

RWellsTattoo

Progressive Web App

1 – Introducción - Objetivos

- Aplicación multiplataforma
- Gestión de citas
- Crear listas de inventario
- Portfolio accesible por cliente final
- Actualizable, mantenible y posicionable
- Uso de tecnologías modernas, diseño sencillo, tendencias UX/UI

2 – Metodología – Ciclo de vida

- El ciclo de vida se ha organizado mediante Scrumban, mezclando el sistema de sprints de Scrum, y fraccionando cada uno en simples de tareas, breves. Cuando se completan todas las tareas del sprint se le entrega una parte funcional al cliente
- El ciclo de vida se divide en, backlog, sprint, wip, done, deploy, verify y ready

Backlog

- Administración:

- Login de acceso a un único usuario, sin registro
- Gestión de citas para el administrador
- Gestión de stock con pequeñas “shopping lists”
- Adaptabilidad a dispositivos móviles. Mobile First

- Landing page:

- Creación de un portfolio o landing page
- Formulario de contacto
- Válido para posicionamiento SEO

Sprints

- Planteamiento del diseño UX/UI
- Arquitectura y creación de las bases del proyecto
- Configuración de la arquitectura
- Creación y configuración del sistema de autentificación
- Creación del sistema de API Rest
- Creación y configuración del store local del navegador
- Creación del componente Inventario
- Creación del componente calendario
- Finalización estética y testeо en dispositivos móviles y de escritorio

3 – Tecnologías

- Lenguaje - IDE – Control de versiones
- Frontend
- Backend
- Administración de Sistemas

Lenguaje - IDE – Control de versiones

- **Javascript como lenguaje**, es el más utilizado del mundo, programar el cliente y el lado del servidor
- **WebStorm como IDE**, integración de un terminal, Git, estructuración de proyectos, actualizaciones, plugins
- **Control de versiones: Git**. nos permitirá ver la evolución de nuestro proyecto. Bitbucket como servicio de almacenamiento
- **Gestor de paquetes: NPM**. Es el gestor de paquetes de Node JS, cualquier librería disponible

Frontend

- **Vue JS.** Es un marco progresivo para construir interfaces de usuario. Es progresivo, reactivo, con una gran comunidad, gran rendimiento
- **Nuxt JS.** Aplicaciones universales utilizando Vue JS. Renderizar y compilar tanto del lado del cliente como del lado del servidor. SEO, tiempo de carga reducido, SSR o SPA
- **Vuetify.** componentes semánticos para Vue JS. Componentes limpios, semánticos y reutilizables que facilitan la construcción de aplicaciones. Diseño bajo Material Design

Backend

- **Node JS.** Entorno multiplataforma para servidor basado en Javascript. **Express**, framework, funcionalidades como el enrutamiento, sesiones y cookies. **Passport JS**, framework, para autenticación de usuarios, OpenID, Oauth o local
- **MongoDB.** Sistema NOSQL más utilizado. Datos en esquema dinámico, integración de datos fácil y rápido
- **Webpack.** Empaquetador de módulos. Separa código en módulos que se utilizan como dependencias en otros
- **Babel JS.** Transcompilador que convierte código de ES6 en ES5

Administración de Sistemas

- **Cloud Computing.** Creación de servidores en internet. Reduce los costes, tiempo de actividad, sitios invulnerables a los ataques. utilizaremos Hetzner y dominio con 1&1 Ionos
- **Ubuntu 18.04 LTS.** Sistema operativo de código abierto. Distribución de Linux basada en Debian
- **Nginx.** Servidor web/proxy de alto rendimiento. Software libre y código abierto
- **Pm2.** Gestor de procesos en producción para aplicaciones Node JS. Mantiene activas las aplicaciones evitando tiempos de inactividad
- **Certbot.** Certificados TLS/SSL y cifrado HTTPS

4 – Planificación - Presupuesto

- Costes directos de desarrollo
- Costes indirectos de desarrollo
- Costes de mantenimiento

Costes directos

- Planteamiento del diseño – 160 euros
- Creación de arquitectura – 360 euros
- Configuración de arquitectura – 100 euros
- Sistema de autenticación – 80 euros
- Servidor API Rest – 260 euros
- Store Local – 100 euros
- Inventario – 100 euros
- Calendario – 80 euros
- Testing – 140 euros
- Entrega

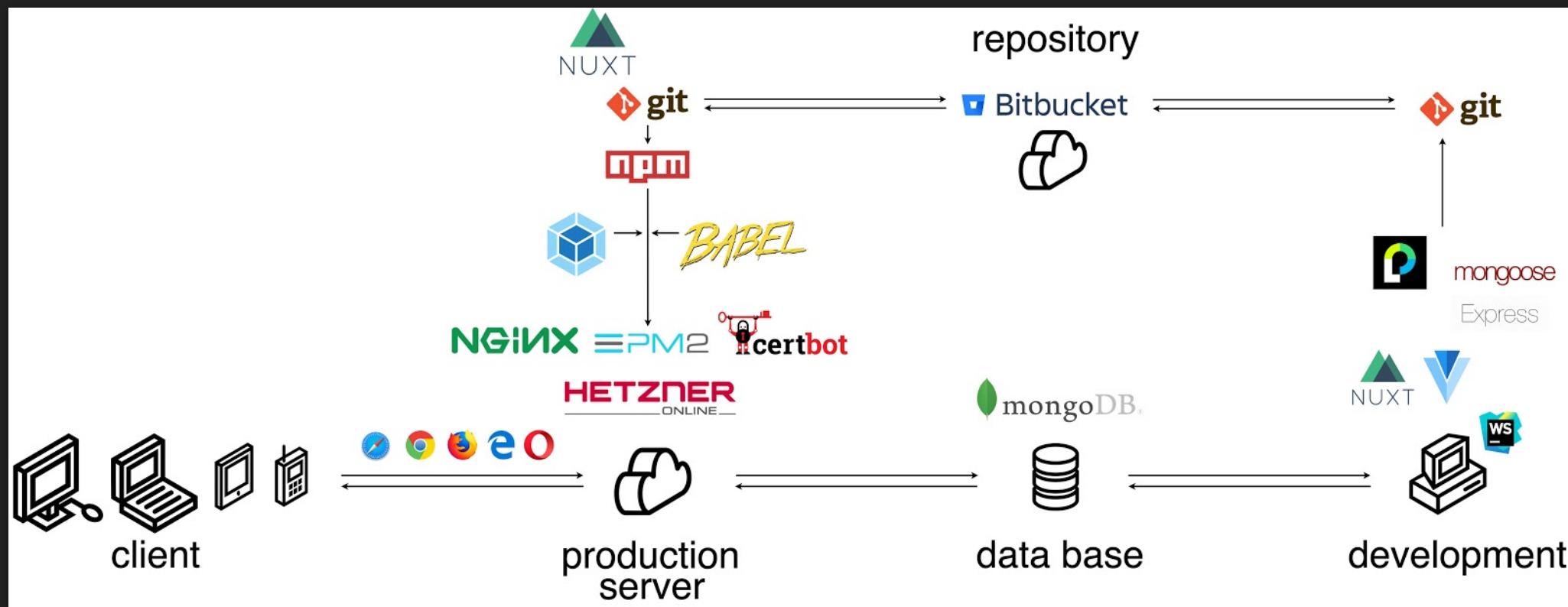
Costes indirectos

- Consultoría a terceros – 50 euros/ hora
- Servicios de almacenamiento – 50 euros/ mes
- Repositorios privados – 70 euros/ mes
- Gastos de prueba – 30 euros/ hora
- Licencias – 12 euros/ mes

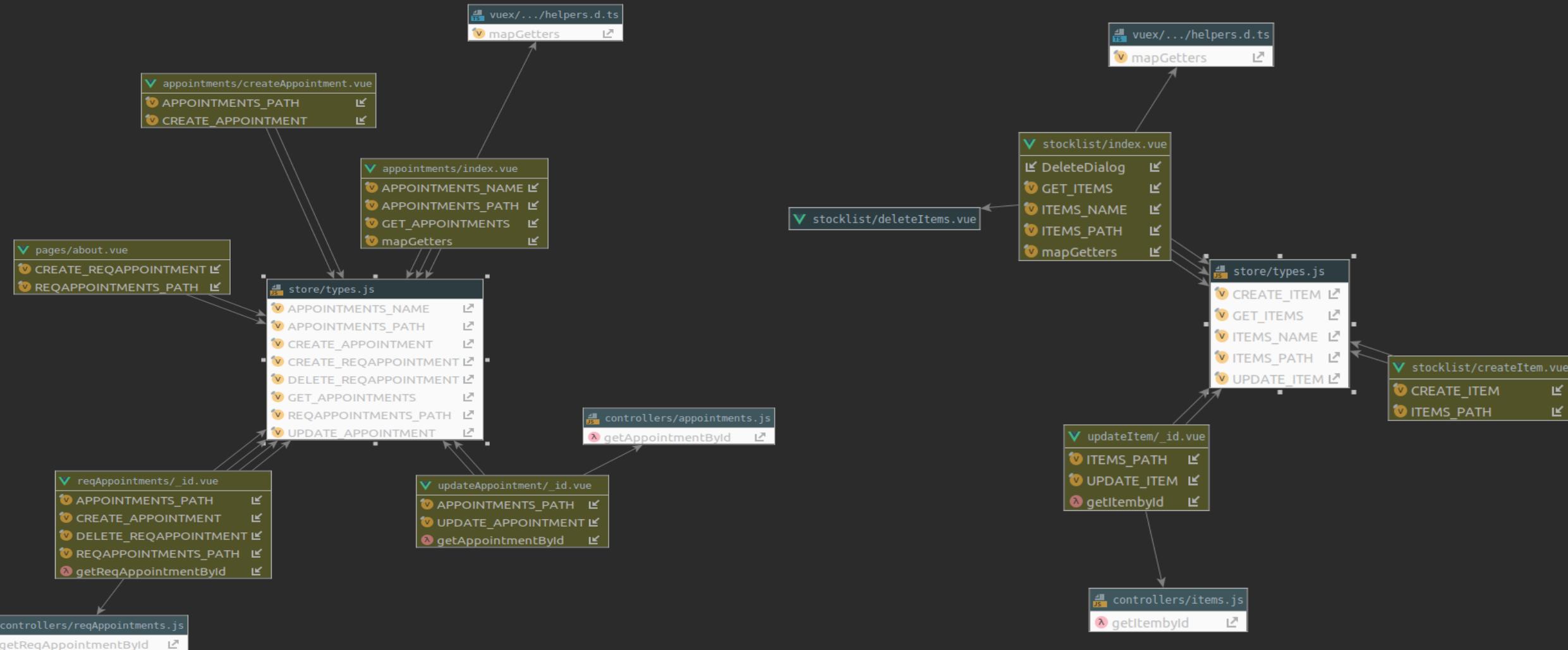
Costes de mantenimiento

- Resolución de indecencias
- Atención 24 horas en los 60 primeros días de implantación
- Atención posterior, en horario laboral a concretar
- Modificaciones mínimas
- Tarifa de 30 euros/ hora

5 – Desarrollo



Análisis



Diseño – Administración de sistemas

- Creación de maquina virtual Ubuntu en Hetzner
- Instalación de Node JS
- Instalación de NGINX y configuración con dominio
- Instalación de PM2
- Instalación de Cerbot
- Instalación de MongoDB
- Creación local del proyecto en Nuxt JS
- Creación del repositorio con Git y Bitbucket

Diseño – Desarrollo local - Frontend

- assets – imágenes, iconografía y otros recursos visuales
- components – componentes reutilizados en varias partes
- layouts – landing page, páginas de error, administración
- pages – componentes de rutas dinámicas
- plugins – paleta de colores en Vuetify
- static – fav.icon para PWA

Diseño – Desarrollo local - Store

- controllers – funciones axios que conectan store con backend
- store – almacenamiento local en el navegador. Estado reactivo
 - index.js – instanciación de las partes del store (state, actions, mutations)
 - types.js – instanciación de funciones del store para front (actions, mutations, name, path)
 - appointments.js (entre otros) – importación de controllers y ejecución de actions, mutations

Diseño – Desarrollo local - Backend

- server – dependencia del backend
 - config – sistema de autenticación de Passport JS
 - controllers – comunicación con axios para base de datos (create, read, update, delete), router de express (api.js), controlador de autenticación
 - models – modelización de datos con mongoose para MongoDB
 - utils – respuestas para el manejo de errores en formato JSON (responses.js)
 - index.js – creación de server y comunicación del resto de los módulos

6 – Pruebas y despliegue

- Registro de único usuario
- Login de usuario
- Logout de usuario
- Creación de petición de cita
- Aceptación de cita
- Creación, Actualización y borrado de cita
- Creación, Actualización y borrado de item

Despliegue

- Pruebas en local
- Pruebas en producción
- Configuración de deploy en Hetzner con git bare
- Automatización con bash y pm2

7 – Conclusiones

- Objetivos alcanzados:

- Aplicación multiplataforma
- Gestión de citas
- Inventario
- Portfolio posicionable

- Objetivos descartados:

- Firebase
- Nuxt JS en vez de Vue JS puro
- Electron JS

Ampliaciones

- Sistema de notificación vía email con Nodemailer.
- Sistema de notificación vía Twilio o AWS.
- Pasarela de pago con Paypal para el depósito o fianza.
- Pasarela de pago con Sprite como alternativa a Paypal.
- Generación de contenedores con Docker.
- Campañas de mailing con Mailchimp.
- Crear una tienda dentro del portfolio para venta de productos.

Valoraciones

- Utilización real de técnicas de Ingeniería del Software
- Herramientas modernas
- Aprendizaje exprés de nuevas tecnologías

Despedida

¡Gracias por la experiencia!

Si se puede soñar, se puede programar.