proveedor de autenticación (ej. oAuth), o comunicarse con un sistema externo (otro proveedor de oAuth, Active Directory o LDAP). A su vez, debe soportar autenticación mediante Facebook, Google y Twitter.

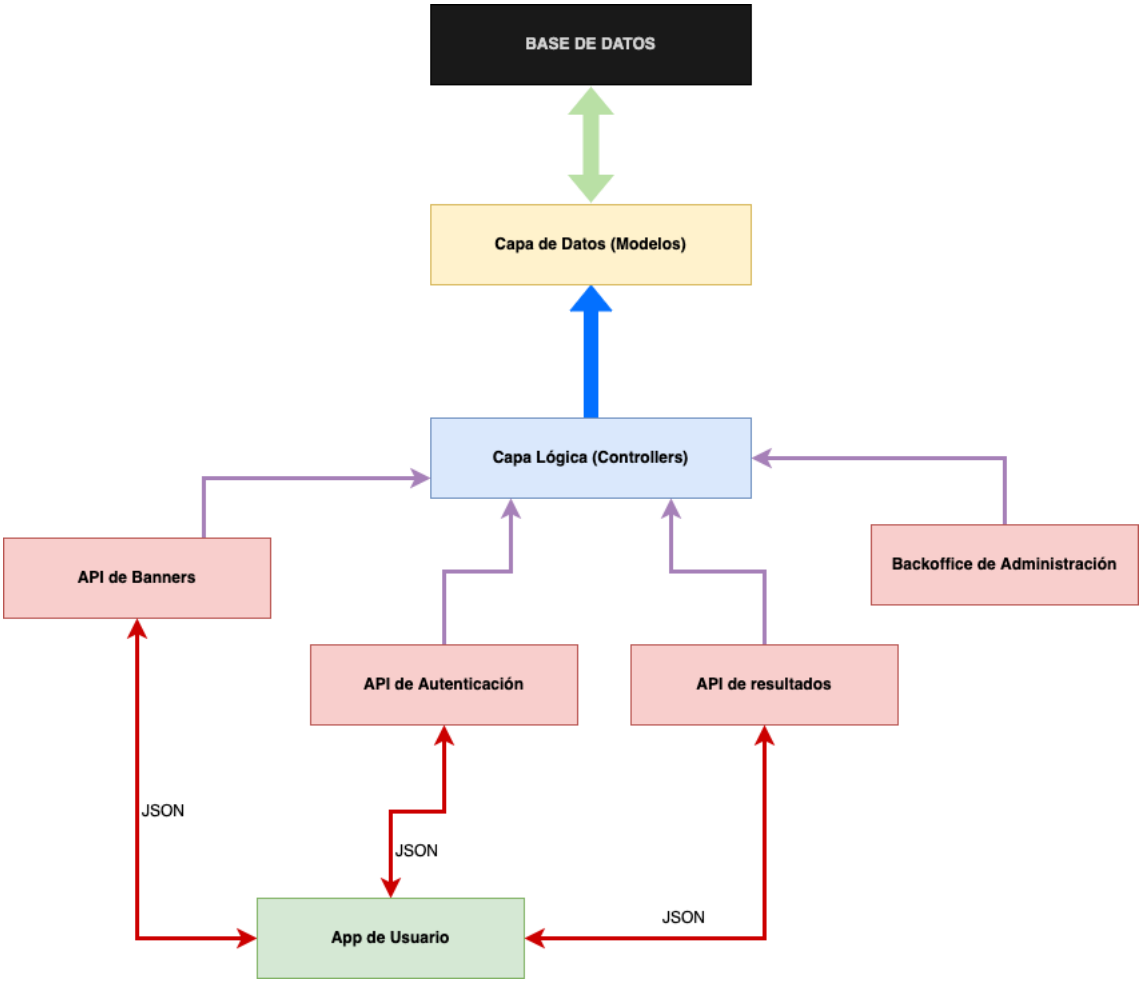
API de Publicidad

Esta aplicación es una simple API pública que retorna en una respuesta JSON con los banners de publicidad que deben mostrarse en un momento dado.

Backoffice de Administración

Esta aplicación de acceso restringido se encarga de gestionar el funcionamiento interno de todo el sistema. Puede desarrollarse como dos aplicaciones (frontend y Backend), o como una aplicación monolítica clásica.

Esquema de Arquitectura del sistema



Implementación

Todas las aplicaciones deben estar “dockerizadas” para ejecutarse en Kubernetes. La interacción de las aplicaciones con otros servicios de la infraestructura debe estar definida de tal forma que la persistencia no se de en los contenedores, y que sea posible la escalabilidad de las aplicaciones.

Se debe tener en cuenta la necesidad de servicios de red para el funcionamiento del sistema, como Base de Datos, Directorio Activo, Caché, Correo electrónico, almacenamiento de repositorios Git, DHCP, DNS, los cuales deben ser implementados.

Ambientes

Se deben tener en cuenta 3 ambientes para desarrollar el proyecto:

- Test: Este ambiente es donde los alumnos desarrollarán el proyecto. Está compuesto por sus PCs personales, en las cuales desarrollarán el software, y donde crearán máquinas virtuales para probar la infraestructura

- Stage: Es el ambiente donde se realizará la entrega final del proyecto. Se ejecutarán los servidores y las aplicaciones en equipos de la escuela. Todos los componentes del proyecto deben estar funcionales

- Producción: Es el ambiente donde correría el software y la infraestructura en su implementación final. No se llegará a implementar, pero debe diseñarse una propuesta de implementación utilizando proveedores de cloud computing, así como se debe entregar un presupuesto de la solución.

Objetivos

- Diseñar y desarrollar todas las aplicaciones que componen el sistema
- Diseñar e implementar la solución de infraestructura de red para el funcionamiento del sistema
- Diseñar e implementar la solución de infraestructura de red para las oficinas de la empresa
- La aplicación web deberá funcionar desde cualquier navegador y ser lo más intuitiva y fácil de usar por el público destinatario

Puntualizaciones

- El sistema deberá manejar varios perfiles de usuario desde el Back Office, y perfiles de usuario anónimo, registrado gratis y registrado pago para la aplicación pública.
- El sistema debe tener tolerancia a fallos (a nivel de datos y de operatividad)
- Se debe llevar un control de históricos y cambios.
- La infraestructura debe implementar interoperabilidad entre plataformas Microsoft y UNIX
- La configuración de los debe implementarse y entregarse mediante un software de provisionamiento de configuraciones (Ansible, Puppet, Chef, etc)
- La autenticación de las aplicaciones debe ser implementada con algún servicio ya existente para dicho fin (oAuth2, LDAP, Active Directory). No se acepta implementar la autenticación y seguridad de forma manual mediante tablas.
- La infraestructura debe implementar soluciones de monitoreo y alertas.
- Tanto la infraestructura como las aplicaciones deben manejar los logs generados de forma centralizada.
- Se debe proveer un sistema de respaldo integral, para todos los servidores.
- La infraestructura para el ambiente de producción debe estar diseñada e implementada de forma que se adapte al uso de tecnologías en la nube. Excepto para el modelo de oficinas, no se acepta implementar como modelo un Data Center local para los servicios.
- Las aplicaciones desarrolladas deben ejecutarse en contenedores Docker
- El código de las aplicaciones debe estar versionado utilizando Git, se deben guardar versiones regularmente en la organización de Github de la escuela. También, mediante Git, se debe implementar Integración y Entrega continuos.
- El componente de Frontend debe desarrollarse utilizando JavaScript y HTML, así como cualquier framework que trabaje con ellos (Angular, ReactJS, VueJS, por ejemplo).
  El componente de Backend debe estar desarrollado en PHP, así como cualquier framework que lo utilice (Laravel, CodeIgniter, por ejemplo).

  Para ambos casos se debe justificar su elección.

  Quedan excluidos frameworks, paquetes o herramientas generadoras de código y lenguajes de 4to nivel como GeneXus
- El motor de base de datos debe ser MySQL o SQL Server, justificando su elección.

El proyecto deberá contar con la debida documentación que corresponda a las distintas etapas del ciclo de vida del proyecto.

Además, se deberá desarrollar una guía de uso en línea enfocada al usuario final, que sea lo más interactiva posible.

Equipos de Trabajo

El trabajo será realizado en grupos de hasta 4 estudiantes. Grupos de más integrantes se analizarán como excepciones. No se aceptarán trabajos individuales.

El grupo deberá haber presentado antes del día 10/05/22 la conformación de los grupos, vía correo electrónico a los docentes.

Pautas para el trabajo

Dado a la naturaleza terciaria y técnica de la carrera, se entiende que las tareas realizadas por el proyecto deben tener el nivel de completitud correspondiente al nivel de la carrera.

El trabajo en el proyecto exige la integración de todos los conocimientos adquiridos en todas las asignaturas en los dos primeros semestres así como la adquisición de nuevos conocimientos por parte del grupo. El hecho de no existir tutorías correspondientes a asignaturas cursadas en el primer y segundo semestre (Diseño web, Conectividad, Seguridad de Redes, Auditoria de Redes, Ingeniería de Software e inglés), no exime la obligatoriedad de la aplicación de dichas asignaturas en el proyecto.

Es de especial importancia el contacto con cada uno de los docentes de las áreas temáticas del proyecto, a efectos de evacuar dudas, en las tutorías que corresponda en cada asignatura. Los docentes podrán solicitar vía correo electrónico cualquier información que estimen pertinente y tendrá la misma validez que hecha en clase.

El proyecto final que se entregue, debe poder ser ejecutado por los docentes en los equipos del Instituto. No se puede entregar ni instalar en máquinas personales de los alumnos.

Entregas

Las entregas se realizarán de forma digital, vía correo electrónico a los docentes, para la documentación.
El código fuente de todos los aspectos del proyecto deben estar hosteados en un repositorio en el espacio de trabajo de Github de la escuela.

Los requerimientos puntuales de cada tutoría, y los criterios de evaluación de las entregas, se acordarán con cada tutor.

Las entregas se realizarán siguiendo el siguiente calendario:

Primer Entrega: viernes 20 de mayo del 2022

Segunda Entrega: viernes 24 de junio del 2022

Tercer Entrega: viernes 15 de Julio del 2022