ANALISIS TÉCNICO HERRAMIENTA COOPERAJUS

Versión 2.0

Joaquin Franco
jfranco@circuitosaljarafe.com
27 de Julio de 2,022

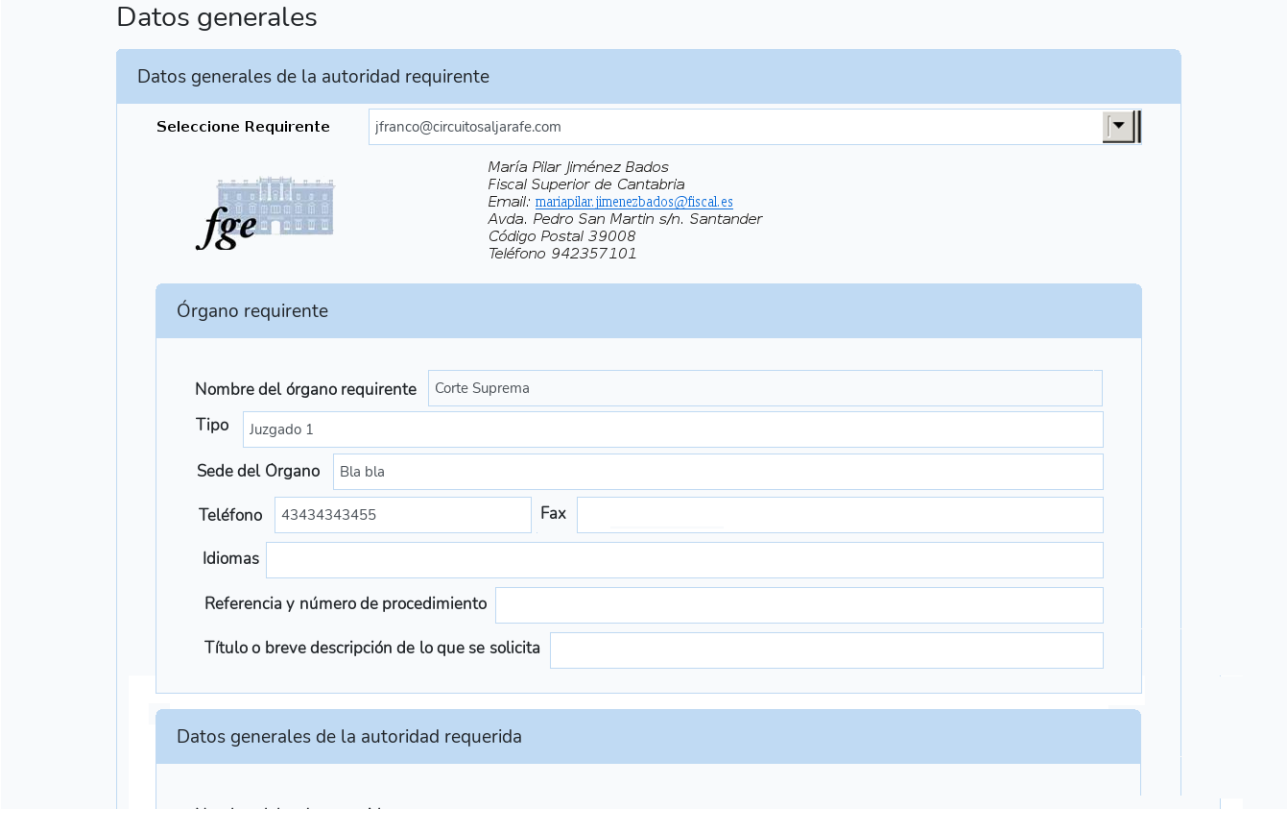
1. UTILIDAD

La usabilidad es muy alta, presentando un esquema muy bien desarrollado para la recogida de datos, simple y fácil de aprender y usar por personal no técnico.

2. EXPERIENCIA DE USUARIO

En éste apartado analizaremos la usabilidad y confortabilidad de la web.
En primer lugar se nota muy poco aprovechamiento del espacio de trabajo, con amplios márgenes que si bien contribuyen a la claridad del contenido en caso excesivo resultan en páginas muy largas con muchos pasos a realizar y con scrolling vertical para navegar por el documento contínuamente.

Veamos un ejemplo de compactación sin detrimento de la claridad en éste diseño:



En la propuesta de diseño o layout se ha reducido el scrolling vertical, se ha considerado que los organismos requirentes pueden ser preconfigurados y salvados en la base de datos, de forma que no haya que rellenar repetitivamente todos los campos.

Dado que existe mucho espacio en horizontal se ha optado por situar los títulos de los campos a la izquierda de éstos, reduciendo la ocupación vertical.

Con ello se logra no solo el reducir el scrolling vertical en la navegación sino posiblemente reducir de 6 a 4 pasos para completar el pedido.

Por otro lado, los navegadores disponen de una utilidad que no ha sido aprovechada. Se trata de que cuando un formulario en una misma URL tiene el mismo nombre de campo nuestro navegador memoriza los datos introducidos anteriormente, de forma que haciendo doble click sobre el campo nos aparecen los datos anteriores sin tener que escribirlos de nuevo.

En cuanto a los textos de ayuda, estos pueden ser introducidos en el mismo campo en gris, en vez de ocupar nuevas líneas en el documento.

Al escribir sobre dichos campos el texto de ayuda desaparecerá para ser reemplazado por los datos introducidos.

Otra opción a evitar ocupar espacio en el navegador con ayudas es crear botones de tipo “?” que situando el ratón sobre ellos nos aparecerá un emergente con toda la ayuda necesaria.

Finalmente el diseño carece de la finura como estilos con bordes redondeados, sobras, línas de separación, iconos y otros detalles que lo hagan visualmente atractivo.

3. TECNOLOGÍA DE LA HERRAMIENTA

Existen diferentes “frameworks” o marcos de trabajo para desarrollar páginas web de una forma rápida, en éste caso se ha usado el framework Laravel que trabaja sobre el intérprete PHP de código.

Si bien los frameworks ahorran muchas horas en desarrollo hay que tener en cuenta que:

1. El programador no conoce en profundidad las utilidades usadas, como son plugins o módulos de framework, lo cual no solo dificulta su mantenimiento sino que añade dependencias externas y posibles inyecciones de vulnerabilidades.

2. El framework usar módulos genéricos, por ejemplo una hoja de estilos CSS incluye cientos o miles de reglas de diseño, cuando solo usamos unas pocas. Esto obliga al usuario a bajarse un fichero de estilos que en algunos casos tiene un tamaño excesivo, ralentizando el rendimiento. Para éste tipo de diseños es conveniente usar un CSS creado específicamente o bien usar los estilos en modo in-line (incluidos en el propio HTML en vez de un fichero CSS externo)

3. El framework incluye módulos en lenguaje PHP para la conversión de formatos. Existen módulos a bajo nivel programados en otros lenguajes para hacer lo mismo , por ejemplo crear un PDF , que tienen un rendimiento notablemente mayor, con lo cual no solo se reduce la carga del servidor sino además se acelera el proceso y mejora la experiencia de usuario.

4. El desarrollo sobre PHP hay que tener en cuenta que es un lenguaje de código interpretado con un rendimiento muy inferior a un desarrollo realizado en lenguajes también interpretados como Perl o Python.

Esto no sería un grave problema si no hubiera que añadir que además según fuentes de NIST.gob e INCIBE PHP llega a presentar hasta más de 4000 vulnerabilidades en un solo año, frente por ejemplo a Perl que presenta menos de 400.
Requiere por lo tanto un cuidadoso trabajo a la hora no solo de elegir la versión de PHP a usar, sino además de cómo programamos el código para evitar fallos de seguridad.
En éste punto hay que tener en cuenta que gran parte del código del framework que trabaja sobre PHP es de terceras partes, y en gran medida desconocido por el desarrollador, con lo cual pueden ser fuente de vulnerabilidades.
Por último PHP a lo largo de su historia ha presentado graves problemas de compatibilidad y escalabilidad, por ejemplo renombrar funciones sin mantener la compatibilidad hacia abajo, lo cual ha causado que actualizar a versiones recomendadas de PHP implicará que todo el código desarrollado dejase de funcionar.
Un caso típico es cambiar el nombre de la función “split” (que es un nombre bastante común en todos los lenguajes de programación) por “str-split” con lo que todas las funciones split de nuestro código dejaban de funcionar.
Otro caso ha sido la supresión de multi-curl , que era usado para ataques a terceros y que ha sido eliminado en las nuevas versiones de PHP, con lo cual los sitios que requerían multi-curl comenzaban a fallar en su funcionamiento.
La solución ideal a todas estas soluciones es desarrollar en lo que se conoce como “código limpio”, aunque suponga mayor inversión de tiempo y conocimientos técnicos por parte de los desarrolladores.
En éste aspecto hay que diferenciar el perfil de diseñador web, quien puede usar cualquiera de éstos frameworks para facilitar su trabajo, del perfil programador web, quien realmente hace código.

5. Para sitios que requieran robustez, seguridad y escalabilidad es preferible usar otros entornos no basados en PHP, si bien ésto se traduce en muchas mas horas de desarrollo, se obtiene un sistema mas fácil de mantener, amigable, conocido y robusto en todos sus aspectos.

6. Utilización de código JavaScript.
Para el funcionamiento de la web se requiere el uso de JavaScript por parte del navegador. Éstos códigos en JavaScript son bajados desde el servidor hasta el ordenador del usuario y ejecutados en éste último, por lo tanto consumen recursos y establecen dependencias de rendimiento según el equipo del usuario.
Se debe reducir al máximo el uso de JavaScript en el lado del cliente, reemplazándolo por funciones bajo filosofía RESS (Responsive Server Side) donde sea el servidor quien haga el proceso en su mayoría.

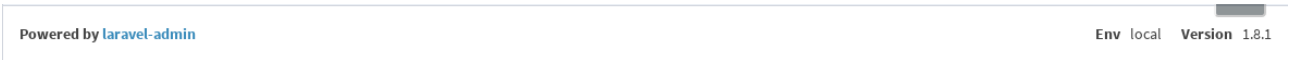
4. OPERATIVA

En general está muy conseguida, pero hay botones como “Validar” que sobrarían explicaciones y actuaciones necesarias por el usuario.
Al completar el formulario se debería hacer cada vez una pre-validación automática sin intervención del usuario solicitándole un nuevo click.
En el caso de que la pre-validación arrojase algún error éste ser notificado y mostrado con claridad para su corrección, y en el caso que la pre-validación sea correcta simplemente no mostrar nada que distraiga al usuario y continuar silenciosamente al siguiente paso.

5. INCLUSIONES DE CÓDIGO

Se observa llamadas a terceros sitios para el correcto funcionamiento de la web, ésto debe ser evitado. El caso observado es la descarga de fonts desde Google.
Supongamos que por cualquier razón la red del cliente tenga bloqueado o restringido el acceso a dichas direcciones externas, en cuyo caso no funcionaría.
No se debería nunca provocar que el navegador del cliente baje contenidos o códigos de terceros sitios.

Se facilita información sobre la plataforma o entorno usada para la aplicación:



Esto puede constituir un factor de riesgo adicional facilitando información útil para un atacante, ya que le bastará investigar sobre posibles vulnerabilidades de la versión usada para iniciar un posible ataque.

6. OTRAS VERIFICACIONES TÉCNICAS

Se observa que se dispone de un fichero genérico de Laravel para la configuración, donde por ejemplo se configura nuestro servidor SMTP, pero no se dispone de ningún check del mismo, esto es si responde a nuestra conexión, si presenta RDNS correctamente (necesario para que muchos servidores de destino acepten nuestros emails) o inclusive si está en alguna lista negra que pudiera restringir la entrega con éxito del correo electrónico.

En éste apartado, podría ser interesante por motivos de seguridad en la red entre el usuario y el servidor forzar a que se requiera una conexión SSL siempre, configurando el navegador para que siempre realice una redirección desde HTTP (port 80) a HTTPS (port 443)
Con ello además se simplificarían las tareas de los técnicos del país que realiza la implantación, reduciéndola a simplemente sustituir el Certificado sin necesidad de editar la configuración del servidor web Apache.

Durante las instalaciones realizadas hasta el momento en 6 paises, se ha detectado que tres de ellos solicitaron posible integración con otras herramientas de las que ya disponen.

Sería una mejora importante considerar no solo la exportación a un PDF o Word intermedio, sino la posibilidad de realizar la firma electrónica a través de la misma web, ya sea enlazando con aplicaciones locales de firma y certificado o incorporando dichas funciones en el código de la herramienta, así como un modelado de exportación de datos.

Este modelado de exportación de datos debería permitir exportar y mapear a sistemas externos mediante algunos de los formatos y procedimientos mas estándares posibles como son exportación en fichero CSV (Comma Separated Values), XML o REST.
La carencia de exportación en otros formatos y estandares ha constituido para algunos países una sería objeción al uso de la herramienta en tanto no podían integrarla con sus sistemas actuales.

Otro aspecto detectado es que no permite la exportación en PDF/Word en lenguajes que no sea español, para lo cual se propuso un par de soluciones a los módulos de código encargados de esas funciones, resultando relativamente fácil de resolver a nivel técnico y facilitando el trabajo a paises con diferente idioma como el caso de Brasil.

Finalmente y no menos importante, la herramienta no dispone de ningún sistema de autenticación del usuario, delegando ésta en los administradores de red y en muchos casos a usar reglas en sus cortafuegos. Debería disponer de su propio sistema de creación/edición de usuarios autorizados y la entrada a la herramienta debería requerir algún tipo de autenticación, aunque fuese la mas simple por medio de usuario y clave.

Los sistemas de autenticación deberían además incluir roles, por ejemplo si un usuario puede o no acceder a cartas rogatorias iniciadas por otro usuario o departamento, qué usuarios tienen permisos para agregar o modificar Convenios, etc.

Desde Brasil nos reportan que el sistema de autenticación y acceso de la herramienta falla, permitiendo el acceso con cualquier usuario y clave.

En cumplimiento de la trazabilidad necesaria impuesta por los Sistemas de Seguridad de la Información según ISO 27001 se hace necesario implementar trazabilidad del uso del software a fin de garantizar dicha seguridad, lo que requiere registro en el servidor mediante accesos únicos que identifiquen las acciones efectuadas con la herramienta por cada usuario.

La norma ISO 27001 es de obligado cumplimiento para cualquier centro de datos y debe cubrir los aspectos de integridad, accesibilidad, seguridad y legal de cualquier software a instalar en los sistemas.

En el caso de España, el Esquema Nacional de Seguridad (ENS) recomienda el uso siempre de software con soporte vivo del fabricante.
Desconocemos específicamente otras normativas y regulaciones en los paises objeto de éste proyecto, pero en general lo que se pretende es que el software entregado por cualquier fabricante o entidad conlleve soportes tanto correctivos, preventivos y evolutivos por parte del fabricante del mismo.

En la instalación en Panamá , se detectó por primera vez un arreglo necesario en el kernel del equipo para validar los drivers de discos duros con su versión Vmware EsXI 6, donde el fallo se

produjo por usar en nuestro kernel por defecto de la máquina virtual drivers abiertos que no fueron reconocidos por el host.
Esto posiblemente fuese reparable por el administrador de Vmware, esto es el entorno de virtualización usado, no obstante en previsión de que pudieran a lo largo del proyecto surgir situaciones como ésta, el entregable que descargaban para construir la máquina virtual con la herramienta CooperaJUS contenía ya otro kernel alternativo.
En el caso de Panamá se completó con éxito la instalación con la versión ya incluida del kernel alternativo sin mayor problema.
El error que se producía era un Kernel Panic al bootear el servidor con indicación de que no podía montar la partición raiz por ser desconocida.

La futura solución a éste caso, para poder usar siempre el kernel optimizado que se creó para el proyecto y no el alternativo, requiere de una instalación piloto con diferentes versiones de virtualización (Vmware EsXI. Hyper-V, otras) a nivel laboratorio.

7. LEGAL

No se ha incluido ningún marco de Licencia bajo la que opera y se distribuye el software.
Este marco debería especificar las garantías si existen, los derechos a uso, modificación y distribución del mismo bajo alguna de las licencias para código abierto existentes.

El software completo del servidor usa licencias GNU (http://www.gnu.org) y Apache.
El kernel puede incluir (no recomendado) código tipo “blob” de algún fabricante para algún drivers, estos son típicamente las interfaces de red RealTek e interfaces gráficas Nvidia, clasificado como non-free (esto es, el fabricante entrega un firmware sin facilitar su código fuente).

El uso de licencias non-free está sujeto a la necesidad por la arquitectura de servidor hardware y los drivers necesarios, por lo que en un sistema totalmente Open Source deberian evitarse estos fabricantes de hardware o hacer uso de drivers genéricos bajo la capa del hipervisor.

8. RECOMENDACIONES FINALES

1. Crear la herramienta con software de calidad y código limpio sin frameworks para diseñadores, realizada por programadores de código.

2. Mejorar la Experiencia de Usuario en el uso de la herramienta mediante una navegación mas ágil

3. Proporcionar salida en otros formatos aparte de PDF/Word que faciliten la integración de la herramienta con otros sistemas del cliente.

4. Definir y aplicar los criterios de Seguridad de la Información, tanto a nivel de accesos como de transmisión de datos.

5. Proporcionar a la herramienta de métodos de importación tanto de las fichas, como de las cartas rogatorias. No existe ningún procedimiento rápido para importar una carta rogatoria recibida y poderla reutilizar

6. Proporcionar a la herramienta las funcionalidades multilenguaje reales.

7. Eliminar el entregable como “contenido web” o sea carpetas de un directorio de servidor web sin incluir todo lo que necesita (interpretes de código, firewall, servidor web, posiblemente servidor de correo electrónico.

9. Generar a partir del modelo usado que ha sido una imagen 100% operativa y funcional del servidor completo una nueva versión mas amigable de instalar sin requerir conocimientos técnicos por parte del cliente, incluyendo menú de instalación gráfico donde poder definir los parámetros de red (la IP, etc) del servidor.

10. Forzar a un modelo securizado , solicitando Certificado (aunque sea privado sin ninguna CA) para trabajar siempre en HTTPS

11. Convertir el entregable a un modelo FDE (Full Disk Encrypt) protegiendo la información de posibles substracciones o copias de volumenes en el centro de datos, o de robo de documentos a través de su acceso desde ordenadores o dispositivos móviles.

12. Establecer un mecanismo de firma digital de documentos.
Existe la dificultad de que no todos los países usan y reconocen los mismos, pero sería suficiente que se usara cualquiera de ellos y sea la misma herramienta la que valida contra un método único.

13. Establecer mecanismos de auto-actualizaciones de versiones del software.

14. Limitar el uso de posibles métodos conocidos como vulnerables y mantener un registro de vulnerabilidades de entorno, no de la aplicación en sí misma, como pueden ser las recientes vulnerabilidades muy graves notificadas por el INCIBE sobre accesos remotos con protocolos RDP de MS Windows.