Recursos

Firebase setcurrentuser

<https://firebase.google.com/docs/auth/web/manage-users?hl=es>

graficos

<https://www.chartjs.org/>

wheather api

<https://openweathermap.org/api>

mapas y folletos interactivos

<https://travishorn.com/interactive-maps-with-vue-leaflet-5430527353c8>

<https://leafletjs.com/>

NECESIDAD QUE VAMOS A CUBRIR

Cada día la gente viaja más y planifica sus viajes con más antelación, por lo que, poder guardar ofertas de hoteles y vuelos para poder verlos más adelante o simplemente poder comparar los precios será bastante útil para los viajeros.

IDEA

Crear una aplicación web que me permita filtrar hoteles por precio y guardar en mis favoritos los hoteles y vuelos con la intención de compararlos.

REFERENCIAS

<https://www.tripadvisor.es/>

<https://www.booking.com>

MODELO DE NEGOCIO

Este proyecto lo puedo monetizar mediante publicidad en AdSense o mediante marketing de afiliados (por ejemplo booking)

FUNCIONALIDADES / CARACTERÍSTICAS

* Registro, identificación y zona privada de usuario
* Buscar hoteles por fecha, habitación y número de ususarios
* Filtrar hoteles por precio
* Información de hoteles
* Guardar hoteles en favoritos y que el usuario pueda verlos(solo los suyos) (poder filtrar por precios)
* Búsqueda de vuelos
* Guardar Vuelos en favoritos y que el usuario pueda verlos
* Mostrar noticias en la página principal(scraping)
* Mostrar la programación del tiempo para viajes en los proximos 30 días
* Posibilidad de contactar con nosotros
* Posibilidad de encontrar hoteles en el mapa de google
* Registra tu propio alojamiento
* Multiidioma
* Facilidad de uso (Usabilidad)
* Diseño elegante y limpio (Diseño)

ÁRBOL WEB / CONTENIDOS

* Página principal o de aterrizaje (**pública**)
* Login (**pública**)
  + Registro (**pública**)
  + Solicitar modificar contraseña (**pública**)
  + Modificar contraseña (**pública con token**)
* Zona de usuario (**privada**)
  + Datos principales de la cuenta (**privada**)
* HOTELES (**pública**)
* VUELOS (**pública**)
* MIS FAVORITOS (**privada**)
  + Guardar y filtrado de hoteles por precio
* Aviso legal, política de privacidad, política de cookies… (**pública**)
* Contacto (**pública**)

PERFILES DE USUARIO

* **Usuarios anónimos**: Tendrán acceso a todo el contenido público exclusivamente.
* **Usuarios registrados**: Tendrán acceso a todo el contenido público y además podrán acceder a la página de favoritos, donde tendrán guardados sus hoteles y vuelos.

PROCESOS

Registro

El registro se hará mediante una serie de datos básicos que serán:

* Email
* Contraseña

Una vez que el usuario haya introducido correctamente sus datos, se le enviará un email de bienvenida con un enlace que verificará la creación de la cuenta.

Además, si los datos introducidos son correctos y la cuenta se ha creado, independientemente de que el email se haya verificado o no, el usuario quedará autenticado automáticamente en la aplicación.

Identificación

Se le pedirá al usuario email y contraseña para acceder. Si todo va correctamente se le enviará a la página principal.

Modificación de contraseña

El usuario introducirá la contraseña asociada a su cuenta en nuestra aplicación. Si todo va bien, el usuario recibirá un correo electrónico informando sobre la solicitud de cambio de contraseña con un enlace para comenzar el proceso de modificación de contraseña.

Cuando el usuario pulsa sobre este enlace, accederá a una pantalla pública en la que deberá introducir la nueva contraseña (siguiendo las buenas prácticas en materia de seguridad)  y deberá pulsar en el botón modificar.

Si la contraseña se modifica correctamente, se le enviará otro email informando sobre el cambio de contraseña.

Confección de búsqueda de hoteles

Tendré una caja de búsqueda en la que se recogerán los siguientes datos:

* Ciudad
* Fecha de checkin
* Noches
* Nº Adultos
* Nº de habitaciónes

Mediante peticiones a una api se pintarán los datos de los hoteles en una caja justo debajo de la caja de búsqueda, estos datos son:

* título del hotel
* foto
* categoría
* precio
* localización
* votos
* numero de comentarios

--------------------------------------------------------------

ARQUITECTURA SOFTWARE

El proyecto estará planteado con una arquitectura headless (arquitectura desacoplada) que permitirá flexibilidad y escalabilidad en el consumo de los datos y en el uso de diferentes aplicaciones cliente.

* **Cliente CV Online\***: Estará encargado de servir como interfaz a los usuarios que accedan a través del dominio miaplicacion.com. Llamará a la API REST para obtener o enviar la información que requiera.
* **API REST**: Se encargará de servir y recibir las peticiones de información u otros contenidos (por ejemplo los CV's en formato pdf)
* **CMS\***: Estará encargado de servir como herramienta para los propietarios y trabajadores de la herramienta para poder gestionar la información que verán los usuarios en el “Cliente CV Conline” y poder dar soporte.

\* Aplicación cliente que se basará en tecnologías web (de navegador).

TECNOLOGÍAS

API REST

**Tecnologías**

* **NODE** - Entorno de ejecución de JavaScript (Base)
* **MONGO** - Persistir datos en el disco

**Dependencias producción**

* **EXPRESS** - Framework de desarrollo
* **BEARER TOKEN** - Permite extraer de la cabecera Authorization el token bearer que se le pase (si no se le pasa ninguno, en la petición tendrá valor nulo). El token estará almacenado en req.token (dentro del middleware)
* **BODY PARSER** - Permite convertir el cuerpo de una petición a formato JSON automáticamente
* **JWT** - Permite crear y validar un token JWT.
* **CORS** - Permite que se pueda llamar a mi API desde cualquier sitio (pudiendo si quiero restringir desde donde se llama)
* **MONGOOSE** - Crear esquemas y modelos para definir los modelos de datos en Mongo
* **NODEMAILER** - Enviar emails (necesitaríamos un email para hacer el envío)
* **MD5** - para encriptar las contraseñas de los usuarios en Mongo
* **HTML-PDF** - convierte un HTML renderizado a fichero PDF
* **STRIPE** - plataforma para realizar cargos a los usuarios en sus tarjetas de crédito/débito
* **NODE-LINKEDIN** - Permite importar datos desde LinkedIn para cumplimentar la información del CV de los usuarios

**Dependencias desarrollo**

* **NODEMON** - Dependencia de desarrollo para poder utilizar la funcionalidad hot reloading.
* **MOCHA** - Testing con JavaScript
* **CHAI** - Testing con JavaScript (Made easy)
* **DOTENV** - Para precargar variables de entorno desde fichero .env

**Sistemas**

* Sistema Linux (distro Ubuntu) 20 x64
* Servidor Node 10+ LTS
* Servidor Mongo 4+ LTS
* SSL Let's Encrypt gratuito
* Envío de emails a través de Amazon SES / Sendgrid

CLIENTE CV ONLINE

**Tecnologías**

* **NUXT** - Framework de desarrollo en el lado del cliente con SSR

**Dependencias producción**

* **AXIOS** - Librería para realizar llamadas HTTP
* **BOOTSTRAP VUE** - Framework maquetación para montar interfaces responsive basadas en Bootstrap
* **PWA** - Habilitar funcionalidades de Progressive Web Application
* **JWT-DECODE** - Decodificación del token JWT
* **VUE-i18N** - Librería para trabajar con internacionalización
* **VUE2-GOOGLE-MAPS** - Librería para la integración con Google Maps

**Dependencias desarrollo**

* **SASS** - Precompilador de CSS
* **DONTENV** - Carga de variables de entorno en Nuxt

CLIENTE CMS

…. habría que especificar las dependencias pero son casi idénticas a las del CLIENTE CV ONLINE

PLANIFICACIÓN

**Tiempo de entrega**: 3 semanas (**completas**)

Bloques a desarrollar:

1. Frontend (Cliente CV Online)
   1. Páginas públicas (página principal, contacto, preguntas frecuente, plantillas y guía confección cv…)
   2. Registro, identificación, solicitar modificar contraseña y modificar contraseña
   3. Proceso de confección de un cv
   4. Zona privada de usuario
2. Backend (API REST)
   1. Endpoints asociados a páginas públicas (información página principal, contacto, preguntas frecuentes, plantillas y contenido guía confección de cv…)
   2. Endpoints registro, login, logout, solicitar modificar contraseña, modificación contraseña
   3. Endpoints para registrar el cv, endpoints para obtener los datos de plataformas externas. Mecanismos para realizar los cobros correspondientes a los distintos usuarios
   4. Endpoints para la zona de usuarios (registro datos de pago, modificación datos del usuario, modificación datos facturación, historial facturas y listado de CV's)

**Semana 1: puntos a y b**

**Semana 2: puntos c y d**

**Semana 3: Revisión, corrección de errores y brillo**

GENERACIÓN DEL MVP

* Crear un curriculum **X**
* Listado de plantillas de curriculums **X**
* Registro, identificación y zona privada de usuario
* Cobro según plan seleccionado
* Multiidioma
* Facilidad de uso (Usabilidad) **X!**
* Diseño elegante y limpio (Diseño) **X**
* Importar CV de otras plataformas (LinkedIn, Tecnoempleo, Facebook)
* Durante el proceso de creación del CV quiero que se guarde el progreso
* Poder descargar mi CV con el diseño y estructura seleccionado en formato PDF **X**
* Crear un CV a partir de otro ya existente
* Descarga de facturas en formato PDF
* Que el usuario pueda ver y gestionar sus pagos, CV's y datos de tarjeta
* Zona de preguntas frecuentes para ayudar al usuario
* Guía de como confeccionar/orientar mi CV
* Posibilidad de contactar con nosotros **X**
* Contenido público indexable (que me puedan encontrar por Google)

CARACTERÍSTICAS MVP

* Creación de un curriculum
* Listado de plantillas
* Facilidad de uso (Usabilidad)
* Diseño elegante y limpio (Diseño)
* Poder descargar mi CV con el diseño y estructura seleccionado en formato PDF
* Posibilidad de contactar con nosotros

ITERACIONES POSTERIORES

* Registro, identificación y zona privada de usuario
* Cobro según plan seleccionado
* Multiidioma
* Importar CV de otras plataformas (LinkedIn, Tecnoempleo, Facebook)
* Durante el proceso de creación del CV quiero que se guarde el progreso
* Poder descargar mi CV con el diseño y estructura seleccionado en formato PDF **X**
* Crear un CV a partir de otro ya existente
* Descarga de facturas en formato PDF
* Que el usuario pueda ver y gestionar sus pagos, CV's y datos de tarjeta
* Zona de preguntas frecuentes para ayudar al usuario
* Contenido público indexable (que me puedan encontrar por Google)

PROCESO DE EJECUCIÓN MVP + ITERACIONES

Bloques de funcionalidades

* Página principal
* Plantillas
* Confección del CV
  + Sin importación de datos
  + Guardado a PDF

**------------------------------------ MPV ALCANZADO ------------------------------------**

* + Con importación de datos
  + Con guardado de progreso
* Contacto
* Guía
* Sistema de usuarios
  + Registro
  + Identificación
  + Solicitar modificar contraseña
  + Modificar contraseña
  + Área privada usuario
    - Edición datos principales
* Preguntas frecuentes
* Cobros
  + Registro de datos de pago
  + Planes de suscripción
  + Selección plan suscripción por el usuario
  + Cargo plan de suscripción correspondiente
  + Registro de datos de facturación
  + Generar factura
  + Historial de facturas área privada usuario
  + Descarga de facturas PDF

PLANIFICACIÓN MVP + ITERACIONES

**Tiempo de entrega**: 3 semanas (**completas**)

Bloques a desarrollar:

**MVP:** Generación del MVP y cumplir con el objetivo básico del proyecto

día 18 de junio

**Iteración 1**: importación externa de datos en cv + guardado del progreso (objetivo añadir valor a mi proceso de creación de CV)

* + Con importación de datos
  + Con guardado de progreso

día 21 de junio

**Iteración 2**: Página contacto + Guía para confección de buenos CV + Sistemas de usuarios

* Contacto
* Guía
* Sistema de usuarios
  + Registro
  + Identificación
  + Solicitar modificar contraseña
  + Modificar contraseña
  + Área privada usuario
    - Edición datos principales

día 23 de junio

**Iteración 3**: Preguntas frecuentes

día 24 de junio

**Iteración 4**: Sistema de cobros (Monetización real de mi proyecto)

* Cobros
  + Registro de datos de pago
  + Planes de suscripción
  + Selección plan suscripción por el usuario
  + Cargo plan de suscripción correspondiente
  + Registro de datos de facturación
  + Generar factura
  + Historial de facturas área privada usuario
  + Descarga de facturas PDF

día 28 de junio

ORGANIZACIÓN DEL ENTORNO DE TRABAJO

Cómo me voy a organizar:

* Editor: **Visual Studio** con las siguientes extensiones:
  + Git
  + DotEnv
  + EditorConfig
  + Rainbow Brackets
  + Visual Studio IntelliCode
* **1 repositorio para cada pata** del software
  + cv-online-pwa
    - master
    - develop
  + cv-online-api
    - master
    - develop
  + cv-online-cms
    - master
    - develop
* Los proyectos en producción se desplegarán en Heroku

Otras consideraciones por repositorio

* Fichero **.editorconfig** en el raiz de cada proyecto
* Ficheto **.gitignore** en el raiz de cada proyecto (ignorado cualquier dato sensible, dependencia, etc.)
* **README.me** con instrucciones de como instalar por primera vez el proyecto
* **Rama master** para versiones en producción (estables) (tageadas con su versión correspondiente)
* **Rama develop** para versiones en desarrollo (inestables)

DISEÑO DE LA API