

**UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS**

**FACULTAD DE INGENIERÍA**

**DIVISIÓN DE ESTUDIOS PROFESIONALES PARA EJECUTIVOS**

**CARRERA DE INGENIERÍA DE SISTEMAS**

**SISTEMA INFORMATICO ESTACIONA AQUÍ**

**PROYECTO PROFESIONAL PRESENTADO POR NOMBRE\_EQUIPO:**

**“GEMAS”**

**NOMBRE DEL(LOS) AUTOR(ES)**

Eder Lizardo Cuestas Morales U201821408

Rosangela Yarina Espinoza Calderón U201821480

Julián Alberto Luque Vilcapoma U201723272 (Coordinador)

Johann Gerardo Miranda Cárdenas U201720902

Luis Antonio Padilla Gonzales U201823593

**PARA EL CURSO DE DESARROLLO PARA ENTORNO WEB**

**PROFESOR:**

**ING. Jaiver James Huiza Pereyra**

**Lima, 10 de agosto del 2018**

Fecha: 10-08-2018

Sección: C31A

Número de Grupo: 01

Correo del coordinador: alberto.luque.vilcapoma@gmail.com

U201723272@upc.edu.pe

Alcance del Proyecto:

El sistema estará alojado en un host que soporte Ruby on Rails y MySQL /SQL Server. Para la programación se usará HTML, CSS, Jquery/Javascript/CofeeScript para entrada de datos, botones y demás apariencias dinámicas. Habrá un área de publicidad de acuerdo al tipo de usuario que ingrese. La aplicación será responsive. Los usuarios podrán calificar con estrellas (0-5) y comentar su experiencia. Los usuarios podrán comentar la experiencia de otros. Los propietarios deben responder a quejas e informar sobre las acciones que tomarán al respecto. La aplicación es un sitio web que se accede a través del navegador. La primera página informa del producto de forma directa y resumida (podría ser un imagen). Claramente habrá 3 botones: Para ingresar un usuario registrado, usuario invitado o para registrar un usuario nuevo. Si anteriormente el usuario había ingresado, el sistema ingresará automáticamente si se habilitó la opción correspondiente en el setup de usuario. Si ingresa como usuario invitado, no pedirá información pero el sistema podría recopilar datos sobre ubicación, número IP u otro que esté disponible en su navegador, siempre conforme a la ley de protección de datos personales.

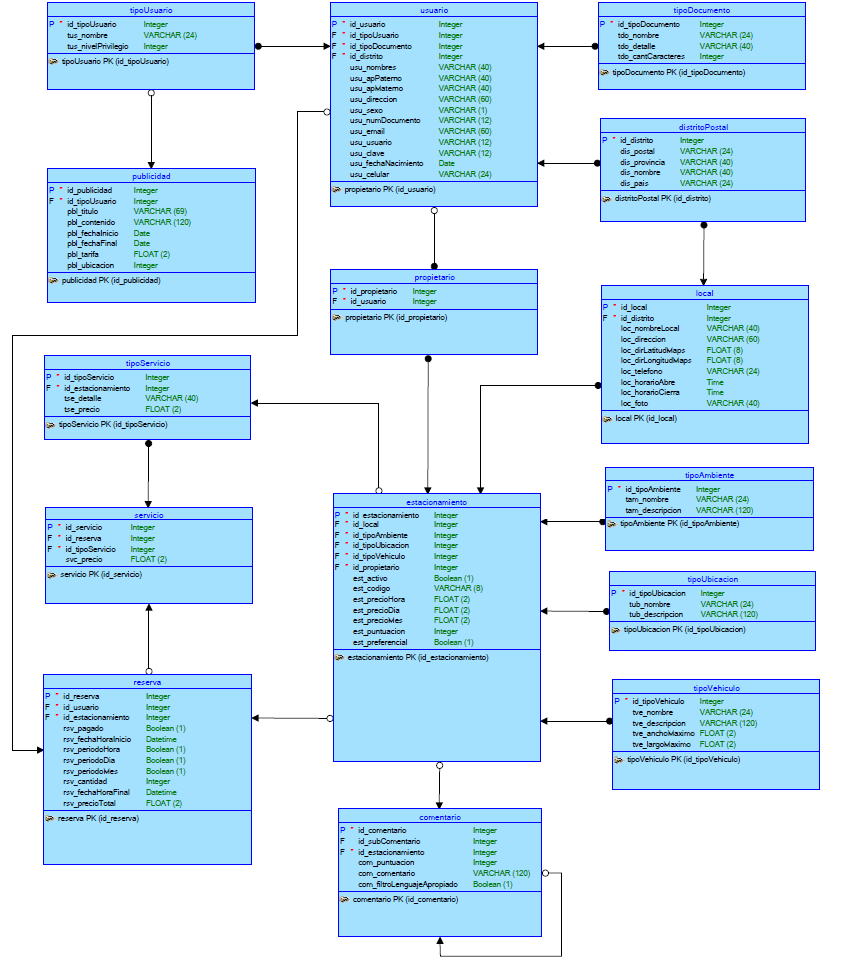
Los usuarios pueden ser: Registrado, Invitado, Propietario y web Master. Sólo el Web Master puede hacer CRUD (Create, Read, Update, Delete) a la tabla de Usuarios propietarios. Un usuario Invitado tiene la opción de leer los locales de estacionamientos disponibles actuales según el local más cerca de su ubicación o al distrito elegido. No tendrá habilitada la opción de reservar, pero podrán informarse de la tarifa. Los usuarios registrados pueden ver los estacionamientos libres, reservarlos y pagarlos\*. (Podría aplicar un descuento a la tarifa). Pueden también reservar estacionamiento para otro día. Los propietarios pueden hacer CRUD a los estacionamientos que son propietarios. El Web Master administra la aplicación, la base de datos y la publicidad. En el menú de Setup debe permitir la configuración de simplicidad de uso, es decir, se debe prevenir cargar de información a las páginas porque el usuario podría usarla cuando está manejando. La advertencia de uso cuando se maneja debe indicarse de forma clara y precisa. Los layout configurables podrían ser dos: Resumido y estándar.

Adicionales por investigar:

* Conexión con servicio/aplicación GoogleMaps/Waze para que el sistema le guíe la ruta más corta a su destino (estacionamiento).
* Captura de ubicación del celular para que sea efectiva lo anterior.
* Manejo por voz.
* \*Manejo de pagos.

Enlace al repositorio de control de versiones en Git

https://github.com/AlbertoLuque/parqueo

**MODELO DE DATOS** 

-- parkApp\_v7.doc

-- Sql para generar la base de datos de la aplicación ESTACIONA AQUÍ

--

if DB\_ID('parkApp') is not null

begin

use master

drop database parkApp

end

--Crear la base de datos en blanco

create database parkApp

go

use parkApp

go

create table comentario

(

id\_comentario int identity(1,1) NOT NULL ,

id\_estacionamiento int NOT NULL ,

com\_puntuacion int ,

com\_comentario varchar(120) ,

com\_filtroLenguajeApropiado BIT

)

go

alter table comentario

add constraint comentario\_PK primary key clustered (id\_comentario)

go

create table distritoPostal

(

id\_distrito int identity(1,1) NOT NULL ,

dis\_postal varchar(24) ,

dis\_provincia varchar(40) ,

dis\_nombre varchar(40) ,

dis\_pais varchar(24)

)

GO

alter table distritoPostal

add constraint distritoPostal\_PK primary key clustered (id\_distrito)

go

create table estacionamiento

(

id\_estacionamiento int identity(1,1) NOT NULL ,

id\_local int NOT NULL ,

id\_tipoAmbiente int NOT NULL ,

id\_tipoUbicacion int NOT NULL ,

id\_tipoVehiculo int NOT NULL ,

id\_propietario int NOT NULL ,

est\_codigo varchar(8) ,

est\_precioHora float (2) ,

est\_precioDia float (2) ,

est\_precioMes float (2) ,

est\_puntuacion int ,

est\_activo BIT NOT NULL ,

est\_preferencial BIT NOT NULL

)

GO

alter table estacionamiento

add constraint estacionamiento\_PK primary key clustered (id\_estacionamiento)

go

create table local

(

id\_local int identity(1,1) NOT NULL ,

id\_distrito int NOT NULL ,

loc\_nombreLocal varchar(40) ,

loc\_direccion varchar(60) ,

loc\_dirLatitudMaps float (8) ,

loc\_telefono varchar(24) ,

loc\_foto varchar(40) ,

loc\_dirLongitudMaps float (8) ,

loc\_horarioAbre DATETIME ,

loc\_horarioCierra DATETIME

)

go

alter table local

add constraint local\_PK primary key clustered (id\_local)

go

create table propietario

(

id\_propietario int identity(1,1) NOT NULL ,

id\_usuario int NOT NULL

)

GO

alter table propietario

add constraint propietario\_PK primary key clustered (id\_propietario)

go

create table publicidad

(

id\_publicidad int identity(1,1) NOT NULL ,

id\_tipoUsuario int NOT NULL ,

pbl\_titulo varchar(60) ,

pbl\_contenido varchar(120) ,

pbl\_fechaInicio DATETIME ,

pbl\_fechaFinal DATETIME ,

pbl\_tarifa float (2) ,

pbl\_ubicacion int

)

go

alter table publicidad

add constraint publicidad\_PK primary key clustered (id\_publicidad)

go

create table reserva

(

id\_reserva int identity(1,1) NOT NULL ,

id\_usuario int NOT NULL ,

id\_estacionamiento int NOT NULL ,

rsv\_pagado BIT ,

rsv\_fechaHoraInicio DATETIME ,

rsv\_periodoHora BIT ,

rsv\_periodoDia BIT ,

rsv\_periodoMes BIT ,

rsv\_cantidad int ,

rsv\_fechaHoraFinal DATETIME ,

rsv\_precioTotal float (2)

)

go

alter table reserva

add constraint reserva\_PK primary key clustered (id\_reserva)

go

create table servicio

(

id\_servicio int identity(1,1) NOT NULL ,

id\_reserva int NOT NULL ,

id\_tipoServicio int NOT NULL ,

svc\_precio float (2)

)

go

alter table servicio add constraint servicio\_PK primary key clustered (id\_servicio)

go

create table tipoAmbiente

(

id\_tipoAmbiente int identity(1,1) NOT NULL ,

tam\_nombre varchar(24) ,

tam\_descripcion varchar(120)

)

GO

alter table tipoAmbiente

add constraint tipoAmbiente\_PK primary key clustered (id\_tipoAmbiente)

go

create table tipoDocumento

(

id\_tipoDocumento int identity(1,1) NOT NULL ,

tdo\_nombre varchar(24) ,

tdo\_detalle varchar(40) ,

tdo\_cantCaracteres int

)

GO

alter table tipoDocumento

add constraint tipoDocumento\_PK primary key clustered (id\_tipoDocumento)

go

create table tipoServicio

(

id\_tipoServicio int identity(1,1) NOT NULL ,

id\_estacionamiento int NOT NULL ,

tse\_detalle varchar(40) ,

tse\_precio float (2)

)

GO

alter table tipoServicio

add constraint tipoServicio\_PK primary key clustered (id\_tipoServicio)

go

create table tipoUbicacion

(

id\_tipoUbicacion int identity(1,1) NOT NULL ,

tub\_nombre varchar(24) ,

tub\_descripcion varchar(120)

)

GO

alter table tipoUbicacion

add constraint tipoUbicacion\_PK primary key clustered (id\_tipoUbicacion)

go

create table tipoUsuario

(

id\_tipoUsuario int identity(1,1) NOT NULL ,

tus\_nombre varchar(24) ,

tus\_nivelPrivilegio int

)

GO

alter table tipoUsuario

add constraint tipoUsuario\_PK primary key clustered (id\_tipoUsuario)

go

create table tipoVehiculo

(

id\_tipoVehiculo int identity(1,1) NOT NULL ,

tve\_nombre varchar(24) ,

tve\_descripcion varchar(120) ,

tve\_anchoMaximo float (2) ,

tve\_largoMaximo float (2)

)

GO

alter table tipoVehiculo

add constraint tipoVehiculo\_PK primary key clustered (id\_tipoVehiculo)

go

create table usuario

(

id\_usuario int identity(1,1) NOT NULL ,

id\_tipoUsuario int NOT NULL ,

id\_tipoDocumento int NOT NULL ,

id\_distrito int NOT NULL ,

usu\_nombres varchar(40) ,

usu\_apPaterno varchar(40) ,

usu\_apMaterno varchar(40) ,

usu\_sexo varchar(1) ,

usu\_numDocumento varchar(12) ,

usu\_email varchar(60) ,

usu\_usuario varchar(12) ,

usu\_clave varchar(12) ,

usu\_fechaNacimiento DATETIME ,

usu\_celular varchar(24) ,

usu\_direccion varchar(60)

)

GO

alter table usuario

add constraint usuario\_PK primary key clustered (id\_usuario)

go

-- Recursive id\_comentario with id\_sebComentario

/\*

alter table comentario

add constraint comentario\_comentario\_FK foreign key (id\_subComentario)

references comentario (id\_comentario)

go

\*/

alter table comentario

add constraint comentario\_estacionamiento\_FK foreign key (id\_estacionamiento)

references estacionamiento (id\_estacionamiento)

go

alter table estacionamiento

add constraint estacionamiento\_local\_FK foreign key (id\_local)

references local (id\_local)

go

alter table estacionamiento

add constraint estacionamiento\_propietario\_FK foreign key (id\_propietario)

references propietario (id\_propietario)

go

alter table estacionamiento

add constraint estacionamiento\_tipoAmbiente\_FK foreign key (id\_tipoAmbiente)

references tipoAmbiente (id\_tipoAmbiente)

go

alter table estacionamiento

add constraint estacionamiento\_tipoUbicacion\_FK foreign key (id\_tipoUbicacion)

references tipoUbicacion (id\_tipoUbicacion)

go

alter table estacionamiento

add constraint estacionamiento\_tipoVehiculo\_FK foreign key (id\_tipoVehiculo)

references tipoVehiculo (id\_tipoVehiculo)

go

alter table local

add constraint local\_distritoPostal\_FK foreign key (id\_distrito)

references distritoPostal (id\_distrito)

go

alter table propietario

add constraint propietario\_usuario\_FK foreign key (id\_usuario)

references usuario (id\_usuario)

go

alter table publicidad

add constraint publicidad\_tipoUsuario\_FK foreign key (id\_tipoUsuario)

references tipoUsuario (id\_tipoUsuario)

go

alter table reserva

add constraint reserva\_estacionamiento\_FK foreign key (id\_estacionamiento)

references estacionamiento (id\_estacionamiento)

go

alter table reserva

add constraint reserva\_usuario\_FK foreign key (id\_usuario)

references usuario (id\_usuario)

go

alter table servicio

add constraint servicio\_reserva\_FK foreign key (id\_reserva)

references reserva (id\_reserva)

go

alter table servicio

add constraint servicio\_tipoServicio\_FK foreign key (id\_tipoServicio)

references tipoServicio (id\_tipoServicio)

go

alter table usuario

add constraint usuario\_distritoPostal\_FK foreign key (id\_distrito)

references distritoPostal (id\_distrito)

go

alter table usuario

add constraint usuario\_tipoDocumento\_FK foreign key (id\_tipoDocumento)

references tipoDocumento (id\_tipoDocumento)

go

alter table usuario

add constraint usuario\_tipoUsuario\_FK foreign key (id\_tipoUsuario)

references tipoUsuario (id\_tipoUsuario)

go

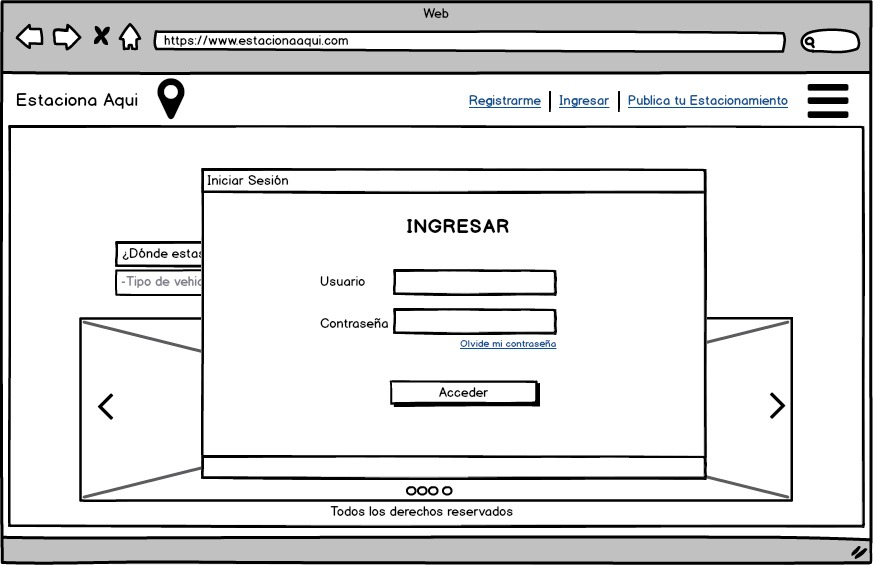
alter table tipoServicio

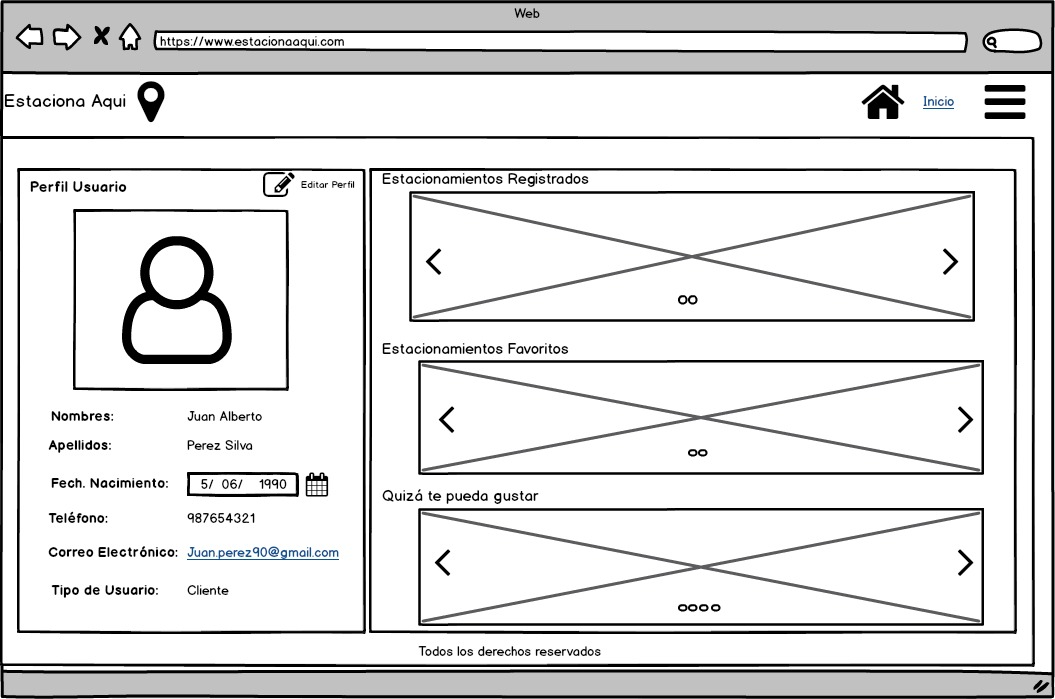
add constraint tipoServicio\_estaconamiento\_FK foreign key (id\_estacionamiento)

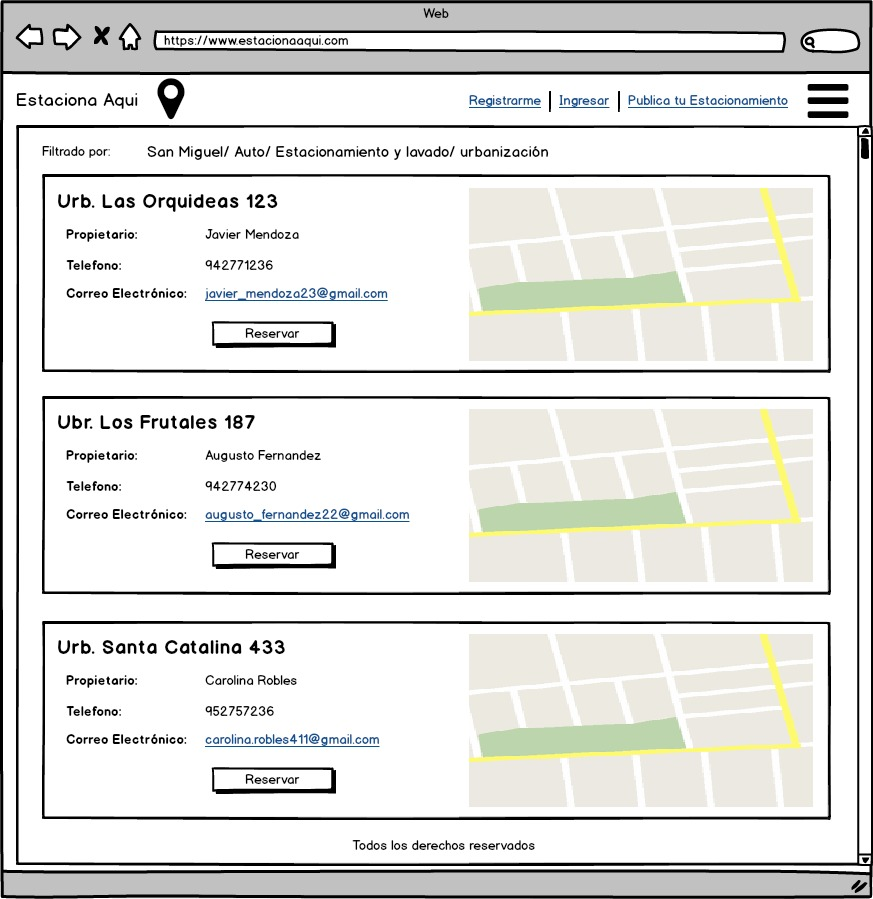
references estacionamiento (id\_estacionamiento)

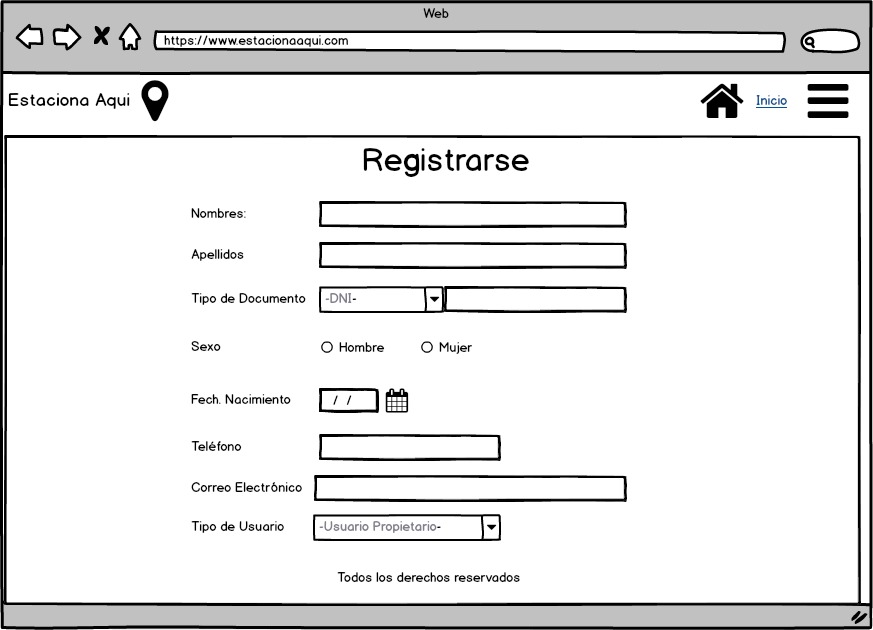
go

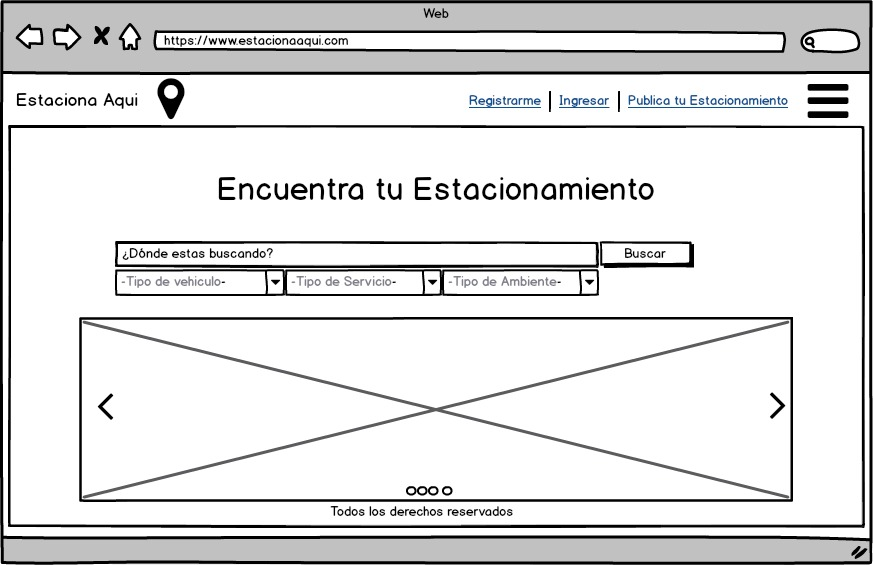
**PROTOTIPO**











Primera versión desarrollada en el aula

