

Manual técnico Te Vi Colombia

16.05.2022

Andrés Mauricio Montoya Sánchez

Te Vi Colombia

Girardot, Cundinamarca Colombia

2022

FASES DEL PROCESO DE DESARROLLO DEL SOFTWARE

Un proyecto de software a medida en la mayoría de los casos suele generar un impacto muy positivo en la empresa en donde se lleva a cabo ya que ha sido diseñado y creado según las necesidades exactas de quién lo ha pedido. Pero para que el impacto sea positivo realmente y que se generen beneficios, el proceso debe ser llevado con orden con buenas prácticas y estándares que aseguren los tiempos promedios, los costos y que el sistema cumpla con el papel para el cual fue creado cumpliendo con todos los requerimientos indicados por el cliente. (Neosystems, 2014); Según lo mencionado, estas fases consisten en:

ANALISIS

En la etapa de análisis se realiza la visión previa del desarrollo final del software, teniendo en cuenta el problema encontrado y la solución a aplicar con tecnologías de desarrollo que se van a mencionar, y aplicando métodos para la recolección y análisis de datos. La metodología de desarrollo seleccionada para el software, permite realizar incrementos y aplicarlos a cada una de las fases. Del resultado del análisis con herramientas para la recolección de datos, se encontraron las siguientes particularidades para el desarrollo del software:

- Visualización de formulación de estrategias para un análisis incentivo en áreas previamente establecidas.
- Estudio de técnicas para el mejoramiento y búsqueda de información requerida por un integrante de la aplicación.
- Eventualidades basadas en funciones de almacenamiento de información, para el consumo y estudio de determinados integrantes de la plataforma.
- Interfaz agradable y amigable para la visualización del usuario.

DISEÑO

Después de la información recolectada de la fase de análisis del desarrollo final, se encuentra que a manera general se utilizará elementos como: base de datos para el almacenamiento de información, interfaz adaptada a cualquier dispositivo para la visualización de datos, determinación de las tecnologías de desarrollo a utilizar para el planteo de información de una emprendedor o usuario registrado.

DESARROLLO

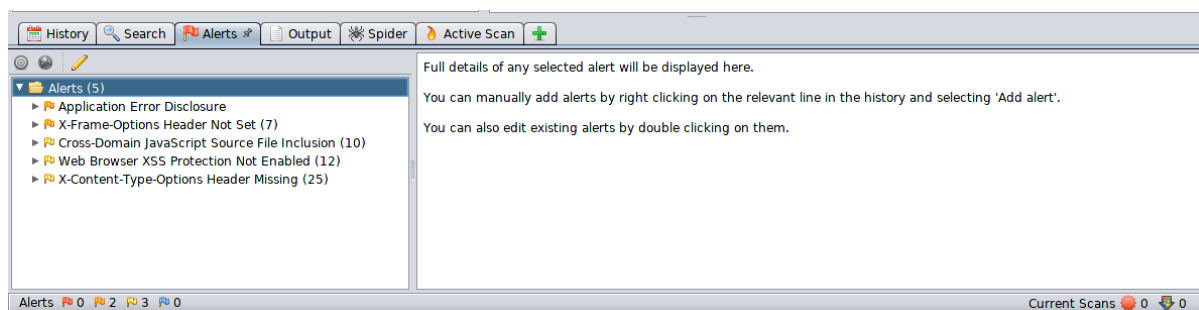
Con las herramientas de desarrollo anteriormente mencionadas y seleccionadas, se realiza el desarrollo de una aplicación bajo la guía de distintos diagramas. Estos se dividen en dos:

- API (Application programming interface). Encargada de recibir y procesar datos para visualizarlos en la aplicación.
- SPA (Single page application). Aplicación web encargada de recibir la información de la API.

PRUEBAS

En el desarrollo de funciones en particulares áreas de la aplicación, se hará sus debidas pruebas, junto a herramientas como Jest que es una tool para el testeo de aplicaciones escritas en Javascript; Asegurando que el código devuelve lo esperado. También se incluye la herramienta de seguridad como OWASP Zed Attack Proxy (ZAP) siendo la tool de seguridad informática más usada en el mundo.

En este caso, estos son los resultados que dio OWASP Zed Attack Proxy (ZAP):



Fuente: Autor

Como se puede ver, no se encontró ninguna vulnerabilidad grave, lo cual está bien. Pero lo interesante en este análisis es que Te Vi Colombia fue puesto a prueba sin tener algún Firewall, CDN o Proxy reverso detrás protegiéndolo, y a pesar de ello, supo protegerse de una manera positiva.

Obviamente, esta aplicación en producción con el servicio de Vercel será más segura.

DOCUMENTACIÓN

- Manual de usuario: Consiste en un documento de texto el cual explicará el funcionamiento de la aplicación. Tratando temas como, qué dispositivos son compatibles, cuantas pulgadas debe de tener un dispositivo y cómo cada función del software trabaja.
- Manual técnico: Se indica el propósito y breve descripción de cada método/función, con su prototipo indicando argumentos y respuestas.

IMPLEMENTACIÓN Y MANTENIMIENTO

La aplicación requiere de inversión económica para el pago de servidores, DNS, Databases as a service, entre otros, que van a hacer implementados en servicios de la nube como lo pueden ser Vercel y AWS for Databases and Storage, con relación a otras tecnologías como lo pueden ser Node.js, GraphQL, React.js, DynamoDB, Redis y demás, éstas cuentan con licenciamiento de software libre, lo cual hace que a nivel de mantenimiento no haya ningún problema financiero o legal. Todo esto corresponde a los gastos de la mano de obra de los administradores.

HERRAMIENTAS

En el desarrollo de la aplicación final, se utilizaron las siguientes herramientas para el diseño y desarrollo:

- React.js
- Next.js
- Node.js
- GraphQL
- DynamoDB
- Redis
- TypeScript

PROPUESTA SOLUCIÓN

El sistema de información para la conexión de tejidos virtuales en Colombia está orientado en un Ambiente Móvil/Web, el cual se basa en el estudio de acompañamiento y ayuda en Emprendimiento, Práctica profesional y Empleabilidad.

Esta exploración logra su objetivo a través de la información que se encuentra almacenada en una base de datos, las cual son representaciones que sintetizan algunos de los factores, parámetros o características más relevantes para seleccionar el beneficiario más apropiado en función de los objetivos perseguidos, las circunstancias del entorno y los recursos y capacidades que este ofrece.

Este software se orienta en los posteriores módulos:

REGISTRO DE USUARIO.

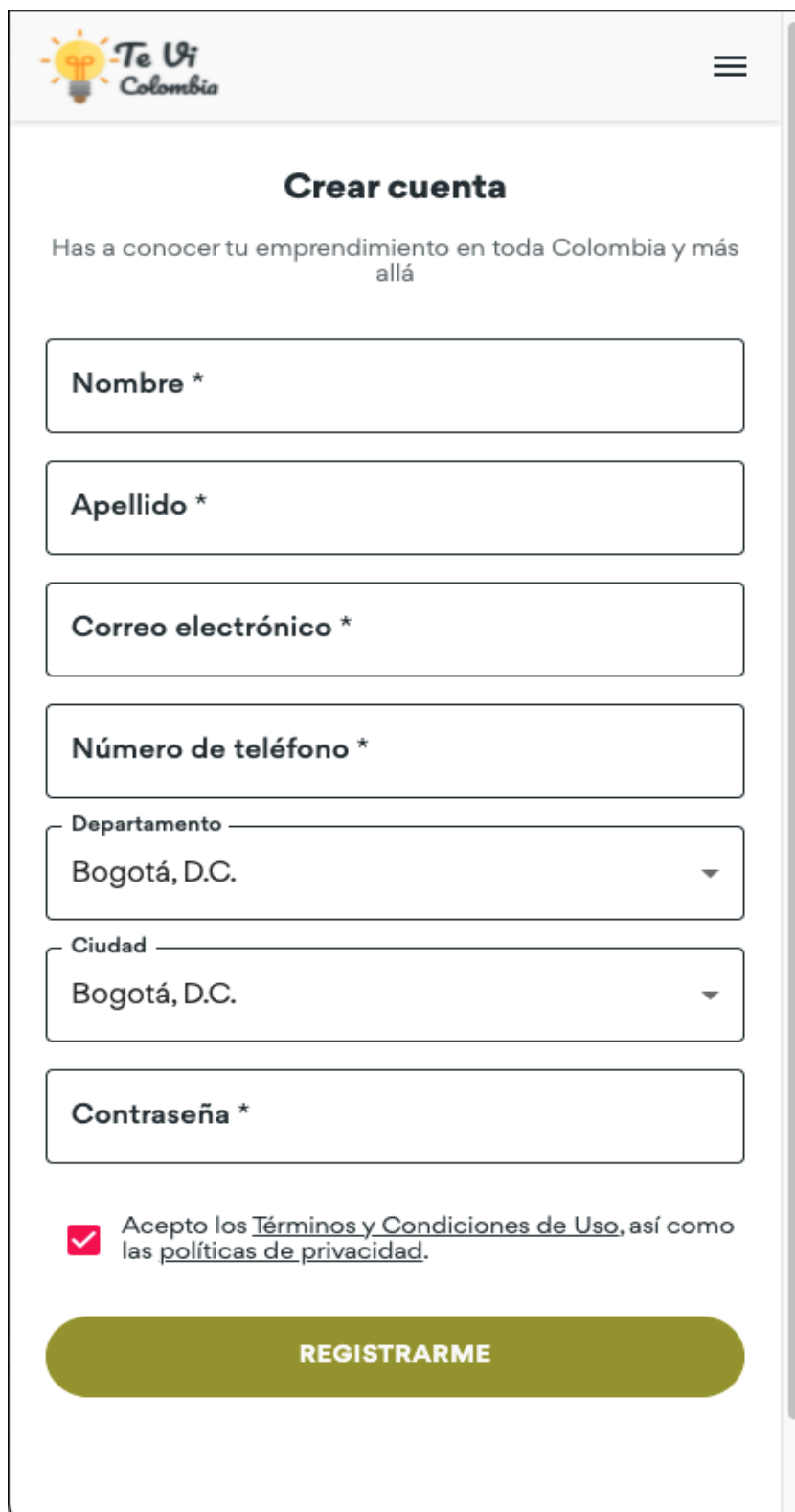
El usuario o emprendedor puede crear una cuenta, con la cual se podrá identificar en el aplicativo.



Diagrama de la interfaz de usuario para el registro de un usuario. El formulario está dividido en varias secciones:

- Logo:** Un campo rectangular en la parte superior central.
- Nombre de Usuario:** Un campo rectangular a la izquierda.
- Apellido de Usuario:** Un campo rectangular a la derecha.
- Correo Electrónico de Usuario:** Un campo rectangular a la izquierda.
- Teléfono de Usuario:** Un campo rectangular a la derecha.
- TIPO DE DOCUMENTO:** Un campo con un menú desplegable a la izquierda. Una línea con una flecha apunta desde este campo hacia un cuadro que contiene "CÉDULA DE CIUDADANÍA" y "TARJETA DE IDENTIDAD".
- NUMERO DE IDENTIFICACIÓN:** Un campo rectangular a la derecha.
- CONTRASEÑA DE USUARIO:** Un campo rectangular a la izquierda.
- En cumplimiento de la Ley 1581 de 2012:** Un checkbox con el texto "En el cumplimiento de la Ley 1581 de 2012 sobre el régimen general de protección de datos personales y sobre manejo de los mismo, acepto los términos presentados por la Comisión Nacional del Servicio Civil." Debajo de este texto hay un botón que dice "Términos y condiciones".
- ENVIAR:** Un botón rectangular en la parte inferior central.

Fuente: Autor

Fuente original:



Crear cuenta

Has a conocer tu emprendimiento en toda Colombia y más allá

Nombre *

Apellido *

Correo electrónico *

Número de teléfono *

Departamento
Bogotá, D.C. ▼

Ciudad
Bogotá, D.C. ▼

Contraseña *

Acepto los [Términos y Condiciones de Uso](#), así como las [políticas de privacidad](#).

REGISTRARME

Fuente: Autor

Únicamente se pide información muy básica para lograr diferenciar a cada usuario o emprendedor.

Los términos y condiciones están inspirados en el registro de la página de empleabilidad de la UNIMINUTO <http://empleabilidad.uniminuto.edu/>, el cual cuenta con la suficiente información descriptiva para establecer una cláusula segura ante los datos de los usuarios.

Términos y condiciones para usuarios.

Los datos personales suministrados por el Titular serán utilizados por UNIMINUTO para identificar, recopilar y registrar a los prospectos de cualquier persona interesada en los servicios que ofrece la plataforma *Te vi Colombia*, con el fin de poder prestar sus servicios de generar conexiones de ayuda y alternativas para poder emprender.

Así mismo los datos se recopilan con el fin de mantener actualizada la información del beneficiario, contactarlo, verificar sus datos, identificar las necesidades de formación y capacitación de estudiantes, realizar informes, análisis de información, búsqueda de tendencias en los servicios, enviar información, obtener indicadores, generar información institucional para los procesos de acreditación, generar reportes a entidades externas y requerimientos internos de información.


Los datos serán objeto de recolección, almacenamiento, actualización y copia de seguridad de acuerdo con el tratamiento establecido en la [Política](#) de Tratamiento de información de UNIMINUTO.

El responsable y Encargado de Tratamiento de los datos será UNIMINUTO.

Sus datos se mantendrán almacenados por el periodo establecido en el Manual de Protección de Datos Personas de UNIMINUTO.

El Titular tiene derecho a conocer, actualizar, rectificar, renovar, solicitar la supresión, presentar quejas y reclamos y demás derecho contenidos en la Ley 1581 de 2012 y sus decretos reglamentarios, respecto de los datos suministrados.

Declara bajo la gravedad de juramento que todos los datos aquí contenidos son exactos y veraces y declara conocer la [Política](#) de Tratamiento de información de UNIMINUTO, en el cual me ha informado de manera previa y expresa los derechos que se asisten, la finalidad, tratamiento y vigencia que se le dará a mis datos personales.



En consecuencia de lo anterior, autorizo expresamente de manera libre, previa, voluntaria y debidamente informada, a la Corporación universitaria Minuto de Dios - UNIMINUTO, para que haga el Tratamiento de datos, de acuerdo con las finalidades y condiciones mencionadas en el aviso de privacidad, el cual declaro conocer y aceptar.

Términos y condiciones para empresas.

Los datos personales suministrados por el Titular serán utilizados por *Te vi Colombia* con el fin de canalizar oportunidades de cooperación y/o donaciones, trabajos conjuntos, proyectos, emprendimiento, etc. De igual manera, se utiliza esta formación para conocer a los aliados de UNIMINUTO.

Los datos serán objeto de recolección, almacenamiento, actualización y copia de seguridad de acuerdo con el tratamiento establecido en el [Manual](#) de Protección de Datos Personales de UNIMINUTO.

El Responsable y Encargado del Tratamiento de los datos será UNIMINUTO.

Sus datos se mantendrán almacenados por el periodo establecido en el Manual de Protección de Datos Personales de UNIMINUTO.

El Titular tiene derecho de conocer, actualizar, rectificar, revocar, solicitar la supresión, presentar quejas y reclamos y demás derechos contenidos en la Ley 1581 de 2012 y sus decretos reglamentarios, respecto de los datos suministrados.

Autorización.

Declaro bajo la gravedad de juramento que todos los datos aquí contenidos son exactos y veraces y declaro conocer el [Manual](#) de Protección de Datos personales de UNIMINUTO, en el cual me ha informado de manera previa y expresa los derechos que me asisten, la finalidad, tratamiento y vigencia que se le dará a mis datos personales.

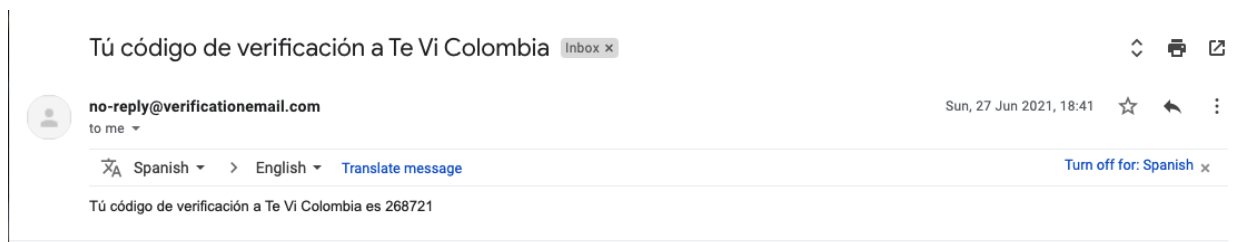
En consecuencia de los anterior, autorizo de manera libre, previa, voluntaria y debidamente informada, a la Corporación universitaria Minuto de Dios - UNIMINUTO, para que haga el Tratamiento de datos, de acuerdo con las finalidades y condiciones mencionadas en el aviso de privacidad, el cual declaro conocer y aceptar.

Términos y condiciones para productos

El objetivo es proteger, promover y garantizar la efectividad y el libre ejercicio de los derechos de los consumidores, así como amparar el respeto a su dignidad y a sus intereses económicos, en especial, lo referente a:

1. La protección de los consumidores frente a los riesgos para su salud y seguridad.
2. El acceso de los consumidores a una información adecuada, de acuerdo con los términos de este, que les permita hacer elecciones bien fundamentadas.
3. La educación del consumidor.
4. La libertad de constituir organizaciones de consumidores y la oportunidad para esas organizaciones de hacer oír sus opiniones en los procesos de adopción de decisiones que las afecten.
5. La protección especial a los niños, niñas y adolescentes, en su calidad de consumidores, de acuerdo con lo establecido en el Código de la Infancia y la Adolescencia.

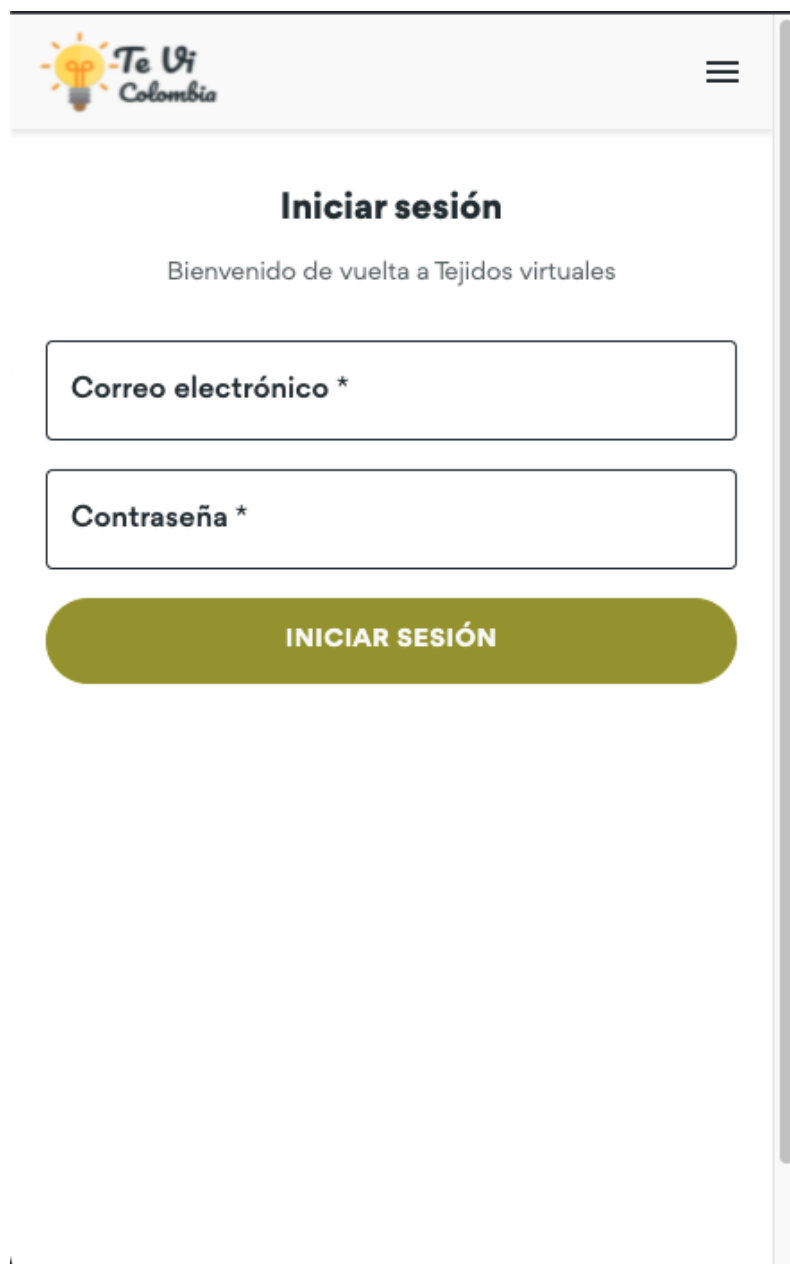
Una vez los términos y condiciones aceptados, el usuario, para saber que el correo ingresado no es falso, se verifica a través de un correo electrónico que el usuario debe de revisar para validar su cuenta de usuario.



Fuente: Autor

LOGIN DE USUARIO.

Una vez el usuario creado y correo validado, este procederá con entrar a la aplicación únicamente con su correo y contraseña, dando clic en *INICIAR SESIÓN*.

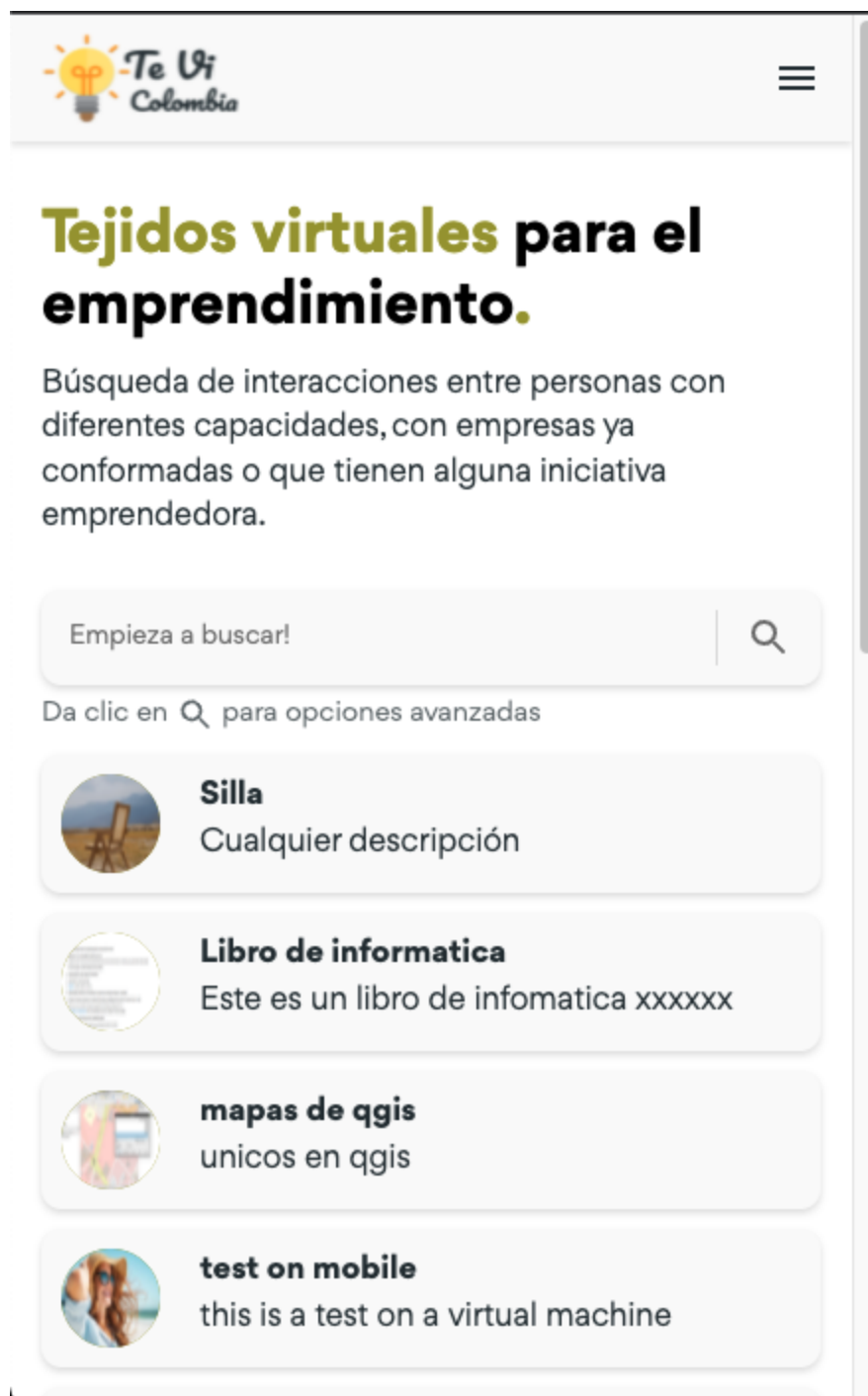


The image shows a mobile application login screen. At the top left is the logo for 'Te Uí Colombia', which includes a lightbulb icon. To the right of the logo is a hamburger menu icon. The main heading is 'Iniciar sesión', followed by the subtitle 'Bienvenido de vuelta a Tejidos virtuales'. Below this are two input fields: 'Correo electrónico *' and 'Contraseña *'. At the bottom is a large, rounded green button labeled 'INICIAR SESIÓN'.

Fuente: Autor

PÁGINA DE INICIO.

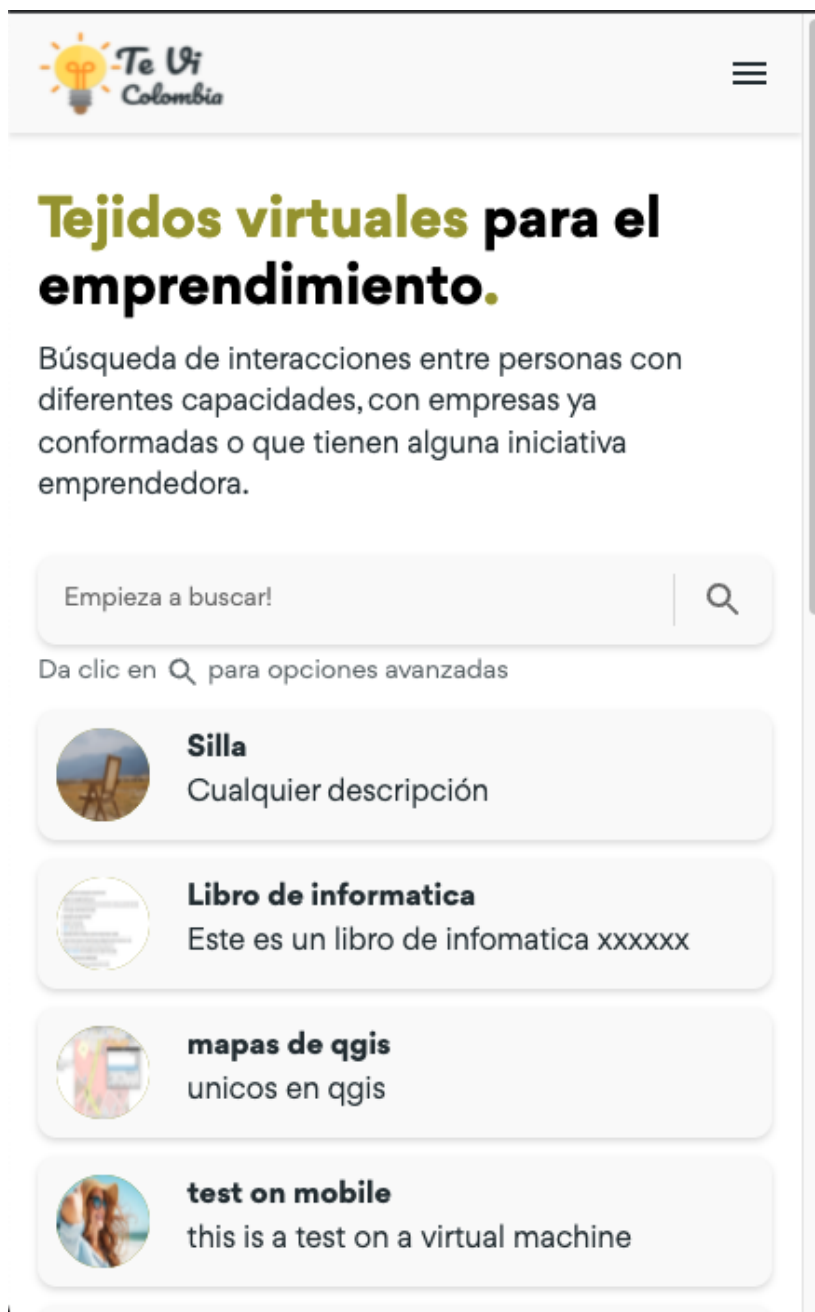
Ya el usuario dentro de la aplicación, lo primero que observará es la página de inicio, la cual contendrá la siguiente vista:




Fuente: Autor

Esta sección cuenta con una de las funcionalidades más importantes del aplicativo, el buscador, el cual permite al usuario buscar cualquier producto que necesite, o persona a contratar.

FUNCIONALIDAD BUSCADOR.



Fuente: Autor



En el caso del buscador, tiene una manera particular de trabajar, dependiendo de la sentencia ingresada, este logrará identificar lo necesario respetando las siguientes opciones:

En el caso de un usuario (Más información en *Mi perfil*), sería:

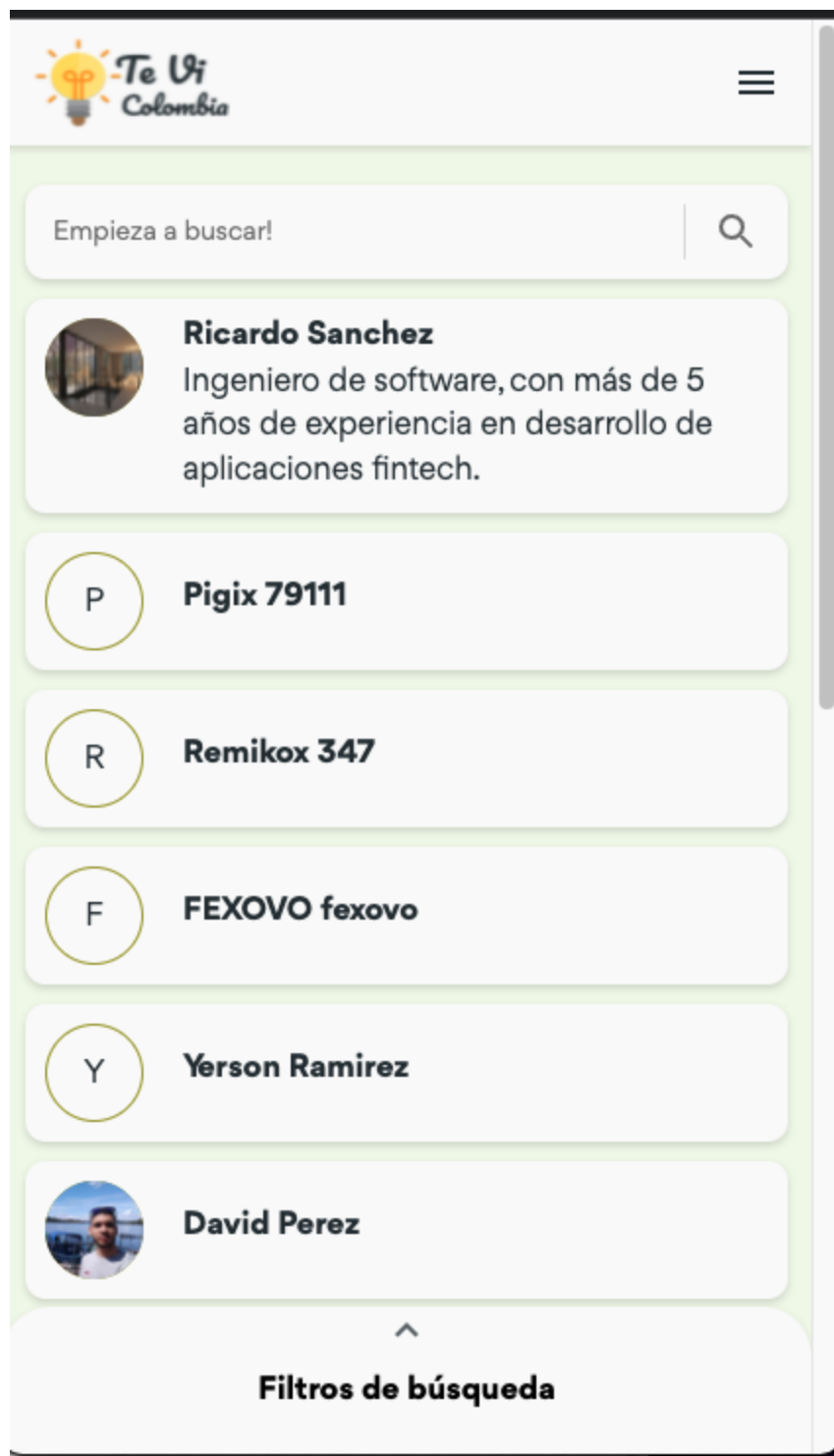
- Nombre.
- Apellido.
- Descripción.
- Correo Electronico.
- País.
- Departamento.
- Ciudad.

En el caso de un producto (Más información en *Producto*), sería:

- Nombre.
- Descripción.
- Tipo.
- Categoría.
- Estado.

Esté se filtra a través de dos categorías “Usuarios” y “Productos”, que se dividen en diferentes parámetros que sirven para que la búsqueda sea más específica.

Interfaz de usuario:



Fuente: Autor

Campos a aceptar:



The image shows a mobile application interface for search filters. At the top, there is a title "Filtros de búsqueda" with a downward arrow icon. Below the title are three dropdown menus: "País" (Country) with "Colombia" selected, "Departamento" (Department), and "Ciudad" (City). Below the dropdowns are two buttons: "LIMPIAR" (Clear) and "BUSCAR POR PRODUCTOS" (Search for products). The interface is clean and modern, with a light gray background and rounded corners.

Fuente: Autor

Interfaz de productos:



Fuente: Autor

Campos a aceptar:



The image shows a mobile application interface for search filters. At the top, there is a small downward arrow icon. Below it, the title "Filtros de búsqueda" is centered. There are three stacked dropdown menus: the first is labeled "Tipo" and has "Producto" selected; the second is labeled "Categoría"; and the third is labeled "Estado". Below these are two rounded rectangular buttons with a yellow border: "LIMPIAR" and "BUSCAR POR USUARIOS". A vertical scrollbar is visible on the right side of the filter panel.

Fuente: Autor

FUNCIONALIDAD DE LISTADOS DE USUARIOS Y PRODUCTOS.

Según la necesidad que tenga un usuario para sus productos o información de si mismo, por ejemplo, según el texto añadido en el textfield y los campos seleccionados, se mostrará lo siguiente:

Si se quiere buscar un usuario con nombre Andrés de Girardot, podemos añadir:

Filtro de búsqueda:



▼

Filtros de búsqueda

País
Colombia ▼

Departamento
Cundinamarca ▼

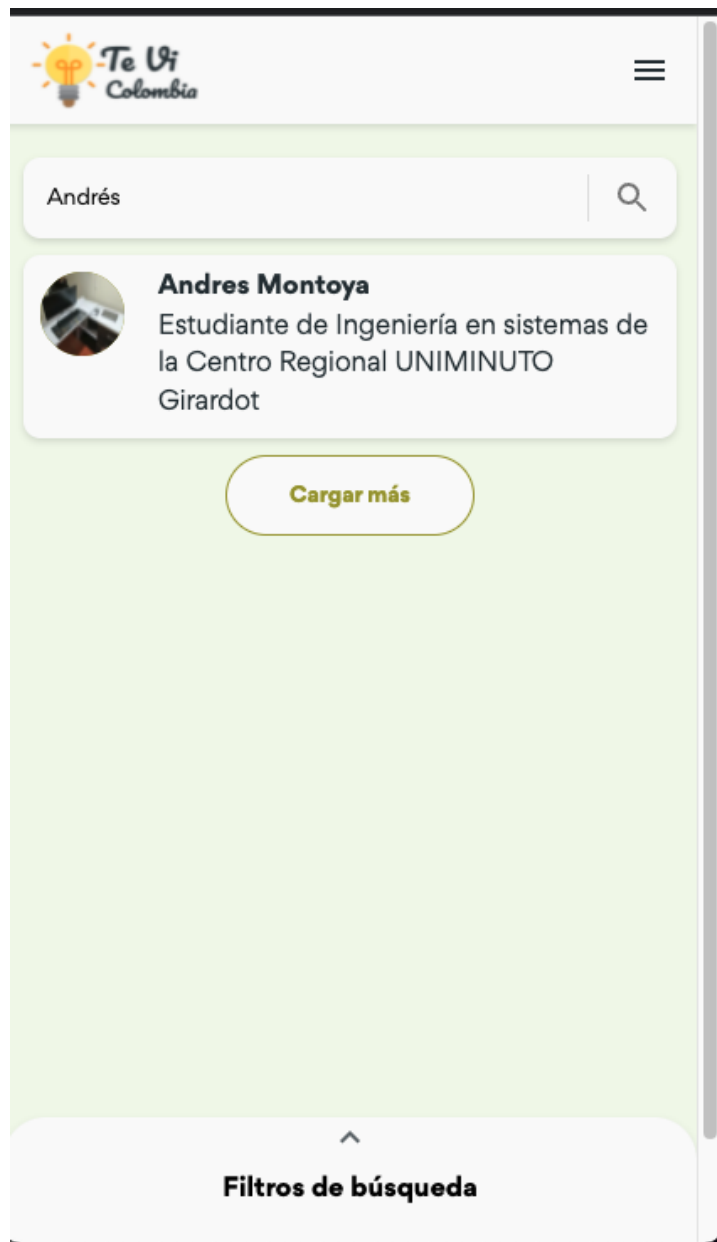
Ciudad
Girardot ▼

LIMPIAR

BUSCAR POR PRODUCTOS

Fuente: Autor

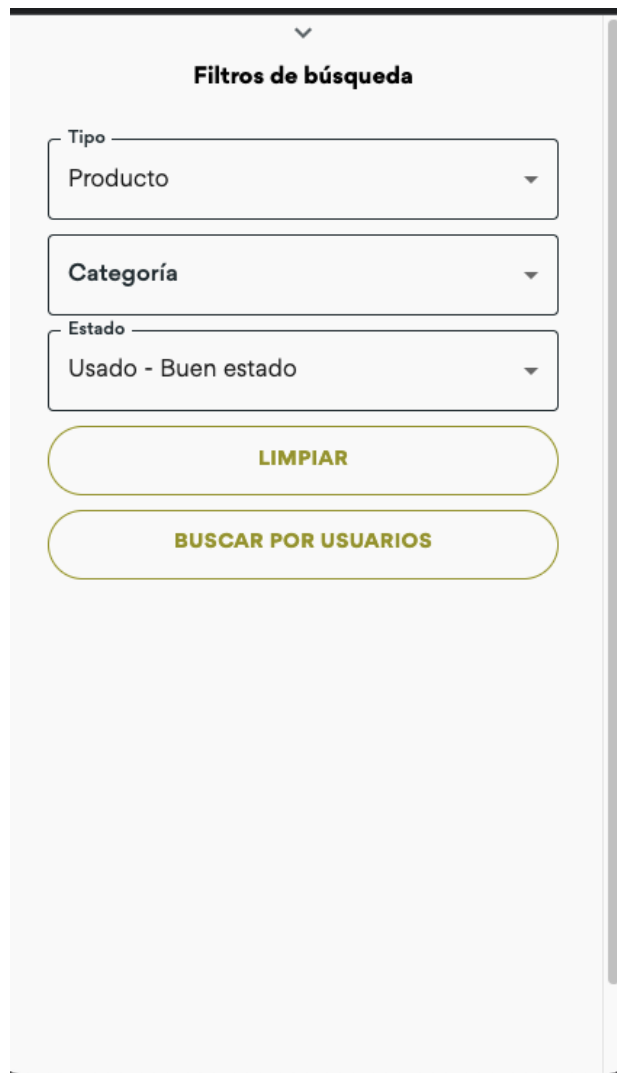
Textfield:



Fuente: Autor

Así de simple es, con relación a un producto, si queremos buscar una silla, que esté en un estado aceptable, simplemente sería:

Filtro de búsqueda:



The image shows a mobile application interface for search filters. At the top, there is a title "Filtros de búsqueda" with a small downward arrow icon. Below the title, there are three filter sections, each with a label and a dropdown menu:

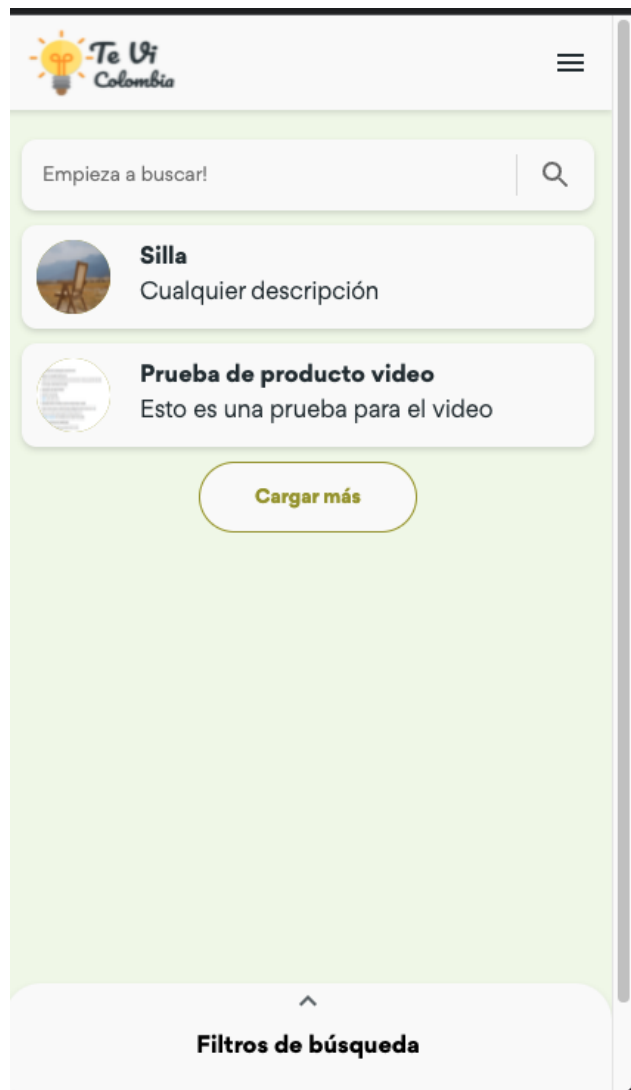
- Tipo**: The dropdown menu is set to "Producto".
- Categoría**: The dropdown menu is currently empty.
- Estado**: The dropdown menu is set to "Usado - Buen estado".

Below the filter sections, there are two buttons with rounded corners and a yellow border:

- A button labeled "LIMPIAR" (Clear).
- A button labeled "BUSCAR POR USUARIOS" (Search for users).

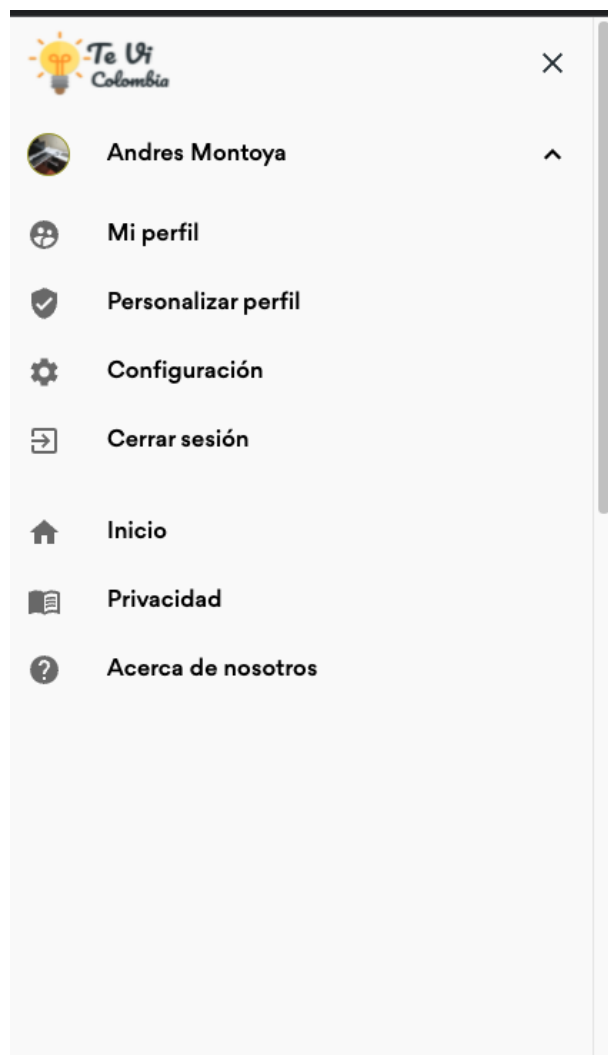
Fuente: Autor

Vemos que es el primer producto.



Fuente: Autor

MENU FUNCIONAMIENTO.



Fuente: Autor

Aquí podemos ver las siguientes opciones:

- Mi perfil: Información del perfil de usuario que actualmente ha iniciado sesión.
- Personalizar perfil: Sección donde el usuario puede editar o eliminar cualquier información que considere.
- Configuración: Sección para cambiar el correo electrónico o contraseña.
- Cerrar sesión: Terminar sesión de usuario.
- Inicio: Página de inicio.
- Privacidad: Página de privacidad.
- Acerca de nosotros: Encontrar más información acerca de Te Vi Colombia.


MI PERFIL.

Esta sección cuenta con mucha información por parte del usuario registrado, ya que en este se deriva información general, qué formación y empleo tuvo y tiene actualmente, qué perfil como profesional o emprendedor tiene y lista de productos.

Toda esta información se divide en:

- Información general.
 - Nombre.
 - Apellido.
 - Descripción.
 - Dirección.
 - Teléfono.
 - Departamento.
 - Municipio.
 - Nacionalidad.
 - Fecha de nacimiento.
 - Sitio web personal.
 - Genero.
 - Redes sociales.
 - Idiomas.

- Formación y empleo.
 - Estudio.
 - Universidad
 - Titulación
 - Disciplina académica
 - Fecha de inicio
 - Fecha de finalización
 - Actividades y grupos
 - Descripción
 - Trabajo.
 - Cargo que ocupa
 - Tipo de empleo
 - Compañía
 - Ubicación
 - Fecha de inicio
 - Fecha de finalización

- 
- Descripción
 - Productos
 - Artículo en venta
 - Lista de imagenes
 - Titulo
 - Precio
 - Categoría
 - Estado
 - Descripción
 - Vehículo en venta
 - Lista de imagenes
 - Tipo de vehículo
 - Precio
 - Ubicación
 - Año
 - Marca
 - Modelo
 - Descripción

Todo este contenido se divide entre secciones, las cuales dependiendo de las que más interés genere al usuario, se mostrará su respectiva información.



Fuente: Autor

PERSONALIZAR PERFIL.

En esta sección se puede cambiar toda la información que el usuario requiera, desde la información personal, estudios y trabajos.

Toda esta información se divide en:

- Información general

- Nombres.
- Apellidos.
- Descripción.
- Genero.
- Teléfono.
- Nacionalidad.
- Departamento.
- Ciudad.
- Dirección.
- Fecha de nacimiento.
- Redes sociales.
- Sitio web.
- Idiomas.
 - Idioma que maneja
 - Nivel del idioma
- Redes sociales
 - Tipo de red
 - Valor o URL de la red social
- Estudios
 - Universidad
 - Titulación
 - Disciplina académica
 - Fecha de inicio
 - Fecha de finalización
 - Actividades y grupos
 - Descripción
- Trabajos.
 - Cargo que ocupa
 - Tipo de empleo
 - Compañía
 - Ubicación
 - Fecha de inicio
 - Fecha de finalización
 - Descripción

Todo este contenido se divide entre secciones, las cuales dependiendo de las que más interés genere al usuario, se mostrará su respectiva información.



The image shows a mobile application interface for 'Te Uí Colombia'. At the top left is the logo, which consists of a lightbulb icon and the text 'Te Uí Colombia'. To the right of the logo is a hamburger menu icon. Below the header is the title 'Información general'. Underneath the title is a circular profile picture showing a person's face. Below the profile picture are several form fields: 'Nombres *' with the value 'Andres', 'Apellidos *' with the value 'Montoya', 'Describe quién eres y qué haces' with the text 'Estudiante de Ingeniería en sistemas de la Centro Regional UNIMINUTO Girardot', 'Genero' with a dropdown menu showing 'Hombre', 'Número de teléfono *' with the value '3213726060', and 'Departamento' with a dropdown menu showing 'Cundinamarca'.

Fuente: Autor

PRODUCTOS

Todos los usuarios tienen la posibilidad de comprar un producto, pero ofrece requiere de un paso extra. Cuando un usuario se registra por primera vez, este tiene que pasar por un proceso de validación, donde se pregunta lo siguiente en un formulario de Google.

- Nombres

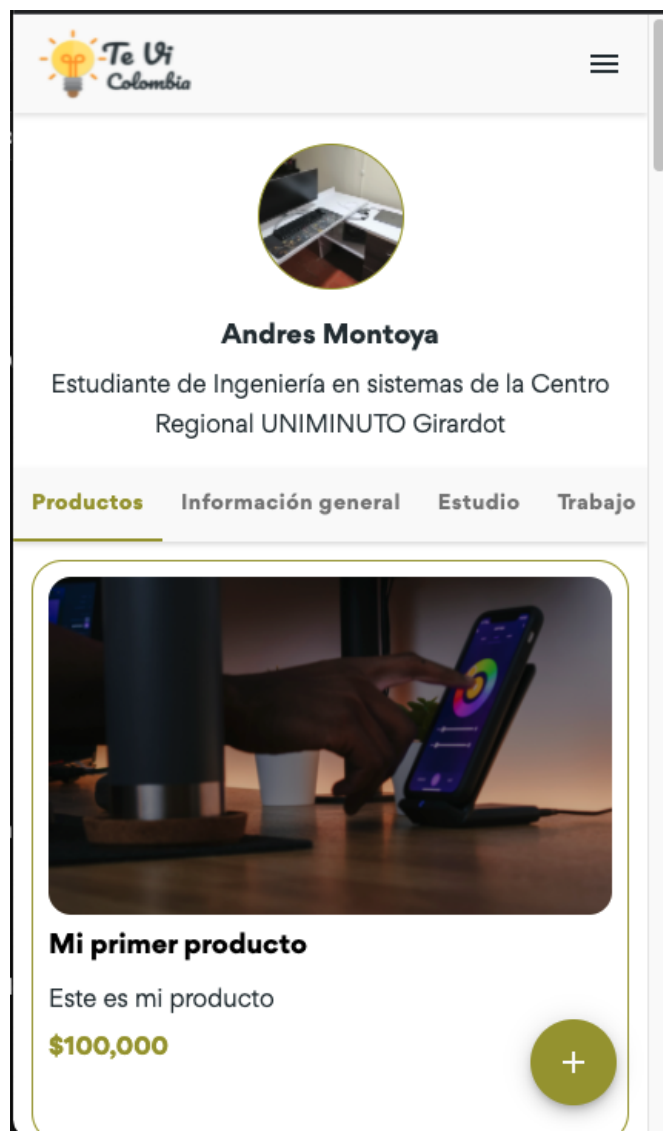
- Apellidos
- Edad
- Tipo de documento
- Número de documento
- Número de teléfono (Debe de estar registrado en Te Vi Colombia)
- Correo electrónico (Debe de estar registrado en Te Vi Colombia)
- ¿Cuál es su nombre de usuario en Te Vi Colombia?
- Nombre de Emprendimiento
- Descripción de Emprendimiento
- Página web
- Redes sociales
- Denominación tributaria
- RUT
- Clasificación de empresas (Si es persona jurídica)

Una vez esta información es validada, el campo de STATUS del usuario pasará a ACTIVE.

phoneN...	picture	socialNe...	state	status	updatedAt	website
3206090207	615015cc-b...	[{"M": {"t...	Cundinama...	ACTIVE	2022-03-0...	https://ww...
3213726060			Bogotá, D.C.	ACTIVE	2022-03-1...	
3213726060			Cundinama...	INACTIVE	2022-03-1...	
3213726060		[{"M": {"t...	Bogotá, D.C.	ACTIVE	2021-07-0...	
3213726060	48d5ab68-f...	[{"M": {"t...	Bogotá, D.C.	ACTIVE	2021-11-1...	https://res...
3213726060			Bogotá, D.C.	ACTIVE	2021-11-2...	
3123726061		[{"M": {"t...	Bogotá, D.C.	ACTIVE	2022-03-1...	
3134894611			Cundinama...	ACTIVE	2022-03-0...	
3205628080			Bogotá, D.C.	ACTIVE	2021-10-0...	
3213726060	5859fa12-9...	[{"M": {"t...	Cundinama...	ACTIVE	2022-03-0...	
3213526896			Bogotá, D.C.	ACTIVE	2022-02-2...	

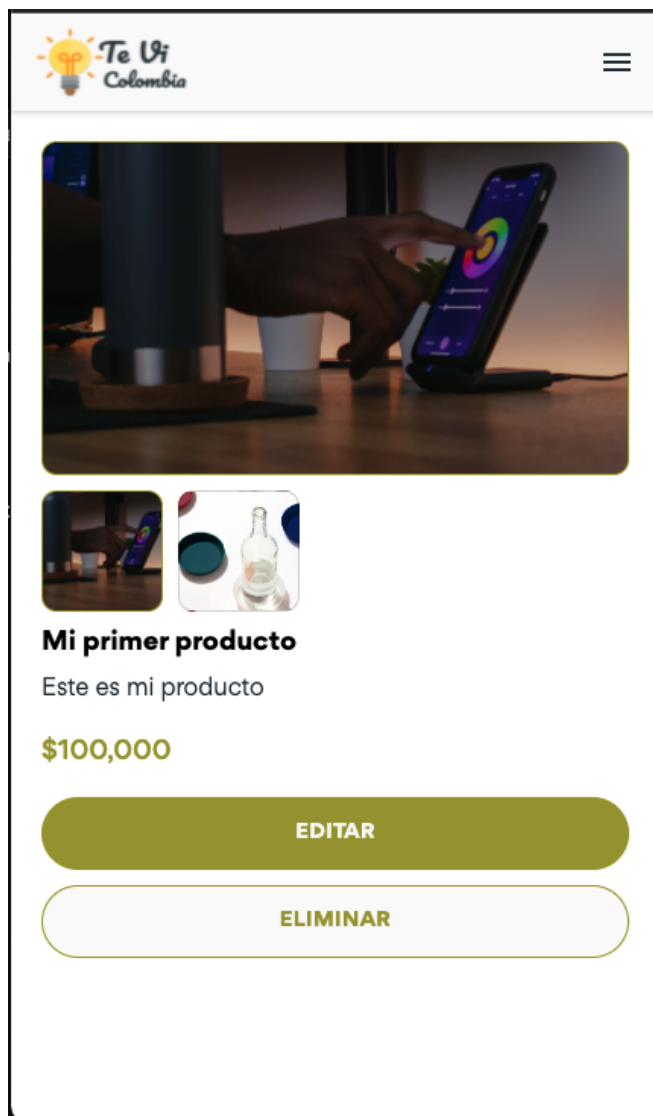
Fuente: Autor

Una vez hecho, el usuario podrá añadir productos.



Fuente: Autor

En la imagen anterior vemos el listado de productos, donde este debe ser:



Fuente: Autor

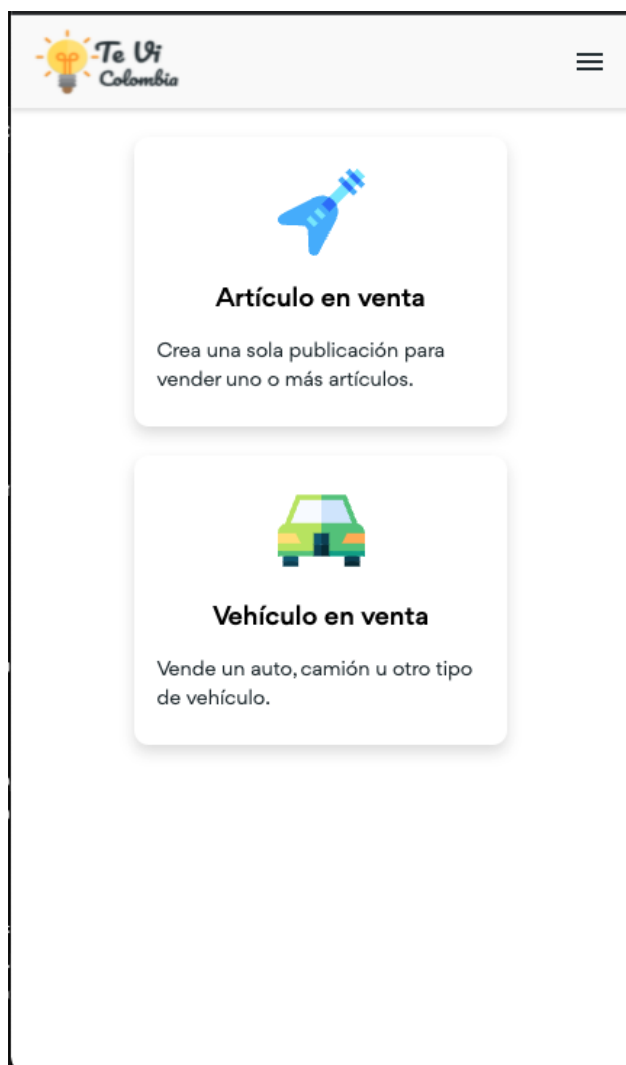
El usuario tiene la posibilidad de editar o eliminar este producto.



The screenshot shows a mobile application interface for adding a product. At the top, there is a header with a logo that says "Te U Colombia" and a hamburger menu icon. Below the header, there are two image thumbnails, each with a red 'X' in the top right corner, indicating they can be removed. Below the thumbnails is a green button with a camera icon and the text "Subir imagenes". Underneath is a section titled "Añade un producto" with several input fields: "Titulo *" containing "Mi primer producto", "Precio" containing "100000", "Categoría" with a dropdown menu showing "Libros", "Estado" with a dropdown menu showing "Nuevo", and "Descripción" containing "Este es mi producto". At the bottom of the form is a green button labeled "EDITAR".

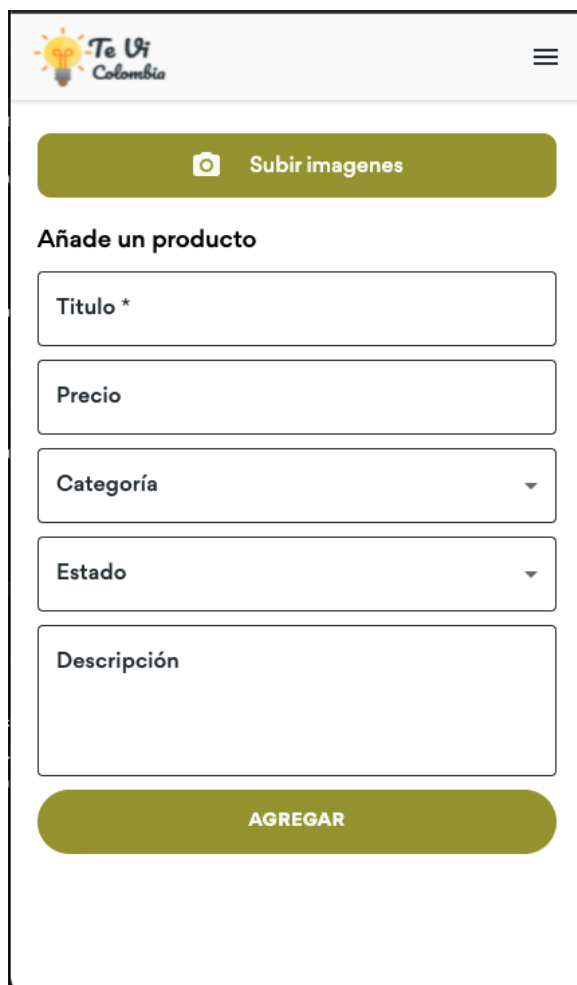
Fuente: Autor

Para crear un producto, el usuario tiene dos opciones, un artículo o un vehículo.



Fuente: Autor

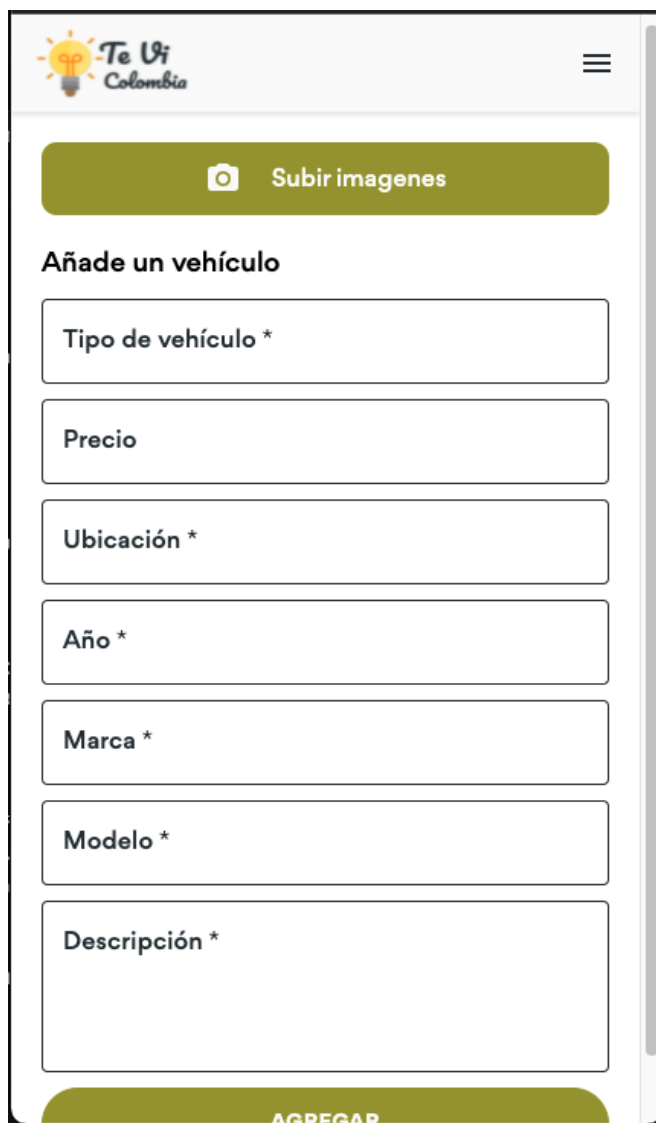
Para un artículo en venta, el usuario debe de ingresar los valores en pantalla.



The screenshot shows a mobile application interface for adding a product. At the top left is the logo for 'Te Uí Colombia' with a lightbulb icon. To the right of the logo is a hamburger menu icon. Below the header is a green button with a camera icon and the text 'Subir imagenes'. Underneath is the heading 'Añade un producto'. The form consists of several input fields: 'Titulo *', 'Precio', 'Categoría' (a dropdown menu), 'Estado' (a dropdown menu), and 'Descripción' (a larger text area). At the bottom of the form is a green button labeled 'AGREGAR'.

Fuente: Autor

Lo mismo es para un vehículo.



The screenshot shows the 'Te Vi Colombia' mobile application interface. At the top left is the logo with a lightbulb icon and the text 'Te Vi Colombia'. To the right is a hamburger menu icon. Below the header is a green button with a camera icon and the text 'Subir imagenes'. Underneath is the heading 'Añade un vehículo'. The form contains several input fields: 'Tipo de vehículo *', 'Precio', 'Ubicación *', 'Año *', 'Marca *', 'Modelo *', and 'Descripción *'. At the bottom of the form is a green button with the text 'AGREGAR'.

Fuente: Autor

Una vez hecho, automáticamente será listado en el buscador, y de una sola vez cualquier usuario puede directamente comprarlo.

ALCANCES FUTUROS

Te vi Colombia tiene la meta de ofrecer el servicio de la herramienta por un año para saber cómo esta influye en la ciudad de Girardot, Cundinamarca. Si esto genera resultados favorables, se puede llegar a la conclusión de expandir esta aplicación a otros lugares de Colombia. Así mismo, se seguirá trabajando en el desarrollo de esta,

agregando la funcionalidad de poder administrar la información de la plataforma o usuarios.

LIMITACIONES

Las limitaciones que se pueden encontrar en el desarrollo e implementación en el sistema de información son:

- Seguridad. Se está trabajando arduamente para que no existan este tipo de errores, para ello lo mejor es frecuentemente actualizar las dependencias de la aplicación.
- Aceptación. Esto puede tomar tiempo, ya que relativamente este software es nuevo y lleva poco tiempo en el mercado.

ESTRUCTURA DE SOFTWARE

BASE DE DATOS

La base de datos principal, donde se guarda la información de cada usuario está basada en el motor de base de datos NoSQL.

Donde la infraestructura es la siguiente:

```
NameTable:
  Type: AWS::DynamoDB::Table
  DeletionPolicy: Retain
  Properties:
    BillingMode: PAY_PER_REQUEST
    PointInTimeRecoverySpecification:
      PointInTimeRecoveryEnabled: true
  KeySchema:
    - AttributeName: id
      KeyType: HASH
  AttributeDefinitions:
    - AttributeName: id
```

```
AttributeType: S
StreamSpecification:
  StreamViewType: NEW_AND_OLD_IMAGES
Tags:
  - Key: Environment
    Value: ${self:custom.stage}
  - Key: Name
    Value: users-table
```

Esta es la estructura básica de cada tabla, ya se sube por cloudformation para la respectiva colocación del misma en AWS.

TABLA USER

Esta se encarga de guardar los usuarios de la aplicación, la primera vez que un usuario va a ingresar a Te Vi Colombia, es necesario que ingresa:

- Id
- Name
- Last Name
- Picture
- Description
- Gender
- Email
- Phonenumber
- Status
- State
- City
- Country
- Address
- Website
- Birthdate
- Languages
 - Level
 - Language
- Social Networks

- Type
- Source

Una vez hecho, simplemente se enviará un correo al usuario para saber si este no es un bot o ingreso un correo falso. Los demás datos de la tabla se llenan en la sección “Personalizar perfil”.

Siendo esta su configuración en GraphQL

```
type User {
  id: ID!
  name: String!
  lastName: String!
  picture: String
  description: String
  gender: String
  email: String!
  phoneNumber: String!
  status: String!
  state: String!
  city: String!
  country: String!
  address: String
  website: String
  birthdate: AWSDate
  languages: [Language]
  socialNetworks: [SocialNetwork]
  studies(limit: Int, nextToken: String): StudiesPage
  works(limit: Int, nextToken: String): WorksPage
  products(type: PRODUCT_TYPES, limit: Int, nextToken: String): ProductsPage
}

type Language {
  level: String!
  language: String!
}

type SocialNetwork {
```

```
type: String!  
source: String!  
}
```

Esta misma tabla está relacionada con Studies, Works y Products.

TABLA STUDY

Esta tabla guarda los datos de estudio de cada usuario, por medio de una relación One to Many a User.

```
type Study {  
  id: ID!  
  userId: ID!  
  place: String!  
  qualification: String  
  discipline: String  
  startedAt: Date!  
  finishedAt: Date  
  activity: String  
  description: String  
  createdAt: Date  
}
```

TABLA WORK

Guardar qué experiencias laborales tiene un usuario. Es una relación One To Many a User.

```
type Work {  
  id: ID!  
  userId: ID!  
  job: String!  
  contract: String!  
  company: String!  
  ublication: String!  
  startedAt: Date!
```

```
finishedAt: Date
description: String
createdAt: Date
}
```

TABLA DE PRODUCTOS

Con qué productos cuenta un usuario.

```
type Product{
  id: ID!
  userId: ID!
  type: PRODUCT_TYPES!
  pictures: [String]
  title: String
  price: Int
  category: String
  status: String
  description: String
  class: String
  location: String
  year: Int
  brand: String
  model: String
}

enum PRODUCT_TYPES {
  ITEM
  VEHICLE
}
```

Ya después de la formación de Schema en GraphQL, van los métodos de la aplicación.

```
type Query {
```

```

getUser(id: ID): User
getProduct(id: ID!, userId: ID): Product
searchProducts(input: SearchProductsInput!): SearchProductsPage
searchUsers(input: SearchUsersInput!): SearchUsersPage
}

type Mutation {
  createUser(input: CreateUserInput!): User!
  updateUser(input: UpdateUserInput!): User!
  createStudy(input: StudyInput!): OperationCompleted
  deleteStudy(id: ID!): Boolean
  createWork(input: WorkInput!): OperationCompleted
  deleteWork(id: ID!): Boolean
  createItem(input: CreateItemInput!): OperationCompleted
  updateItem(input: UpdateItemInput!): Product!
  createVehicle(input: CreateVehicleInput!): OperationCompleted
  updateVehicle(input: UpdateVehicleInput!): Product!
  deleteProduct(id: ID!): Boolean
  payment(input: PaymentInput!): String
  sendEmail(input: SendEmailInput!): Boolean
}

```

REDIS COMO GUARDADO DE SESSIONS.

Cada vez que un usuario entra a la aplicación, se genera una cookie, está cuenta con una sesión para poder diferenciar cada usuario, esta sesión es guardada en Redis.

Esta se puede guardar perfectamente en DynamoDB a través de una tabla nueva. Esta metodología es muy vieja y a largo plazo una gran mala idea, para evitar problemas y la identificación de usuarios sea eficaz, se usa esta In Memory database. Más adelante se explica su funcionalidad en la aplicación y como la API logra hacer este proceso.

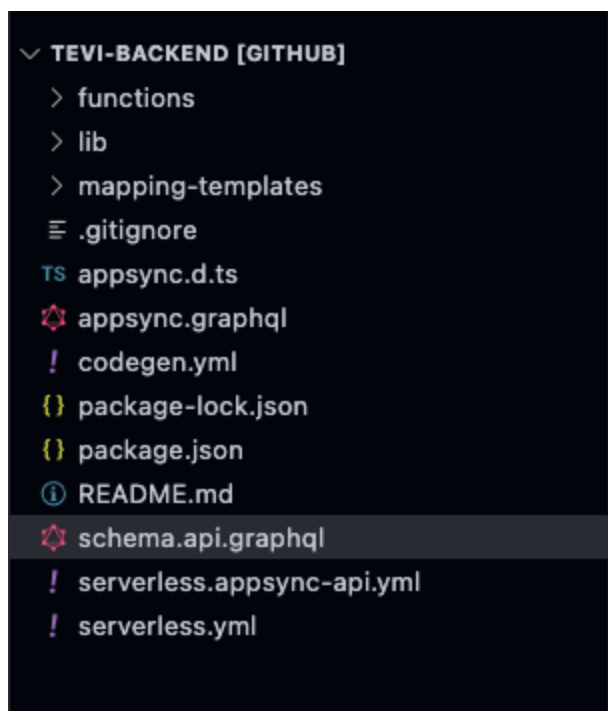
APPLICATION PROGRAMMING INTERFACE (A.P.I)

La A.P.I está basada en una tecnología llamada GraphQL, que es manejado por el runtime engine Node.js, a través de Lambdas y resolvers escritos en VTL. Este permite que el cliente consuma los datos que se necesitan, reduciendo el peso del response y ayudando a que no haya un overflow de información innecesaria. Cuenta con un

esquema sólido, confiable y bien definido que permite tener un excelente control en la aplicación, así mismo la experiencia de desarrollo es inigualable.

La estructura del proyecto se basa en que tenemos una carpeta llamada `src` la cual cuenta con todo el source de la A.P.I, esta se basa en que:

- **Schema:** Este es lo que tomará en cuenta GraphQL para la creación de la API, que metodos necesita un usuario para hacer cualquier acción
- **Cloudformation:** Toda la infraestructura que se defina en AWS, servicios como Cognito, S3, DynamoDB, Lambda, AppSync están implementados aquí.
- **Mapping-templates:** Todos los VTL que dependen únicamente de DynamoDB, esto para obtener, guardar, registrar, editar y eliminar cualquier información sin problemas de Loading.
- **Lib:** Configuración de librerías y métodos que se utilizan en las funciones.
- **Functions:** Listado de funciones que utiliza la api de GraphQL, según si la lógica es complicada, están se escriben en Node.js para su respectivo procesamiento de información.



Fuente: Autor

La primera implementación es el schema, que basado en lo explicado en las tablas anteriores, este define como la API tiene que estar conformada, este sería el siguiente:

```
type Query {
  getUser(id: ID!): User
  getProduct(id: ID!, userId: ID): Product
  searchProducts(input: SearchProductsInput!): SearchProductsPage
  searchUsers(input: SearchUsersInput!): SearchUsersPage
}

type Mutation {
  createUser(input: CreateUserInput!): User!
  updateUser(input: UpdateUserInput!): User!
  createStudy(input: StudyInput!): OperationCompleted
  deleteStudy(id: ID!): Boolean
  createWork(input: WorkInput!): OperationCompleted
  deleteWork(id: ID!): Boolean
  createItem(input: CreateItemInput!): OperationCompleted
  updateItem(input: UpdateItemInput!): Product!
  createVehicle(input: CreateVehicleInput!): OperationCompleted
  updateVehicle(input: UpdateVehicleInput!): Product!
  deleteProduct(id: ID!): Boolean
  payment(input: PaymentInput!): String
  sendEmail(input: SendEmailInput!): Boolean
}
```

Todos los métodos que utiliza, estos toman unos inputs, serían:

```
input LanguageInput {
  level: String!
  language: String!
}

input SocialNetworkInput {
  type: String!
  source: String!
}

input CreateUserInput {
  name: String!
```

```
lastName: String!  
email: AWSEmail!  
phoneNumber: AWSPhone!  
state: String!  
city: String!  
country: String!  
}  
  
input UpdateUserInput {  
  name: String  
  lastName: String  
  picture: String  
  description: String  
  gender: String  
  phoneNumber: AWSPhone  
  state: String  
  city: String  
  country: String  
  address: String  
  website: AWSURL  
  birthdate: AWSDate  
  languages: [LanguageInput]  
  socialNetworks: [SocialNetworkInput]  
}  
  
input StudyInput {  
  place: String!  
  qualification: String  
  discipline: String  
  startedAt: AWSDate!  
  finishedAt: AWSDate  
  activity: String  
  description: String  
}  
  
input WorkInput {  
  job: String!  
  contract: String!
```

```
    company: String!
    location: String!
    startedAt: AWSDate!
    finishedAt: AWSDate
    description: String
  }

input CreateItemInput {
  pictures: [String!]!
  title: String!
  price: Int
  category: String!
  status: String!
  description: String
}

input UpdateItemInput {
  id: ID!
  pictures: [String]
  title: String
  price: Int
  category: String
  status: String
  description: String
}

input CreateVehicleInput {
  class: String!
  pictures: [String!]!
  location: String!
  year: Int!
  brand: String!
  model: String!
  price: Int
  description: String
}

input UpdateVehicleInput {
```

```
id: ID!
class: String
pictures: [String]
location: String
year: Int
brand: String
model: String
price: Int
description: String
}

input SearchProductsInput {
  value: String!
  page: Int!
  type: PRODUCT_TYPES!
  category: String
  status: String
}

input SearchUsersInput {
  value: String!
  page: Int!
  country: String!
  state: String
  city: String
}

input PaymentInput {
  productId: ID!
  userId: ID!
}

input SendEmailInput {
  to_email: String!
  subject: String!
  html: String!
}
```

```
enum PRODUCT_TYPES {  
  ITEM  
  VEHICLE  
}
```

A partir de ahí se toman los modelos de la base de datos para devolver los valores, esto para que el Front-end tenga la información que necesita, sin ningún inconveniente en la petición.

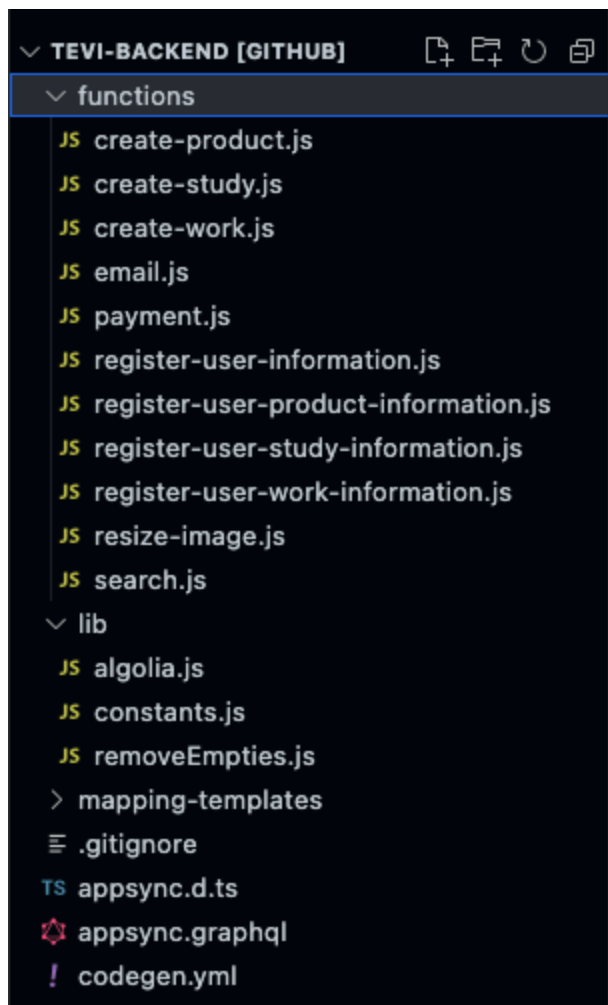
Mapping-templates



Fuente: Autor

Hay mutaciones y queries que requieren simplemente una conexión básica a la base de datos, es por ello que estos están definidos, sus operaciones por lo regular son CRUD que con un buen performance hacen su tarea.

Functions and Libs



Fuente: Autor

Cuando se requiere trabajar en lógica compleja, están las lambdas, donde se centran en optimizar procesos complejos en la base de datos y utilizar librerías o configuraciones extras para cumplir con los objetivos. Conexiones como Stripe, SendGrid, Algolia, y muchas otras se hacen a través de lambda, para mantener la permanencia de lógica e implementación en la RED de la API.


SINGLE PAGE APPLICATION

Todo lo que ve el usuario es completamente reactivo, haciendo que la experiencia de usuario sea completamente única, para lograr este cometido se uso la Librería React.js junto a Next.js.

React.js es una librería escrita en JavaScript desarrollada por Facebook para facilitar la creación de componentes interactivos, reutilizables, para interfaces de usuarios. En cambio Next.js es un complemento para esta biblioteca, ya que añade Server Side Rendering (SSR) y SEO, mejorando el uso cotidiano de React, esta herramienta fue hecha por la empresa Zeit. Así mismo, como también tiene la funcionalidad de un aplicativo móvil, está hecho en Ionic.

La estructura que sigue este proyecto está en la carpeta `client`, la cual cuenta con las siguientes carpetas:

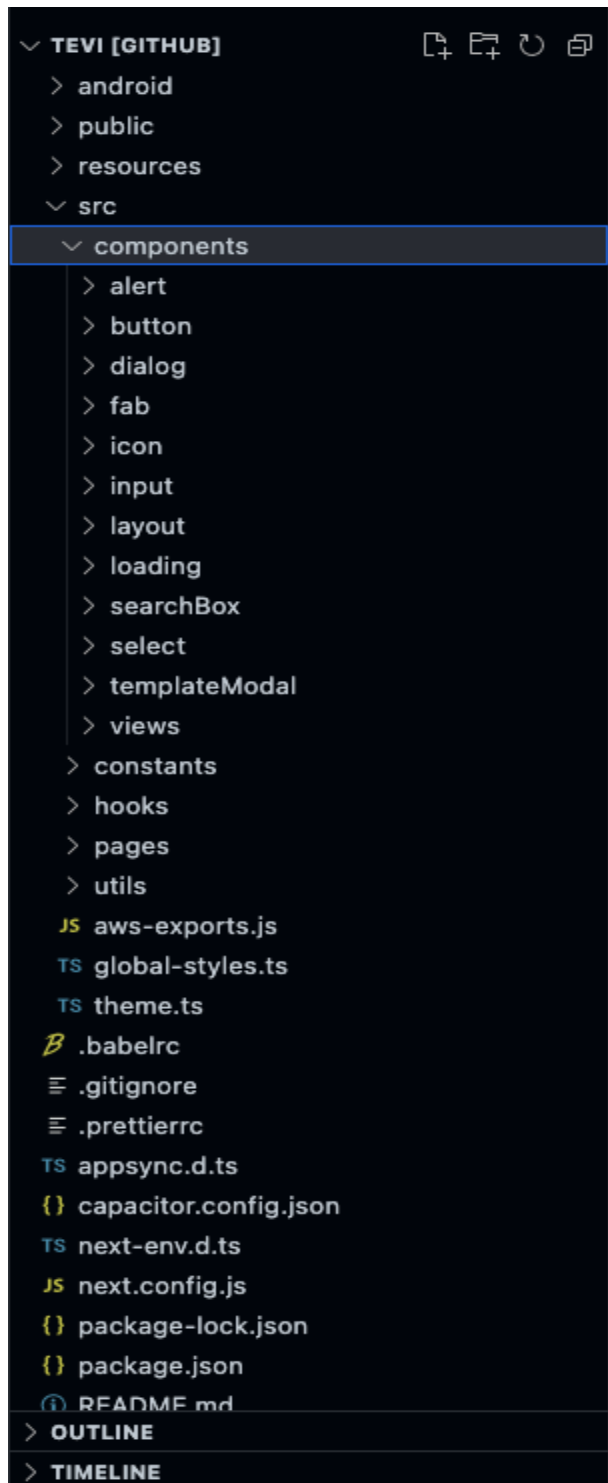
- **Components:** Como su nombre lo indica, son los componentes que usa la aplicación, estos están divididos de las funcionalidad que cumple una empresa o usuario, así mismo el Layout y componentes compartidos (reutilizables).
- **GraphQL:** Son aquellas que se utilizan más de una vez en diferentes componentes.
- **Lib:** Los datos que genera Apollo GraphQL tienen que ser devueltos por la vista, de una forma en la que soporte Server Side Rendering y a la vez no genere problemas, lo que se encuentra en esta carpeta son scripts que cumplen esta función, así mismo también son utilizados para la autorización de usuarios.
- **Pages:** Next.js utiliza esta carpeta para identificar qué rutas tendrá la aplicación, y que código tendrá el componente de esta ruta.
- **Static:** Son archivos estáticos que utiliza la aplicación, como archivos CSS, imágenes, y el `manifest.json` que necesita el PWA de Te Vi Colombia.
- **Utils:** Simplemente son los archivos JSON que necesita los input selects, la validación de los diferentes inputs, y un archivo con nombre `normalizeErrors` que ayuda a normalizar los errores que devuelve la A.P.I para que sean compatibles con Formik.
- **Demás archivos:** Configuración de Ionic para la creación de la aplicación en Android, esta toma todo el código ya transpirado, para pasarlo a una APK junto a toda las Environment variables y demás configuraciones que son necesarias para el desarrollo de la misma.




```
✓ TEVI [GITHUB]
  > android
  > public
  > resources
  > src
  ℘ .babelrc
  ≡ .gitignore
  ≡ .prettierrc
  TS appsync.d.ts
  {} capacitor.config.json
  TS next-env.d.ts
  JS next.config.js
  {} package-lock.json
  {} package.json
  ⓘ README.md
  TS styled.d.ts
  ≡ tevi.jks
  TS tsconfig.json
  {} vercel.json
```

Fuente: Autor

La primera carpeta es Components, esta carpeta trae diferentes carpetas con más componentes que cumplen diferentes funciones.



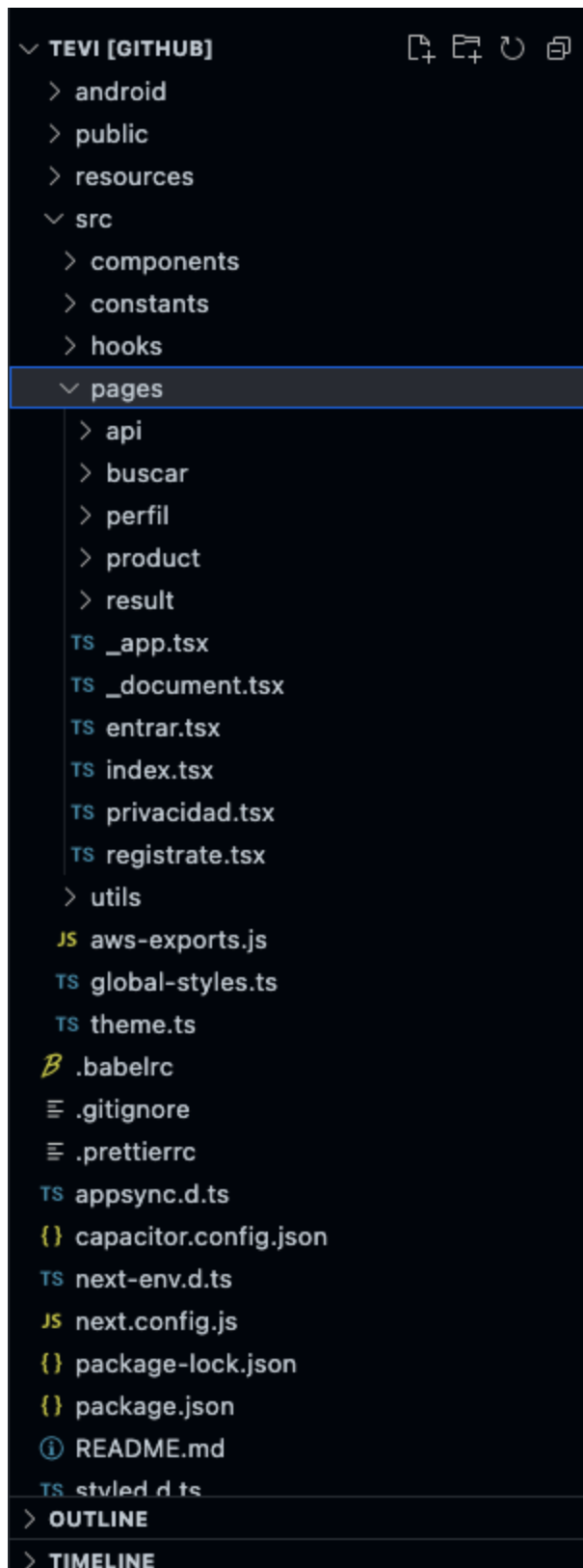
Fuente: Autor




Si ya ha leído el manual de usuario, o ya ha manejado la aplicación Te Vi Colombia y ha usado todas sus funcionalidades, comprender este listado no será difícil, ya que aquellas funcionalidades que se repitan más de una vez entre usuarios y productos, se encuentran con su propia carpeta. En el caso de Shared son componentes que se comparten entre sus antecesores, un ejemplo podría ser los input que tiene Te Vi Colombia, todos estos son un solo componente que se reutiliza en muchos lados.

Los demás archivos, que son Home y Search simplemente son todo lo que se ve en el inicio, el buscador, el formulario de ayuda, la lista de los mejores usuarios o emprendedores, etc... En el caso de Layout, es el footer, el header, entre otros que se ven en todas las secciones de Te Vi Colombia.

En `Pages` son las rutas que cuenta Te Vi Colombia, con su respectivo estilo y estado:



Fuente: Autor



El archivo `__app.tsx` se usa para globalizar la configuración de Apollo GraphQL y configurar el Layout. En `__document.tsx` es simplemente el `` del HTML que toda típica aplicación tiene.

Bibliografía:

Montoya Sánchez. (2019). *MANUAL TÉCNICO*. Girardot, Colombia.

<https://www.uniminuto.edu/biblioteca>