FactoríaBIZ

Evaluación FactoríaBIZ

Joshua Sangareau Quesada

Fase 1: Evaluación técnica mediante cuestionario

El objetivo de esta fase es hacernos la idea del conocimiento que tienes sobre algunos aspectos básicos. Ninguna pregunta necesita una explicación o desarrollo extenso, con unas pocas líneas debe ser suficiente. Si no sabes responder alguna, puedes indicarlo o dejarla en blanco. Copia esta fase en un email respondiendo a frueda@factoriabiz.com con tus respuestas.

Conocimiento de Internet

- Explica las partes de las que se compone una URL como "https://juaj:xcv@clientes.factoriabiz.com:8080/proyectos/ingenieria
  - https: Es el protocolo con el que se van a comunicar cliente-servidor.
  - juaj:xcv@ -> Usuario y contraseña (en ese orden) con la que se va a autenticar el usuario en el acceso al servidor.
  - clientes.factoriabiz.com IP o dominio en este caso (clientes->subdominio, factoriabiz->nombre del dominio). Es la localización donde está alojado el servidor
  - :8080 es el puerto del servidor en el cual se va a conectar el cliente para interactuar.
  - /proyectos/ingenieria es la ubicación específica de un recurso en el servidor.

- ¿Qué es HTTP y HTTPS? ¿Por qué es importante utilizar HTTPS? Pon un ejemplo. ¿Cuándo es viable usar HTTP?

  HTTP y HTTPS son protocolos de comunicación entre el cliente y el servidor. HTTPS es la versión segura y cifrada de HTTP. Se usa HTTPS cuando en la conexión se manejen datos sensibles como usuario y contraseña. Los datos serán cifrados para la comunicación. Es recomendable usar HTTPS siempre que se pueda.

  Se puede usar HTTP en el caso de no tener información delicada en la comunicación o en el manejo de información de forma local.

- ¿Qué son los puertos y cuál es su importancia en las conexiones? ¿Qué entiendes por DNS?

  Los puertos son las "puertas" por las que se va a comunicar un servicio. Dan la posibilidad de estar realizando distintas comunicaciones a la vez.

  DNS es un sistema que traduce el nombre de dominios a las direcciones IP. Por ejemplo al hacer ping a www.factoriabiz.com se recibe respuesta de la IP: 82.98.175.116 indicando así la IP asignada a ese nombre de dominio.

- ¿Cómo funciona un servidor de DNS? ¿Qué es un dominio y cómo se relaciona con un servidor ISP?

  Al escribir un nombre de dominio es enviado al servidor DNS que trata de traducir el nombre del dominio y devolver la IP como respuesta.

  El dominio es el nombre único que se le ha dado a una dirección web. Por ejemplo factoriabiz.com

  ISP no sé qué es o cómo funciona.

- ¿Has trabajado con servidores? Explica brevemente tu experiencia.

  Si he creado servicios de servidor sencillos tanto en Python como en Java, recibiendo y mandando información. Esta parte la estamos aprendiendo justamente ahora en el grado superior (DAM) que estoy cursando en la actualidad. Ahora mismo ha sido muy básico lo que he trabajado pero seguiremos ampliando conocimientos en la 2ª evaluación de dicho curso.

  También hemos creado una máquina virtual en Oracle para utilizarla como servidor de un CRM-ERP y para poder ejecutar nuestros programas de servidores de Java y Python.

- Explica en qué consiste y para que se utiliza el protocolo SSH.

  SSH es un protocolo que permite la conexión de un equipo a otro por un canal seguro. Utiliza un cifrado para proteger la conexión.

- Qué puedo hacer si tengo un cliente que me pide montar la web que acabo de terminar de programar en sudominio.com sustituyendo su antigua web en un proveedor de hosting que no es compatible con el sistema de desarrollo actual. Dime qué opciones podemos darle al cliente para gestionar lo que necesita y qué operaciones hay que controlar o coordinar con el cliente.

  No sé responder a esta pregunta. El curso que estoy realizando (DAM) no está tan orientado a web por lo que no hemos estudiado este tipo de problemas.

- Si recibes un error 404 al navegar a una página web, ¿qué significa y qué podrías verificar para solucionarlo?

  El error 404 suele surgir cuando se está consultando una información o recurso que no existe en el servidor.

  Se puede verificar que la dirección url esté correctamente escrita, verificar la conexión a internet o recargar la página para descartar posibles errores temporales.

  Si el error es por parte del servidor probablemente haya un error de estructura en la ubicación de los recursos.

- Imagina que estás configurando un sitio web y necesitas habilitar un certificado SSL. ¿Qué pasos ejecutarías?

  No sé responder a esta pregunta.

- Si un usuario reporta que no puede acceder a un dominio específico, ¿qué pasos seguirías para diagnosticar el problema?

  No sé responder certeramente a esta pregunta, pero yo como usuario comprobaría si el antivirus o el firewall están bloqueando el dominio por alguna razón. También me cercioraría si está bien escrito el dominio al que deseo acceder y probaría desde otro dispositivo a realizar la conexión.

- Explica, que tendrías que hacer respecto al DNS, si eres propietario del dominio midominio.com y quieres crear un nuevo hosting pruebas.midominio.com.

  No sé responder a esta pregunta. No he gestionado todavía ningún dominio.

- ¿Cómo puedes acceder a un servidor de internet mediante SSH de forma segura y sin contraseña?

  En mi caso he realizado este tipo de conexión a mi servidor en Oracle con Filezilla con el protocolo SFTP. También he realizado una conexión ssh con mi servidor a través de la consola con el comando ssh -i “archivo-con-la-contraseña.ssh” “ubuntu”@”direccion IP”

- Explica la diferencia entre el protocolo DNS, la zona DNS y el servidor DNS.

  El protocolo DNS traduce los dominios en una IP para la conexión.

  Zona DNS no se lo que hace.

  El servidor DNS es un sistema que almacena los dominios y responde a las peticiones de servicio de los clientes enviando la dirección IP asignada al dominio recibido.

Conocimiento PHP y Laravel

- Define en tus palabras conceptos como MVC, API y CRUD.

  CRUD: del inglés Create Read Update Delete. Es la denominación que se le da a las funciones básicas que se realizan con una BBDD.

  Pongamos que deseamos gestionar productos.

  Create -> (sentencia INSERT) inserta los datos generados en el producto a la BBDD.

  Read-> (sentencia SELECT) consulta en la BBDD los datos del producto o productos deseados.

  Update-> (sentencia UPDATE) actualiza el dato o datos deseados del producto en la BBDD.

  Delete-> (sentencia DELETE) elimina el dato o el producto entero deseado de la BBDD.

API: Es un conjunto de reglas, acciones y protocolos que se encargan de enlazar una parte del proyecto con otra, una aplicación con otra.

MVC: es el patrón de diseño modelo-vista-controlador. Se compone de:

Modelos-> Es la parte encargada de manejar los datos y definir la lógica del proyecto.

Vistas-> Es la parte visual del proyecto con la que va a interactuar el cliente. Recibe los datos y se encarga de mostrarlos de forma amigable al usuario o cliente.

Controladores-> Es la parte que interactúa entre las vistas y los modelos, recibe la información de las vistas y actualiza los modelos cambiando la información mostrada en la vista.

- ¿Cuál sería un uso práctico de las “facades” en un proyecto Laravel?

  Por ejemplo un uso práctico es en el archivo web.php cuando se configuran las rutas usar el facade Route::resource(‘producto’, ProductoController::class). Por medio del facade Route y su método estático resource accederemos a todo los métodos definidos en la clase ProductoController.

  Ayudan a acceder a las clases del contenedor de servicios de la aplicación y se mantiene el código más limpio no teniendo que instanciar las clases facilitadas por los facades. Facilitan el trabajo del programador/a.

- Si te piden explicar el propósito de un “Service Provider” en Laravel, ¿cómo lo resumirías? ¿sabrías explicar qué problema originaría no usar services providers?

  Sirven para configurar la lógica de los servicios que usará la aplicación, los enmarca y permite usarlos en cualquier punto de la aplicación.

  Los problemas de no usarlos puede ser repetición de código si se usa el mismo servicio en varias partes de la aplicación y falta de escalabilidad en la aplicación.

- ¿Qué harías para organizar un proyecto Laravel de manera eficiente en cuanto a carpetas y recursos?

  Ser consistente en cómo llamar a las variables, carpetas y archivos. Se debe respetar la estructura de carpetas introduciendo cada fichero o directorio en su ruta correcta. Por ejemplo si voy a crear un controlador de producto, crear un archivo llamado ControladorProducto.php en la ruta ./App/http/controllers en la cual se llama a una vista llamada por ejemplo mostrar.blade.php en la ruta ./resources/views/producto y un modelo llamado Producto.php en la ruta ./App/models

- Considera un caso donde una API externa no responde en tu aplicación Laravel, ¿cómo diagnosticarías el problema?

  Es buena práctica comprobar si el API externo está funcionando correctamente o está caído.

  Se pueden ver los errores o excepciones generados por la API en el log. A través de eso se podrá diagnosticar cual es el problema y solucionarlo.

Bloque general

- Describe una situación en la que tengas que organizar un proyecto nuevo. ¿Qué pasos seguirías para entender las herramientas necesarias?

  En mi caso empezaría por analizar el objetivo del proyecto y plantear objetivos a cumplir, como debe funcionar la aplicación y resultados que debe mostrar o realizar el proyecto.

  Analizar el objetivo deseado de la aplicación es muy importante porque desde el resultado de este análisis, se formará la “hoja de rutas” a seguir. Si es un cliente el que ha planteado el proyecto, se debe mantener buena comunicación y dejar muy claras las acciones que debe realizar el proyecto.

  Realizaría un prototipo del proyecto para poder presentar al cliente y debatir posibles cambios o modificaciones.

  Una vez se tenga claro el objetivo y el prototipo aprobado, prepararía la “hoja de rutas”.

La primera parte sería definir con qué herramientas y tecnologías conozco y estoy formado para poder desarrollar el proyecto correctamente. Por ejemplo, que framework usar para el backend como puede ser Laravel, qué tecnología usar para el frontend como Angular y que BBDD se usará como puede ser Mysql.

Tras esto empezaría la fase del desarrollo del proyecto.

- Has terminado tu app web, que haces ahora para ponerla en funcionamiento, brevemente lo que harías.

  Primero configuraría un entorno de producción en un servidor (ya sea local o en la nube) y subiría al servidor el proyecto. Prepararía los datos iniciales que debe tener la BBDD para su correcto funcionamiento. Tras esto se debe hacer pruebas de funcionamiento para ver que todo esté correctamente.

- ¿Sabes lo que significa devops? Explica el término si lo conoces.

  Combina las palabras desarrollo y operaciones, no conozco el término pero creo que es la forma de estructurar el desarrollo con técnicas de trabajo.

- Cómo planificarías las pruebas a realizar en tu software.

  Nunca he realizado pruebas a nivel profesional. Pero he realizado multitud de códigos o programas los cuales le hago pruebas como:

  A nivel estructural del código lo ejecuto comprobando que las funciones deseadas funcionan correctamente.

  Cuando hay que interactuar con el usuario, “maltrato” lo máximo posible el código introduciendo valores incorrectos para tratar las excepciones y errores lanzados, intentando emular los posibles errores que pueden cometer los usuarios al usar mi programa.

- ¿Qué crees que es mejor? (selecciona tu respuesta)
    - Programar y detenerte en el detalle y probarlo antes de pasarlo a un compañero para su integración. Aquí puedes resolver incidencias antes de la integración, a cambio de ser más lento en tu entrega.
    - Programar rápidamente y pasarlo cuanto antes a un compañero. Aquí la entrega es muy rápida, las incidencias que se detecten a posteriori deberás revisarlas más tarde para poder solucionarlas.

  Mi respuesta es: Programar y detenerme en detalle y probarlo antes de pasarlo a un compañero...

- Respecto a la documentación técnica del código fuente, base de datos o un proyectos (selecciona tu respuesta).
    - Es mejor no escribir mucho, de todos modos, el que sabe, sabrá lo que hace mirando el código.
    - Es mejor detallarlo todo lo posible, esto ayudará a otra persona o a mí mismo a entender el contexto de cada parte del proyecto/código más rápidamente en otro momento.
    - Ni mucho, ni poco, con mantenerlo más o menos basta, así consigo un balanceado entre trabajo y las ventajas de la documentación.

  En caso de comentar el código, hay diversidad de opiniones por lo que he visto en el mundo de la programación.

  Yo pienso que hay que comentar el código lo justo y necesario (y de una manera clara y concisa) para no influenciar en la persona que está viendo tu código. Quizá no te expliques bien y se confunde más de lo que se ayuda. Si tienes un clean code, se debe poder entender tu código sin muchos comentarios o incluso ninguno.

  Dicho esto, yo en mis proyectos de aprendizaje hago bastantes comentarios para cuando los necesite consultar entender cómo lo hice. Pero esto es a nivel personal, comentarios para mi y para mi propio aprendizaje. Cuando realice mis proyectos a nivel profesional, quiero comentar mi código lo justo para que sea entendible y conciso.

  A nivel de documentación es importante explicar bien y detallar todo lo posible.

  Mi respuesta es: Es mejor detallarlo todo lo posible, esto ayudará a otra persona o a mí...