

GRUPO 2

Auditoría y Calidad en Sistemas de Información

Carlos Stipisich

Lic. Gestión de la Información



Nuestro equipo



**Pablo
Goitisoló**

Licenciado



**Frida
Narbó**

Licenciada



**Emmanuel
Federico**

Licenciado



**Diego
Peppert**

Licenciado

Visión

 **LicitAR:** plataforma web para licitaciones públicas en Argentina.

 **Objetivos:** digitalización, transparencia, trazabilidad y eficiencia.

 **Tecnología:** .NET Core 9, MVC, Razor Pages, SQL Server, Azure.



Justificación del Desarrollo

Desarrollo interno en PPS siguiendo buenas prácticas, adaptado al contexto local.

Justificación de la Auditoría

Alta criticidad del proceso de licitación pública.

Evaluación

Incluye seguridad, confidencialidad, trazabilidad, cumplimiento legal, calidad técnica y prevención de riesgos antes del despliegue.

Alcance

Alcance de la Auditoría

✓ **Evaluación funcional y técnica:** Registro de proveedores, Gestión de licitaciones y publicación, Ofertas digitales y adjudicación y Trazabilidad del proceso.

🧩 **Componentes técnicos auditados:** Seguridad de acceso y control de roles, Arquitectura de microservicios y API Gateway, Azure SQL, App Services, Frontend MVC y Validaciones, logs y documentación técnica.

✗ **Exclusiones:** Infraestructura física fuera de Azure y Mantenimiento postproducción.

👤 **Alcance organizacional:** Auditoría realizada por el equipo de desarrollo (4 integrantes) con roles definidos.

Adquisición vs. Desarrollo Propio

Contexto del Proyecto

- LicitAR nace en la Práctica Profesional Supervisada, desarrollo interno fue la única opción viable y académicamente alineada.

Justificación del Desarrollo Interno

- Equipo con perfiles técnicos y funcionales, permite personalización y aprendizaje práctico, refuerza buenas prácticas en diseño, pruebas y documentación.

Comparación con Sistemas Reales

- Referencia: COMPR.AR, ideas funcionales adaptadas a modelo accesible y didáctico.

Ventajas

- Experiencia formativa completa, control total sobre el sistema, trabajo colaborativo con simulación profesional.

Limitaciones

- Curva de aprendizaje técnica, recursos y tiempos limitados, sin evaluación de costos comerciales.



Metodología de Planificación y Desarrollo

LicitAR se desarrolló con metodología Scrum en 15 sprints. Se usaron Jira y Confluence para tareas y documentación. El equipo trabajó en forma colaborativa con arquitectura MVC, aplicando buenas prácticas. Se realizaron pruebas por sprint y se utilizó SonarQube para controlar la calidad del código.

⚙️ Metodología Ágil – Scrum

- Un total de 15 sprints, con una duración de una semana cada uno
- Entregas parciales, revisión continua y mejoras iterativas
- Herramientas: Jira (tareas), Confluence (documentación).

🧩 Organización del Trabajo

- Inicio: planificación con backlog
- Cierre: retrospectiva y ajustes
- Documentación: historias de usuario, decisiones, pruebas, diagramas

👥 Roles del Equipo

- Diego & Pablo: desarrollo, arquitectura, Azure
- Emmanuel & Frida: pruebas funcionales, documentación
- Trabajo colaborativo y paralelo, basado en Scrum



Metodología de Planificación y Desarrollo

LicitAR se desarrolló con metodología Scrum en 15 sprints. Se usaron Jira y Confluence para tareas y documentación. El equipo trabajó en forma colaborativa con arquitectura MVC, aplicando buenas prácticas. Se realizaron pruebas por sprint y se utilizó SonarQube para controlar la calidad del código.

🔧 Enfoque Técnico

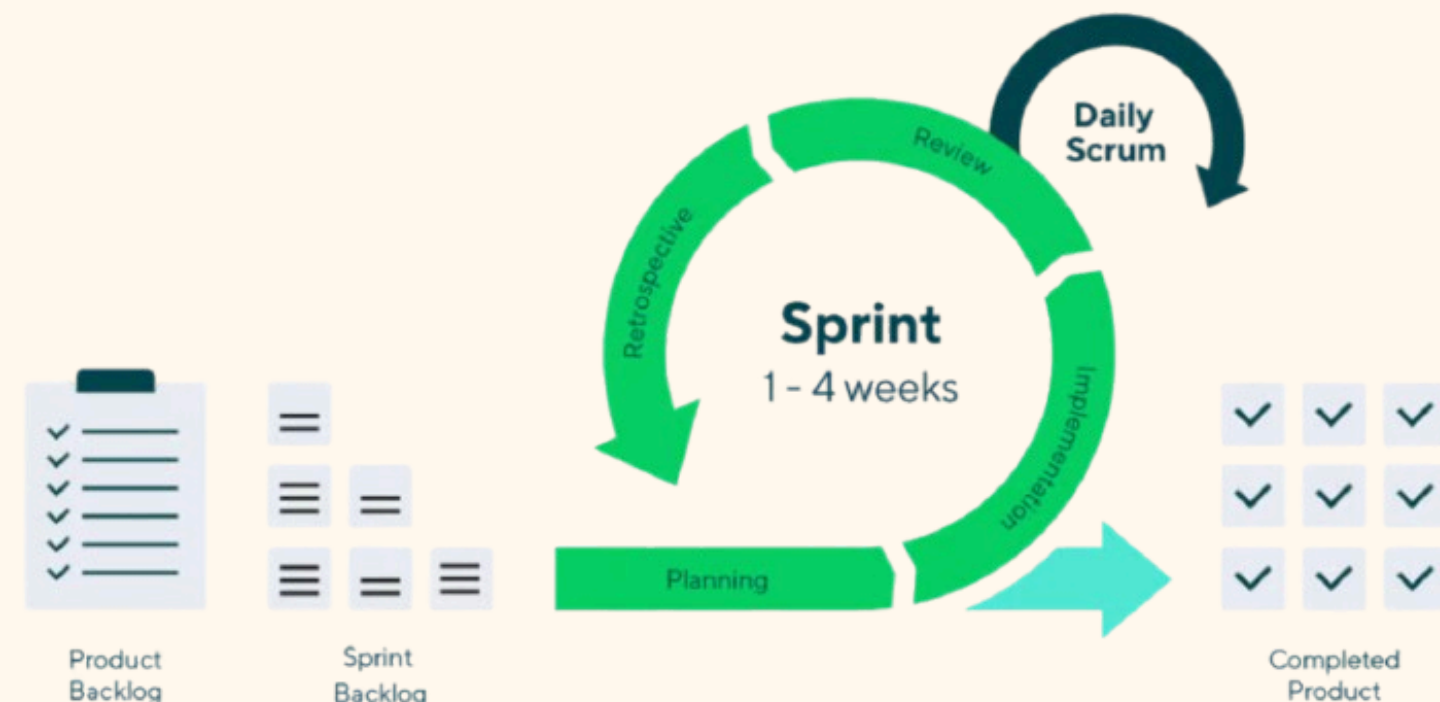
- Arquitectura MVC: separación clara de capas
- Buenas prácticas: validaciones, errores, seguridad

🧪 Estrategia de Pruebas

- Pruebas manuales funcionales por sprint
- Verificación con criterios de aceptación
- Registro y corrección inmediata de errores

✅ Control de Calidad – SonarQube

- Integrado al pipeline de QA
- Detecta: código duplicado, bugs, vulnerabilidades, code smells
- Mejoró la calidad estructural del proyecto



Arquitectura, Mantenimiento y Escalabilidad

🕸 Patrón Arquitectónico: MVC (ASP.NET Core)

- Modelo: Lógica de negocio y entidades persistentes.
- Vista: Interfaz visual con Razor Pages, HTML, Bootstrap.
- Controlador: Maneja la interacción entre modelo y vista.

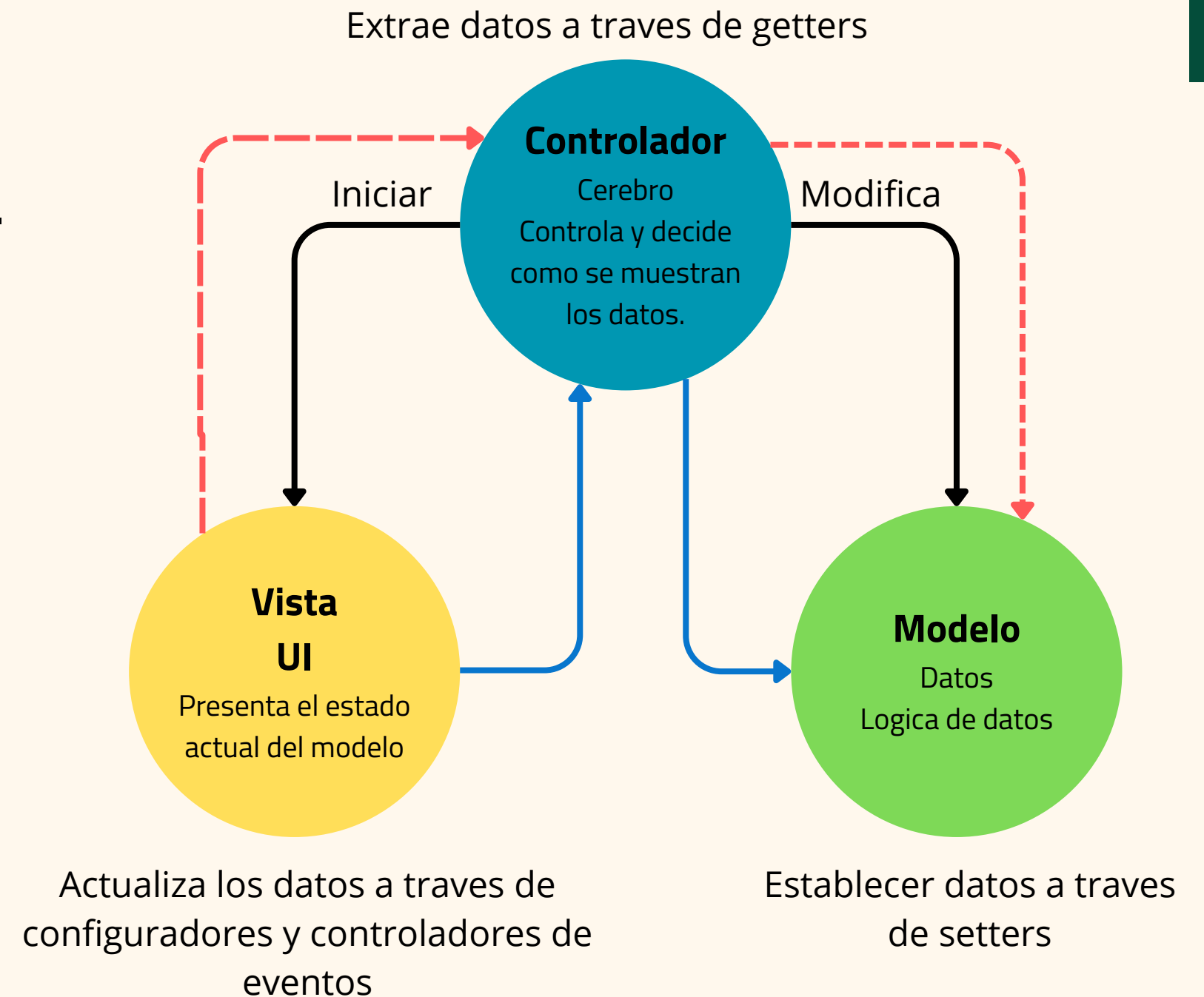
🧩 Estructura en Capas (Proyectos Separados)

◆ Proyecto LicitAR.Core

- Entidades: Licitación, Proveedor, Oferta.
- Managers: Lógica de negocio (ej. CrearLicitacion()).
- DbContext: Manejo de base de datos con EF Core.

◆ Proyecto LicitAR.Web

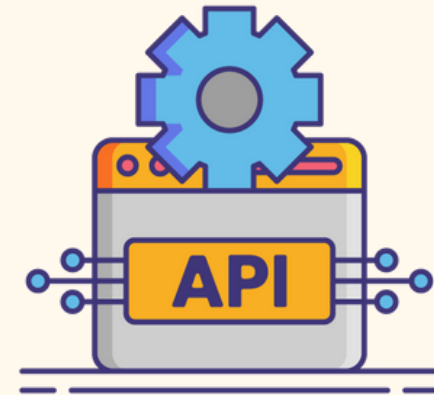
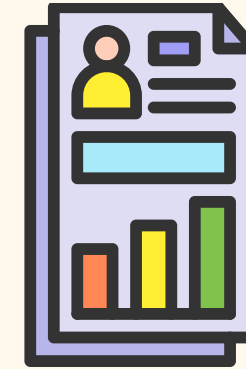
- Controllers: Gestionan solicitudes HTTP.
- ViewModels: Adaptan datos para la vista.
- Vistas: Razor + Bootstrap, responsivas y dinámicas.



Arquitectura, Mantenimiento y Escalabilidad

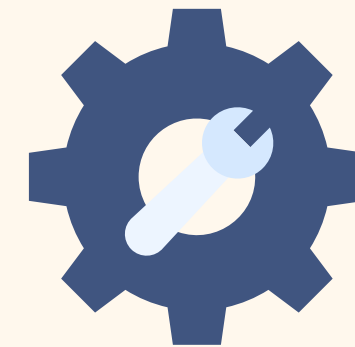
Escalabilidad

- Nuevas funciones sin romper el sistema.
- Core reutilizable en otras interfaces (API, mobile).
- Posible escalado horizontal para mayor rendimiento.



Mantenimiento

- Detección y corrección de errores más simple.
- Refactors seguros y localizados.
- Trabajo en equipo simultáneo por capas (frontend, backend, lógica).



Beneficios Clave

- Separación de responsabilidades (principio SOLID).
- Reutilización de lógica y componentes.
- Mejor mantenibilidad, escalabilidad y testeo.
- Bajo riesgo en cambios futuros



Accesibilidad Web

El proyecto implementa buenas prácticas básicas de accesibilidad web, como etiquetas adecuadas y navegación por teclado. Se usaron tecnologías compatibles con futuras mejoras, priorizando la usabilidad. A futuro, se busca cumplir con estándares internacionales (WCAG) y normativas locales para una experiencia más inclusiva.

Situación Actual

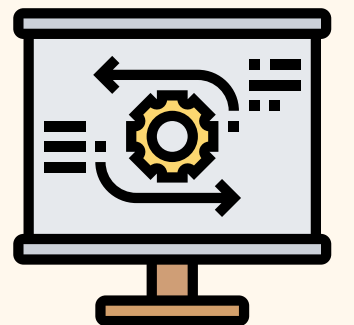
- Imágenes con alt, botones con aria-label, formularios bien etiquetados.
- Navegación por teclado y estructura semántica para lectores de pantalla.

Tecnología y Diseño

- ASP.NET MVC + Razor + Bootstrap.
- Diseño mobile-first con soporte básico de accesibilidad.

Futuro

- Implementar pautas WCAG y normativa nacional.
- Mejorar contraste, validaciones accesibles y compatibilidad total con lectores de pantalla.



Evidencia de Minutas y Avances

Relevamientos y planificación

Sesiones periódicas con cliente y equipo, bajo metodología Scrum.

Documentación centralizada

Confluence: minutas, historias de usuario, reportes, evidencias.

Gestión de tareas

Jira: backlog, sprints, trazabilidad completa.

Control de versiones

Commits, cambios y seguimiento en el repositorio de código.

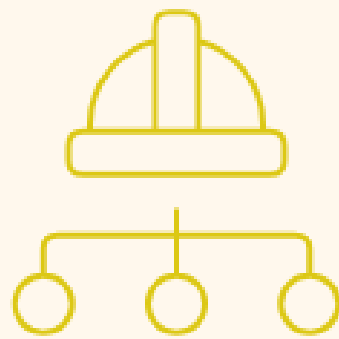
Anexos disponibles

Minutas, tableros y reportes adjuntos en el espacio de trabajo.



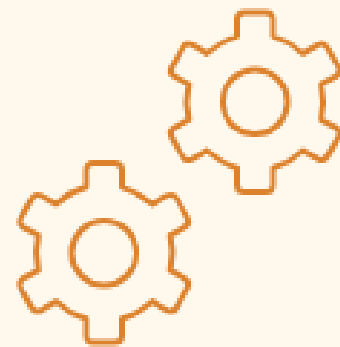
Arquitectura y Tecnologías

Arquitectura de software y tecnologías



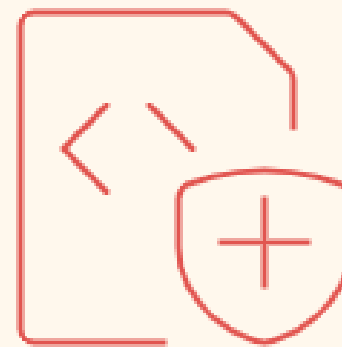
Arquitectura monolítica por capas

El sistema sigue el patrón clásico MVC implementado en ASP.NET Core 9.



Patrón MVC

El patrón MVC se implementa para separar la lógica de presentación (vistas),



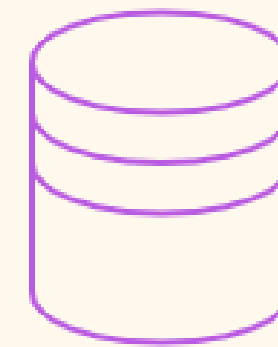
Lenguaje y Framework

Lenguaje C# y framework .NET Core 9, con librerías de soporte adicionales para



Frontend

ASP.NET Core MVC, Bootstrap 5 y jQuery para lograr una interfaz moderna y responsiva.



Base de datos

Azure SQL Database, una solución escalable y gestionada en la nube de Microsoft,

Gestión de Componentes Vulnerables

Se revisó el uso de librerías de terceros en el sistema. Se aplican buenas prácticas como el uso de versiones actualizadas y control desde el entorno de desarrollo. No se detectó uso de componentes obsoletos, y se contempla un plan para actualizaciones futuras y reducción de dependencias innecesarias.

Evaluación:

Durante la auditoría se identificó que LicitAR utiliza herramientas y librerías de terceros (como Bootstrap, jQuery y .NET).

Medidas Implementadas:





- Se verificó el uso de herramientas conocidas y actualizadas.
- Se comprobó el análisis del código fuente con SonarQube.
- Se constató el control de versiones mediante Visual Studio.
- Se detectó la realización de pruebas básicas de seguridad.
- Se validó que solo se incluyeron librerías necesarias, minimizando riesgos.



Comunicaciones Seguras





Se revisaron las medidas de seguridad en comunicaciones y código. La app usa HTTPS, base de datos aislada y gestión segura de secretos. El código se distribuye compilado, sin protecciones avanzadas como ofuscación.

Comunicaciones Seguras

-  Uso de HTTPS con certificado SSL emitido por Azure.
-  Base de datos alojada en red privada (VNet), sin exposición pública.
-  Almacenamiento seguro de claves y contraseñas.
-  Protección contra CSRF y XSS implementada en formularios.



Protección del Código

-  El sistema esta desarrollado Código en .NET compilado se transforma el código para que no se vea el original.
-  Despliegue al servidor solo como artefacto compilado (.dll), sin exposición del código fuente.
-  No se aplica ofuscación ni firma digital actualmente, el codigo esta seguro se ejecuta solo en Azure.
-  No cuenta APIs públicas, así que la superficie expuesta mínima.



Almacenamiento de la Información

Se revisa cómo se almacena y protege la información en la aplicación LicitAr, evaluando la infraestructura en Azure, la seguridad de los datos, los respaldos y la separación entre entornos.

✓ Evaluación de Controles Implementados

- ☁ Alojamiento y Plataforma
 - ✓ LicitAR se aloja en Azure App Service plataforma tipo (PaaS), con alta disponibilidad y escalabilidad.
 - ✓ App y base de datos en el mismo entorno para facilitar la gestión.

📦 Base de Datos

- ✓ Los datos se guardan Azure SQL Database con respaldo automático y alta disponibilidad.
- ✓ Servicio gestionado con redundancia y alta disponibilidad.

🔒 Controles de Seguridad y Respaldo

- ✓ Toda la información es cifrada al guardarse y al transmitirse.
- ✓ Azure garantiza copias automáticas y protección activa de datos.

🧩 Segmentación de Ambientes

- ✓ Se utilizaron Entornos separados: Desarrollo, Pruebas y Producción.
- ✓ Accesos diferenciados por rol para mayor seguridad.

Auditoría Infraestructura

Revisión de Infraestructura en Azure

Hosting, Servicios, Licencias e Infraestructura

- ✓ Alojamiento utilizado en App Service Plan F1 (Free) bajo **sistema Windows**. (Azure Subscription 1)
- ✓ La base de datos se aloja en Azure SQL Database, dentro del mismo grupo de recursos (LicitAR_RG), con un límite de hasta 5 GB de almacenamiento.
- ⚠ Plan F1 presenta limitaciones operativas para el App Service: 1 GB disco, 1 GB RAM, 60 min CPU/día compartida. No apto producción o alta disponibilidad.
- ✓ Las direcciones IP son dinámicas y asignadas por Azure (sin configuración personalizada).
- ✓ La infraestructura opera bajo plan gratuito, sin costos de licencias adicionales.

Gestión de Usuarios

El sistema permite que los usuarios se registren automáticamente como Oferente, que pueden ver licitaciones y enviar ofertas. Solo el Administrador puede crear Entidades Licitantes y gestionar usuarios y roles.

Políticas de Uso

El sistema aplica normas para un uso seguro: confidencialidad de credenciales, prohibición de usos indebidos y mayor responsabilidad para Administradores y Entidades Licitantes.

Registros de Actividad

Se registran eventos clave como inicios de sesión, presentación de ofertas, acciones administrativas y gestión de licitaciones. Los logs están disponibles para el Administrador.

Monitoreo de Actividad

Monitoreo básico con logs, auditoría ante incidentes o fiscalización, Se almacena la última vez que el usuario inició sesión, incluyendo dirección IP, fecha, hora y navegador utilizado.

Manejo de Roles

Roles: Administrador, Entidad Licitante, Oferente, Auditor; solo el rol de administrador gestiona usuarios y asigna roles sensibles; Cada rol tiene permisos específicos según su función.



Política de contraseñas

Reglas y medidas de seguridad implementadas:

Política de Contraseñas

Requisitos:

- Mínimo 10 y máximo 100 caracteres
- Al menos: 1 mayúscula, 1 minúscula, 2 dígitos y 1 carácter especial

Expiración de Contraseñas

- No configurada en esta etapa

Almacenamiento Seguro

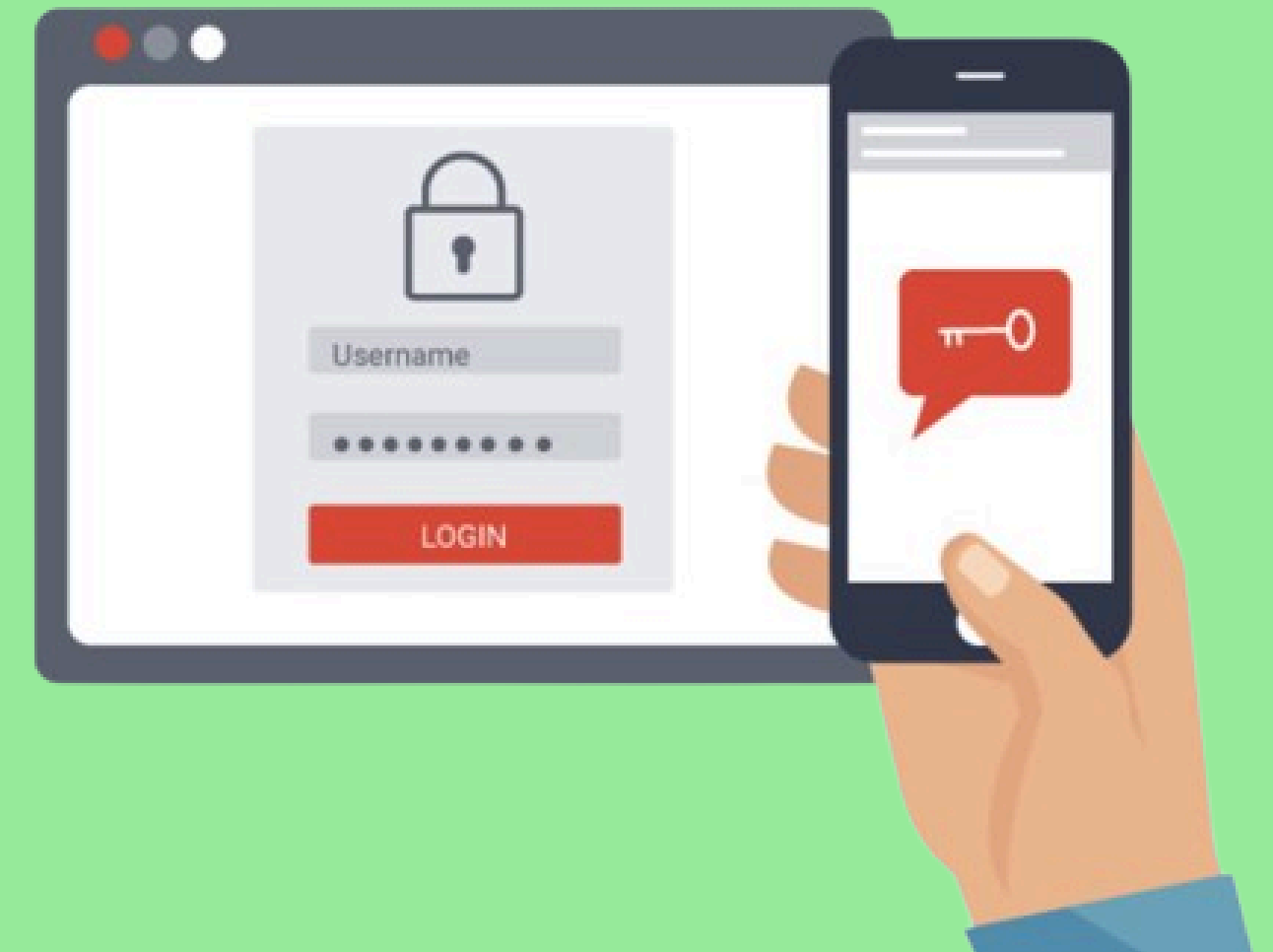
- Contraseñas cifradas con SHA-256

Doble Factor de Autenticación (2FA)

- No implementado aún
- Previsto para futuras versiones

Asignación y Recuperación

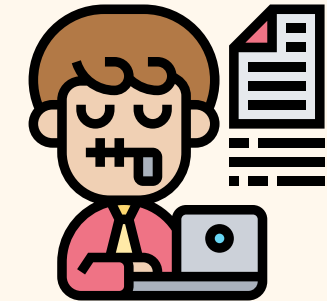
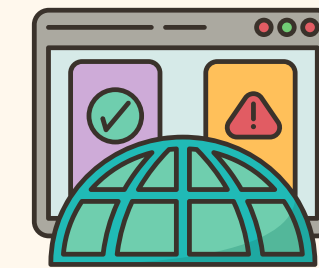
- Registro manual por Administrador con email válido
- Recuperación por autogestión mediante validación de email activo



Privacidad y Seguridad

Acceso y Tipo de Aplicación

Aplicación web, accesible desde navegadores en PC y móviles.

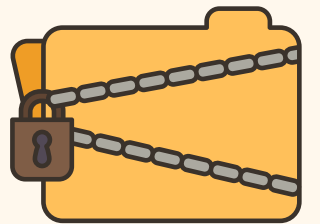


Datos Recopilados

CUIT, razón social, domicilios y datos de proveedores (personas físicas o jurídicas).

Tratamiento de Datos

No considerados sensibles según Ley 25.326, pero tratados como datos personales con seguridad y confidencialidad.

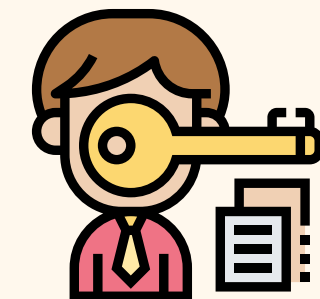


Consentimiento

No se solicita consentimiento explícito (datos voluntarios), no se comparten con terceros ni usan APIs externas.

Medidas Técnicas de Seguridad

Autenticación por usuario/contraseña, HTTPS, control de roles.



Futuro

Planea política de privacidad accesible en interfaz para reforzar cumplimiento legal.

Cumplimiento Legal

Se ajusta a principios de legalidad, finalidad, consentimiento, seguridad y confidencialidad de la Ley de Protección de Datos Personales.

Metodología y Versionado

Se utiliza Git con repositorio en GitHub, aplicando buenas prácticas como commits trazables, revisiones por pull request y estrategia de ramas clara (dev para desarrollo, main para versión estable). Esto permite control de cambios, trazabilidad y auditoría efectiva del historial de código.

Metodología y Software para Versionado

Uso de Git distribuido, gestionado en GitHub.

Prácticas colaborativas

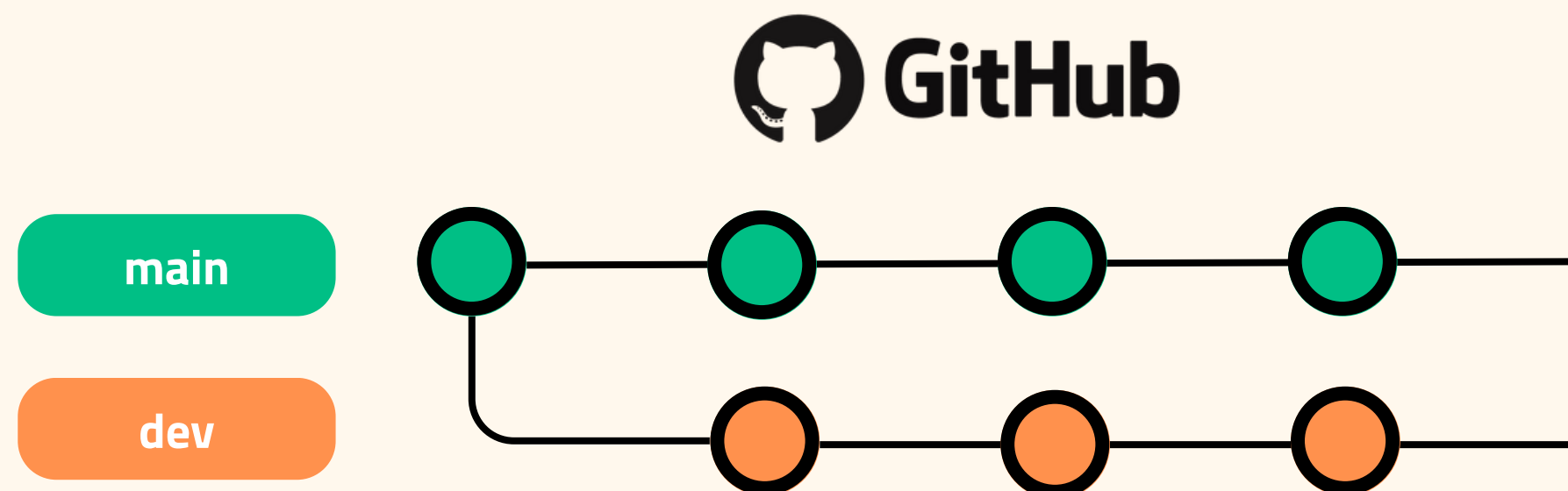
Commits frecuentes y descriptivos, pull requests para revisión, code reviews.

Estrategia de ramas

dev para desarrollo activo (Azure App Service dev), main para versión estable (Azure App Service QA).

Beneficios

Trazabilidad, control de cambios, colaboración eficiente.



Protección de ataques externos

Protección frente a ataques externos

Desplegada en Azure App Service, plataforma con seguridad integrada.

Antimalware y antivirus

Protección automática con Microsoft Defender para OS, evita ejecución de código malicioso.

Actualizaciones automáticas

Parches y actualizaciones aplicados automáticamente por Microsoft.

Entorno controlado

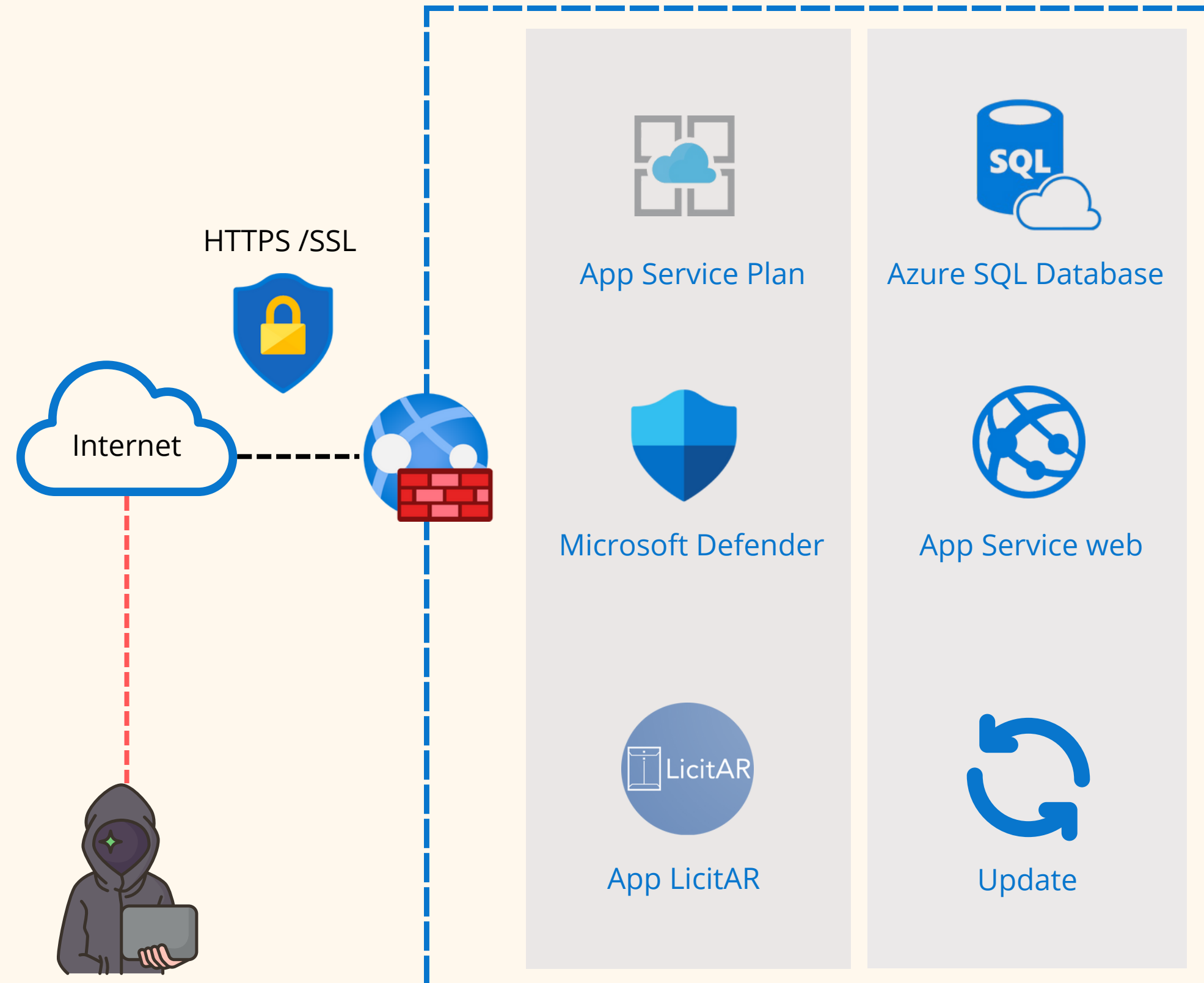
Plataforma PaaS, sin gestión directa del SO por el equipo, menor superficie de ataque.

Otras medidas

Conexiones protegidas con SSL/TLS, tokens antiforgery contra CSRF, codificación segura para mitigar XSS e inyecciones.

Conclusión

Protección sólida sin necesidad de antivirus adicionales en la aplicación.

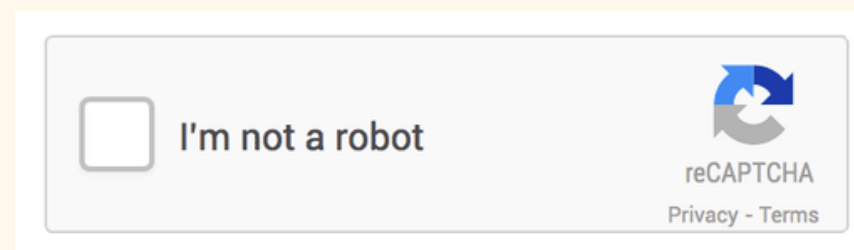


Análisis de Riesgos

Riesgos	Nivel	Nivel	Nivel	Mitigación
Pérdida de acceso a la cuenta de Azure	Media	Alta	Alta	Uso de cuentas compartidas seguras, respaldo de credenciales en lugar seguro. Monitoreo de consumo, migración anticipada a un plan pago si es necesario. Pruebas antes del despliegue, uso de entornos de staging. Copias de seguridad automáticas habilitadas, restauración punto en el tiempo Capacitación básica, documentación clara y asistencia de tutoriales oficiales Verificación periódica del estado de los backups y restauraciones de prueba. Uso de control de versiones (GitHub) y documentación constante.
Superación del límite del plan gratuito (F1)	Alta	Media	Alta	
Fallas en la implementación o despliegue de la aplicación	Media	Media	Media	
Errores en la base de datos o pérdida de datos	Baja	Alta	Media	
Falta de experiencia técnica del equipo	Alta	Media	Alta	
Problemas con los backups (no restaurables, error en cofre)	Baja	Alta	Media	
Cambios no documentados o mal control de versiones	Media	Media	Media	

Uso de CAPTCHA

- ✗ Actualmente no hay CAPTCHA para validar usuarios humanos en formularios.
- 🚀 Planificado para futuras versiones en puntos críticos para evitar bots y abusos.
- 🗝 Autenticación multifactor (MFA) prevista para usuarios con permisos elevados o acceso sensible.
- 🛣 Estas mejoras forman parte del roadmap de seguridad para entornos productivos.
- ⚙ Actualmente, el sistema está en etapa de desarrollo y demostración.





Gracias

