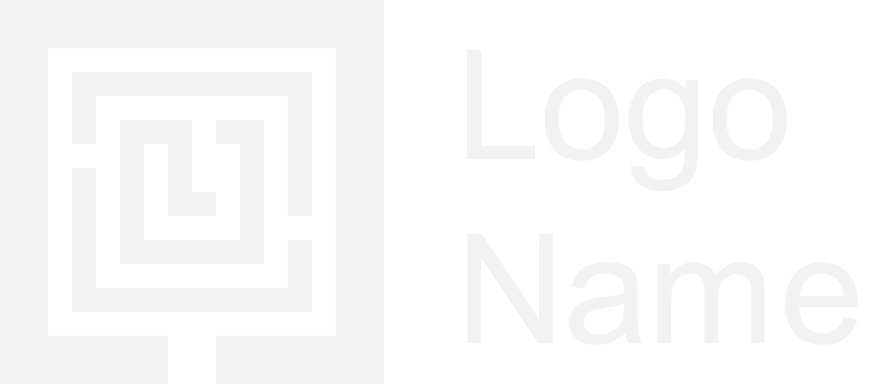
|  |
| --- |
| COURSE PROJECT  FINAL REPORT  Quintetu Company |
| Team Members  Segura Martinez, Johan Manuel u201910377  Daniel Ulises, Barrionuevo Gutiérrez u201922128  Gianpiero Jesus, Flores Tineo u20211B796  Gianpool Greg Pacheco Salambay u202117419 |



**Universidad Peruana de Ciencias Aplicadas**

Ingeniería de Sistemas y Computación | Ingeniería de Software

CC52 | Ingeniería de Software

Ciclo 2022-01

CONTENT

￼

￼

￼

c

CAPITULO 1: INTRODUCCIÓN

## Startup Profile

### Descripción de la Startup

## Somos Quintetu Company, un grupo formado por estudiantes de la Universidad Peruana de ciencias aplicadas para brindar soluciones tecnológicas a problemáticas existentes. Apasionados por la tecnología nos unimos y servimos a favor de la comunidad

## Misión: Brindar apoyo, seguridad, confianza y desarrollo a nuestros clientes

## Visión: Buscamos ser la empresa peruana líder en soluciones tecnologías que otorguen un impacto positivo a la comunidad

## Nuestro producto principal, y del que hablaremos a continuación, es “Auto Fast” un aplicativo móvil que permite tanto a los trabajadores en el rubro de alquileres de automóviles como también a las personas que necesiten de un auto para su transporte una facilidad en la obtención de algún o algunos vehículos. En el primer público, se enfoca en la parte logística y digitalización de su empresa mientras que, en el segundo, la practicidad del alquiler.

### Perfiles de integrantes del equipo

|  |
| --- |
| Johan Manuel Segura Martínez  Ingeniería de Software |

|  |  |
| --- | --- |
| **DESCRIPCION PERSONAL**    Soy Johan Manuel Segura Martinez y tengo 20 años. Nací en Lima y me encuentro cursando el 3er ciclo de la carrera de ingeniería de software en la Universidad Peruana de Ciencias Aplicadas (UPC).    La carrera de software me apasiona porque me gusta desarrollar aplicaciones que resuelvan necesidades reales. Mis hobbies son tocar el piano, la guitarra y aprender programación. |  |
| **COMPROMISO**    Me comprometo a dar todo de mi en este curso y con el equipo para sacar la nota máxima posible. | **HABILIDADES**     * Trabajo en equipo * Proactivo * Actitud Positiva |

|  |
| --- |
| “Caer está permitido. ¡Levantarse es obligatorio!”  -  “Anónimo” |

|  |
| --- |
| Gianpiero Jesus Flores Tineo  Ingeniería de Sistemas e Información |

|  |  |
| --- | --- |
| **DESCRIPCION PERSONAL**  Soy Gianpiero Flores y tengo 20 años. Elegí la carrera de ingeniería de Sistemas porque me interesó el diseño, programación y aplicar tecnologías informáticas. Actualmente me encuentro cursando el 4to ciclo de mi carrera. |  |

|  |  |
| --- | --- |
| **COMPROMISO**  Me comprometo a esforzarme y apoyar a mi equipo para poder lograr una buena calificación, y avanzar en nuestro objetivo de formarnos como profesionales. | **HABILIDADES**     * Creatividad * Trabajo en equipo * Resolver problemas |

|  |
| --- |
| "El único modo de hacer un gran trabajo es amar lo que  haces"- Steve Jobs |

|  |
| --- |
| Daniel Ulises Barrionuevo Gutiérrez  Ciencias de la computación |

|  |  |
| --- | --- |
| **DESCRIPCION PERSONAL**  Soy Daniel Barrionuevo, tengo 20 años y actualmente curso el 4to ciclo de la carrera de ciencias de la computación. Escogí esta carrera ya que me gusta programar y diseñar aplicaciones |  |

|  |  |
| --- | --- |
| **COMPROMISO**  Me comprometo a cumplir con las tareas asignadas y apoyar a mis compañeros en todo lo posible para así avanzar todos juntos y cumplir con las tareas asignadas. | **HABILIDADES**     * Responsabilidad * Trabajo en equipo * Pensamiento crítico |

|  |
| --- |
| "Por muy alta que sea una montaña, siempre hay un camino hacia la cima."  -  Anónimo |

|  |
| --- |
| Gianpool Greg Pacheco Salambay  Ingeniería de Sistemas e Información |

|  |  |
| --- | --- |
| **DESCRIPCION PERSONAL**  Me llamo Gianpool Greg Pacheco, estudiante de años de la UPC. Estoy cursando el quinto ciclo de la carrera. Escogí esta carrera porque me gusta la tecnología y quisiera poder emplearla para un bien mayor. |  |

|  |  |
| --- | --- |
| **COMPROMISO**  Me comprometo a apoyar a mis compañeros y cumplir con los trabajos encargados | **HABILIDADES**     * Conocimientos de Programación * Trabajo en equipo |

|  |
| --- |
| "El genio se hace con un 1% de talento y un 99% de trabajo"- Albert Einstein |

## Solution Profile

### Antecedentes y problemática

## What?

## Los alquileres de autos tradicionales presentan una serie de inconveniencias que repercuten a la comodidad del cliente puesto a que estos tienden a demorar mucho tiempo en cuestión de trámite y papeleo como cuenta Juan Pablo Robles en el diario Gestión (2022) que hace algunos años tuvo que alquilar un vehículo, pero perdió tiempo, dinero y paciencia en firmas, seguro, etc. Es así como el tiempo es el principal factor contraproducente en los alquileres de autos tradicionales, aquel sustantivo crucial y significativo es importante salvaguardarlo en ocasiones no banales, por lo que es importante disminuir el tiempo de espera para obtención del vehículo.

## Where?

## En las zonas urbanas en donde se concentra mayormente o casi en su totalidad esta problemática ya sea en concesionarios de autos para alquiler o en el ciberespacio en la busca de algún automóvil

## When?

## Esta problemática ocurre al momento de querer alquilar un vehículo

## Who?

## Los principales involucrado en esta problemática, básicamente, son las personas que rentan autos y los que desean alquilar algunos

## Why?

## Esto ocurre debido al bajo nivel de logística brindada por las empresas de rentas de autos, tanto en el manejo de la información de los autos como el papeleo.

## How?

## Esto impacta de manera negativa a estos negocios presentándose de esa manera un déficit económico en este mercado consumista para las empresas de rentas y una pérdida de tiempo y dinero para su clientela. Es por ello que crearemos este aplicativo llamado “Fast Auto” que minimizará enormemente el tiempo de obtención de un auto y mejorará la logística de los empresarios de rentas.

## How much?

## Para esto emplearemos el uso de la tecnología y el manejo de una base de datos que permitirán el almacenamiento de información y la data importante, para un buen desenvolvimiento del sistema ante la problemática. Así mismo, nos otorgará información importante para tomar decisiones cruciales en un futuro.

**1.2.2. Lean UX Process**

**1.2.2.1. Lean UX Problem Statements**

Actualmente, existen diversas empresas que brindan el servicio de alquiler de autos. Estas empresas ofrecen a sus clientes una variedad de vehículos con diferentes precios y capacidad de pasajeros.

Hemos observado que algunas de estas empresas tienen un precio elevado de alquiler por día o tienen el precio cotizado en dólares.

¿Cómo podemos facilitar al cliente un precio asequible, además de brindar el precio en moneda local?

**1.2.2.2. Lean UX Assumptions**

**Business Assumptions**

* Creemos que nuestros clientes necesitan de unos precios variados, que se manejen con la moneda local. Además, brindarles ofertas y descuentos por un mayor tiempo de alquiler del vehículo.
* Estas necesidades se pueden resolver con una app que brinde diferentes tipos de vehículos.
* Nuestros clientes iniciales son turistas y aquellas personas que tengan licencia para conducir, pero no tengan un vehículo.
* El valor #1 que nuestros clientes quieren del servicio es tener un buen auto a un precio razonable.
* Nuestros clientes pueden obtener estos beneficios adicionales, ofertas y descuentos por el tiempo de alquiler.
* Vamos a adquirir la mayoría de nuestros clientes a través de publicidad en redes sociales y anuncios en sitios web de viajes.
* Produciremos dinero a través del alquiler del vehículo que ofreceremos a los clientes.
* Nuestra competencia principal en el mercado serán empresas como Budget, Hertz y Kayak que ofrecen un servicio similar.
* Los venceremos debido a los diferentes tipos de autos, tales como todoterreno, familiar, de carga, etc.
* El mayor riesgo del servicio es que un cliente sufra un accidente don alguno de nuestros vehículos.
* Resolveremos estos problemas con constantes mantenimiento a los vehículos.

**Client Assumptions**

* ¿Quiénes son los usuarios?

Los usuarios son los turistas y personas que no poseen un auto, pero tengan licencia para manejar.

* ¿Dónde encaja nuestro producto en sus trabajos o vidas?

Para nuestros usuarios, este servicio ayudara a poder visitar a diferentes lugares en sus días de descanso.

* ¿Qué problemas resuelve nuestro producto?

El problema que resuelve es en el ahorro de taxis que puedes generar, además de que poder elegir la capacidad de pasajeros que permitirá una mayor comodidad en el vehículo.

* ¿Cuándo y cómo es usado nuestro producto?

El servicio es usado cuando una persona que tenga un auto quiera alquilar uno para poder desplazarse y visitar más lugares.

* ¿Qué características son importantes?

Es importante que los clientes reciban un auto en buenas condiciones, que haga sentir al cliente seguro.

* ¿Cómo debería verse y comportarse nuestro producto?

Debe presentar una interfaz de fácil comprensión, el cual permita al cliente hacer filtros para elegir el tipo de auto, precio, capacidad según su preferencia.

**1.2.2.3. Lean UX Hypothesis Statements**

* **HYPOTHESIS 1**

Creemos que implementar un servicio de entrega del auto a un punto que seleccione el cliente, además de poder recoger en oficina. De esta manera facilitaremos al cliente poder entregar el auto en la puerta de su casa.

Sabremos que hemos tenido éxito.

Cuando veamos que hay un incremento de clientes que usan este servicio de entrega.

* **HYPOTHESIS 2**

Creemos que ofrecer ofertas y descuentos por el tiempo de alquiler que utilizara el cliente.

Sabremos que hemos tenido éxito.

Cuando veamos que hay un incremento de clientes que alquila autos por una semana o más tiempo.

* **HYPOTHESIS 3**

Creemos que ofrecer un filtro que permita al usuario elegir diferentes tipos de vehículos con precios según la demanda del cliente.

Sabremos que hemos tenido éxito.

Cuando veamos que los clientes obtienen el auto deseado a un buen precio y no hay quejas de falla de búsqueda.

**1.2.2.4. Lean UX Canvas**



CAPITULO 2: RESEARCH

**2.1.** **Análisis competitivo**

Las apps con dichos requisitos se encuentran en la siguiente tabla comparadas con nuestra App

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Features** | **Auto Fast** | **budgetperu** | | **Hertzperu** | | **kayak** | |
| **Es una app móvil de alquiler de autos** | ✓ | ✓ | |  | | ✓ | |
| **Vista de calendario accesible, calendario de pagos, etc** | ✓ | ✓ | |  | |  | |
| **Recojo de auto en oficina** | ✓ | ✓ | | ✓ | | ✓ | |
| **Entrega de auto** | ✓ |  | |  | |  | |
| **Reviews de los anteriores usuarios** | ✓ | ✓ | | ✓ | | ✓ | |
| **Ajuste de precio según demanda** | ✓ |  | |  | |  | |
| **Cuenta con guía de funciones del carro** | ✓ | ✓ | |  | |  | |
| **Los autos cuentan con GPS** | ✓ | | ✓ | |  | | ✓ |
| **Presencia internacional** | ✓ | | ✓ | |  | | ✓ |
| **Soporte en español** | ✓ | | ✓ | | ✓ | | ✓ |

Nuestra aplicación de alquiler de autos ofrece dos ventajas clave: ajuste de precio según la demanda y entrega de auto. El ajuste de precio según la demanda permite ofrecer tarifas flexibles para optimizar el uso de nuestros vehículos y ofrecer precios competitivos. La entrega de auto permite a los clientes recibir y devolver sus vehículos directamente en la ubicación de su elección, lo que hace que el proceso sea más fácil y conveniente. Estas ventajas nos permiten ofrecer un servicio más flexible y destacar en el mercado de alquiler de autos.

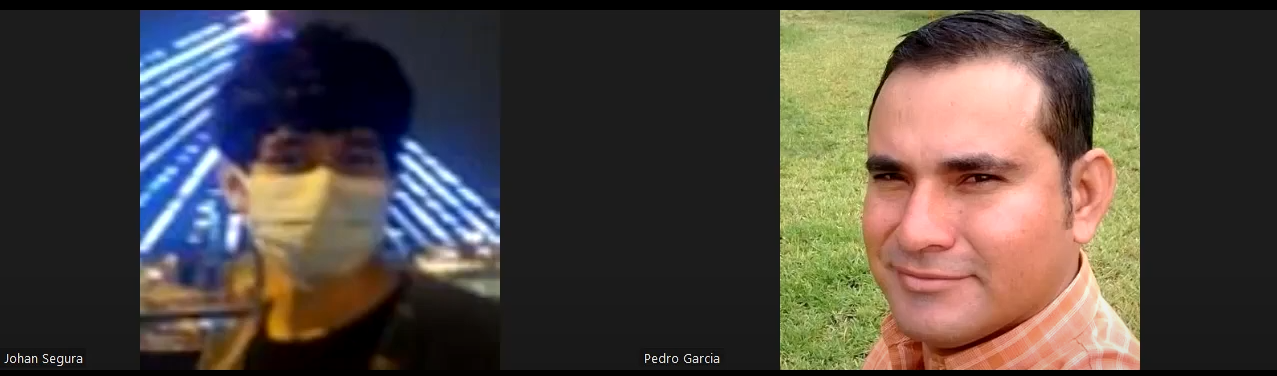
**2.2.** **Análisis clientes**

* El 100% de los entrevistados prefiere recibir el auto alquilado a domicilio
* El 66% de los entrevistados ya han probado aplicaciones similares y las ven muy útiles al momento de realizar viajes en los que no puedan llevar su propio vehículo
* El 66% quisiera que la aplicación ponga los precios y formularios precisos para disminuir el tiempo del trámite del alquiler
* El 33% quisiera poner en alquiler su propio vehículo por medio de la aplicación para otros clientes.

**Preguntas**

* ¿Cuál fue su experiencia alquilando autos?
* ¿Considera sencillo el proceso de alquiler de un auto?
* ¿En qué casos consideraría utilizar la aplicación?
* ¿Qué sugerencia le daría a una aplicación de alquiler de autos?

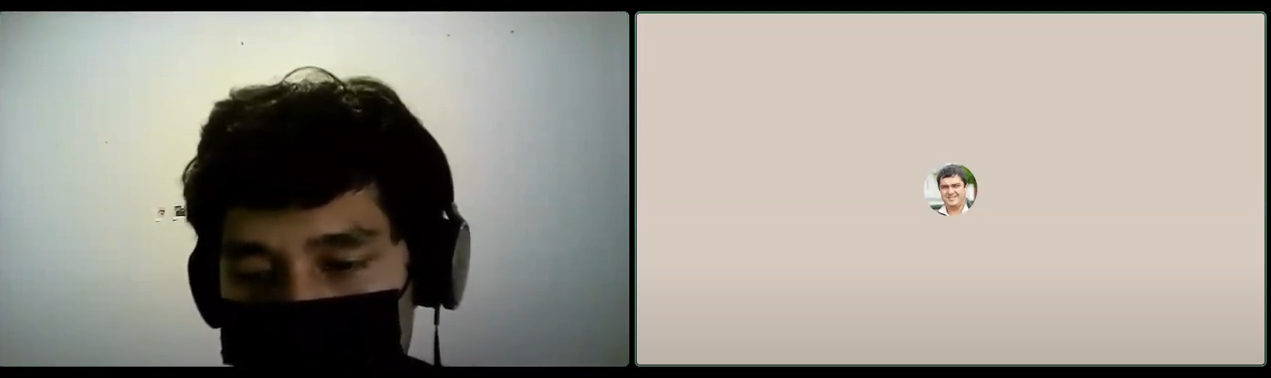
**Videos**

Video 1: <https://www.youtube.com/watch?v=5awYAmayXXU>Nombre: Pedro García

Edad: 47

Descripción de la entrevista:

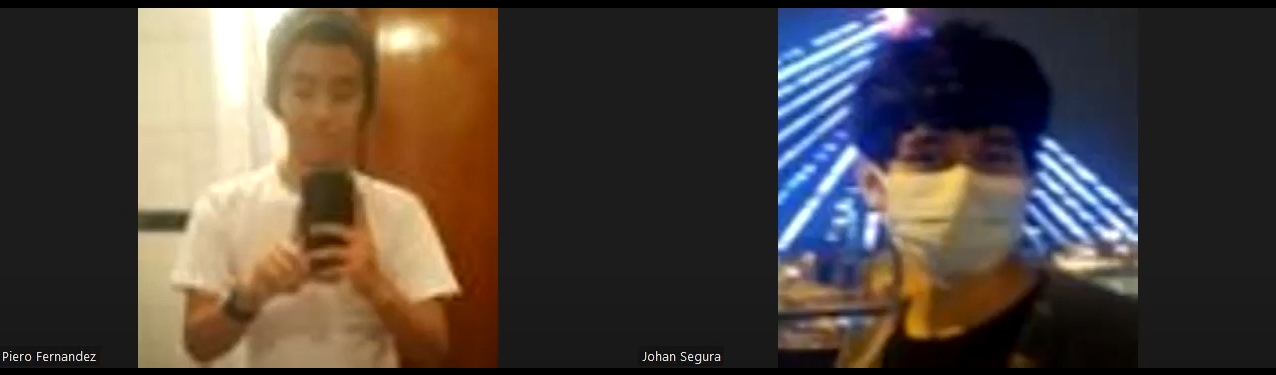
Pedro García nos comentó que ya había usado este tipo de aplicaciones debido a que viaja mucho por trabajo, sin embargo, tuvo algunos inconvenientes con el proceso de alquiler. Asimismo, le gustaría recibir el auto a domicilio ya que no siempre está en casa y podría delegarle la tarea de recibirlo a algún familiar. Además, nos sugirió pedir la información puntual y necesaria para hacer el proceso más rápido.

Video 2: <https://www.youtube.com/watch?v=j0CKWebpyBw>Nombre: Juan Mendoza

Edad: 54

Descripción de la entrevista:

Juan Mendoza nos comentó es su primera vez probando una aplicación de alquiler de autos y que lo usaría por motivos de trabajo. Asimismo, le gustaría recibir el auto a domicilio ya que así ahorraría mucho tiempo al no tener que ir hasta el local. Además, nos sugirió que las personas externas a la empresa puedan usar la aplicación para poder alquilar sus autos por medio de esta.

Video 3: <https://www.youtube.com/watch?v=VcUEocLxyUQ>Nombre: Piero Fernández

Edad: 31

Descripción de la entrevista:

Piero Fernández nos comentó es su primera vez probando una aplicación de alquiler de autos y que lo usaría en caso de irse de viaje ya que no podría llevar su vehículo y la aplicación le facilitaría mucho movilizarse. Asimismo, le fascina recibir el auto a domicilio ya que se le hace más cómodo. Además, nos sugirió que los precios se puedan visualizar desde un principio ya que esto agilizaría mucho el proceso.

**User persona**

|  |  |  |
| --- | --- | --- |
| **Nombre**  Jorge Herrera | | **Segmento**  Cliente |
|  | **Bio**  Jorge es un ingeniero de sistemas que le gusta pasar tiempo en familia. Normalmente viaja debido a su trabajo y de vez en cuando le gusta llevarse a su familia con él. | |
| **Metas**  **•** Pasar más tiempo con su familia  **•** Sentirse seguro al realizar el alquiler | |
| **Sexo**  Hombre    **Edad**  38    **Nacimiento**  Lima, Peru    **Estado Civil**  Casado    **Profesión**  Ing. Civil | **Motivaciones**  **•** Ganar tiempo al momento de realizar el alquiler  **•** Encontrar una aplicación que agilice el tramite  **•** Ahorrar costos | |
| **Frustraciones**  **•** Procesos lentos  **•** Miedo a estafas | |
| **Perfil**  **-** Apertura a la experiencia   |  |  | | --- | --- | |  |  |     **-** Responsabilidad   |  |  | | --- | --- | |  |  |     **-** Extraversión   |  |  | | --- | --- | |  |  | | **-** Amabilidad   |  |  | | --- | --- | |  |  |     **-** Estabilidad emocional   |  |  | | --- | --- | |  |  | |
|  |

|  |  |  |
| --- | --- | --- |
| **Nombre**  Pedro Quispe | | **Segmento**  Empresa |
|  | **Bio**  Pedro es un gerente de una empresa de alquiler de autos que le gusta socializar con los demás y aprender cosas nuevas. | |
| **Metas**  **•** Aprender cosas nuevas que puedan ser de ayuda en su vida diaria  **•** Tener buenos trabajadores  • Agilizar las ventas  • Tener información precisa para tomar mejores decisiones | |
| **Sexo**  Hombre    **Edad**  53    **Nacimiento**  Lima, Peru    **Estado Civil**  Soltero    **Profesión**  Gerente | **Motivaciones**  **•** Hacer que su empresa lidere el mercado  • Ser el mejor en su puesto y ser reconocido | |
| **Frustraciones**  **•** Ser nuevo en el mercado | |
| **Perfil**  **-** Apertura a la experiencia   |  |  | | --- | --- | |  |  |     **-** Responsabilidad   |  |  | | --- | --- | |  |  |     **-** Extraversión   |  |  | | --- | --- | |  |  | | **-** Amabilidad   |  |  | | --- | --- | |  |  |     **-** Estabilidad emocional   |  |  | | --- | --- | |  |  | |
|  |

CAPITULO 3: DATABASE DESIGN

**3.1.** **Entities**

**Entidad Empleados**

En esta entidad se guardarán los datos de identificación de la persona, tales como nombres, apellidos y DNI. Además, se registrarán con un id que registrara su local de trabajo.

**Entidad Vendedor**

Esta entidad guardara el registro de ventas que genero un vendedor.

**Entidad Cliente**

En esta entidad se van a guardar los datos personales del cliente y una id de dirección para posibles entregas a domicilio.

**Entidad Dirección**

En esta entidad se van a guardar la dirección, ciudad, distrito y departamento del cliente para llevar el auto a domicilio.

**Entidad Local**

En esta entidad se guardarán los datos de dirección del local.

**Entidad Auto**

En esta entidad se registrarán la placa del auto, color, modelo y marca del auto disponibles al cliente.

**Entidad Contrato**

En esta entidad se guardarán la id del comprobante de pago, datos del vendedor, datos del cliente, la fecha de inicio y la fecha de fin y el seguro que se brindara al cliente.

**Entidad Seguro**

En esta entidad se registrará un id que identificará el tipo de seguro, la empresa de seguro, el auto y el monto a pagar.

**Entidad Medio de Pago**

En esta entidad se van a guardar los métodos de pago por transferencia o efectivo.

**3.2. Atributes**

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

VendedorId: Es una llave foranea de tipo integer que contiene el identificador de un vendedor

ClienteId: Es una llave foránea que conteiene los datos del cliente.

ComprobanteId: Es una llave foránea que conteiene los datos del comprobante

Fecha inicio y Fecha Entrega: Contiene la fecha de inicio y fin del contrato.

SeguroId: es un identificador que contiene el nombre de la empresa que asegura al carro.

Direccion: Es un atributo de tipo varchar el cual contiene la dirección.

AutoId: es un identificador que contiene a la información del auto.

Pantalla de computadora con letras

Descripción generada automáticamente con confianza mediad

Nombres y apellidos: Es un atributo de tipo varchar y contiene el nombre y apellidos del empleado.

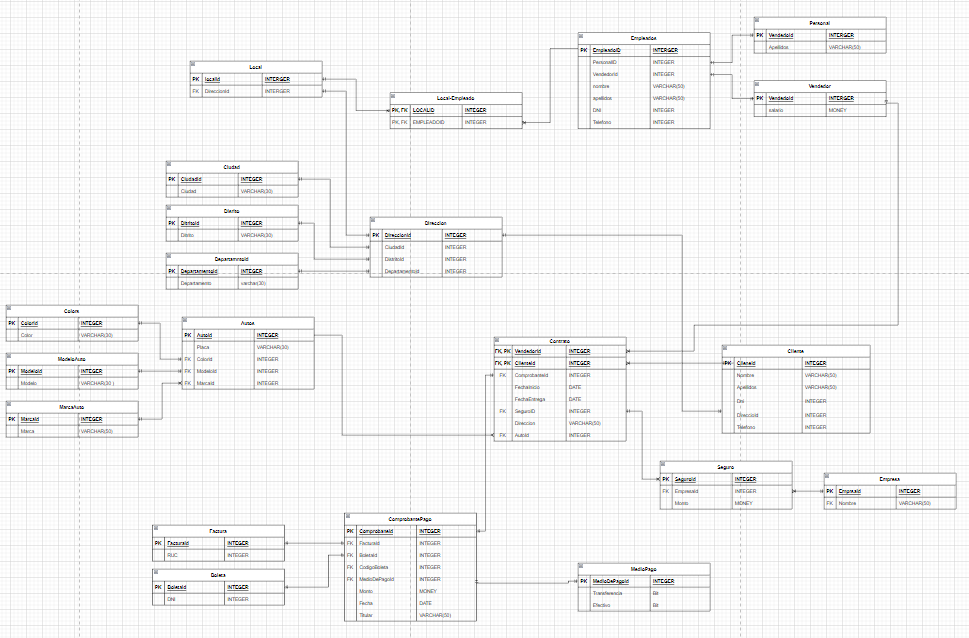
Dni: Es un atributo de tipo integer que almacena el identificador del vendedor, sin embargo, en este caso el vendedor tiene un propio identificador

LocalId: Es una PK que esta relacionado con una tabla que genera una relación entre empleado y local.

VendedorId: Es una llave foranea de tipo integer que contiene el identificador de un vendedor.

PersonalId: Es una llave foránea de tipo integer que contiene el identificador de un personal.

:

**3.3. Database Diagram**   
   
  
<https://drive.google.com/file/d/1jmDhk1Ua3-589g1DbybZ2qZFWEtIgL8G/view?usp=sharing>

**4. Selección de base de datos Relacional**

Elegí SQLserver como el motor de base de datos para mi proyecto debido a su amplia adopción en la industria, estabilidad y confiabilidad, escalabilidad y facilidad de uso. La comunidad de usuarios y recursos disponibles en línea también me proporciona una mayor tranquilidad en caso de problemas o necesidad de asistencia. En general, SQL server es una opción probada y confiable para aplicaciones de negocio críticas y su escalabilidad lo hace ideal para manejar grandes volúmenes de datos y transacciones simultáneas.

**4.1. Script Base de datos en SQLserver**

LINK: <https://drive.google.com/file/d/1-9Q_ibt4FcQfsR00ljVgOY3xqpGwf2SR/view>

USE [AutoFast]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Autos] Script Date: 21/02/2023 13:58:52 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Autos](

[AutoID] [int] NOT NULL,

[Placa] [varchar](10) NOT NULL,

[ColorID] [int] NOT NULL,

[ModeloID] [int] NOT NULL,

[MarcaID] [int] NOT NULL,

PRIMARY KEY CLUSTERED

(

[AutoID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[boletas] Script Date: 21/02/2023 13:58:52 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[boletas](

[BoletaID] [int] NOT NULL,

[Dni] [varchar](10) NOT NULL,

PRIMARY KEY CLUSTERED

(

[BoletaID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

**5. Selección de base de datos no relacional**

Opté por MongoDB como el motor de base de datos para mi proyecto debido a su flexibilidad y escalabilidad, especialmente en aplicaciones que manejan grandes cantidades de datos no estructurados. La capacidad de MongoDB de manejar grandes volúmenes de datos, junto con su capacidad de escalar horizontalmente, lo hace ideal para aplicaciones con requisitos de alto rendimiento. Además, la capacidad de MongoDB para trabajar con datos en formatos flexibles y no estructurados, como JSON, simplifica la integración de diferentes tipos de datos en mi aplicación. En general, MongoDB es una excelente opción para aplicaciones web modernas y escalables que manejan grandes cantidades de datos no estructurados.

**5.1. Database Diagram**

Diagrama

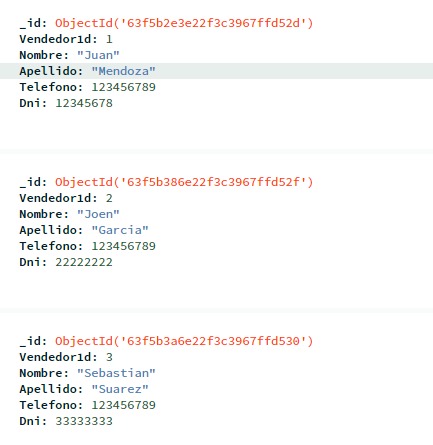
Descripción generada automáticamente

**5.2. Database Collections**

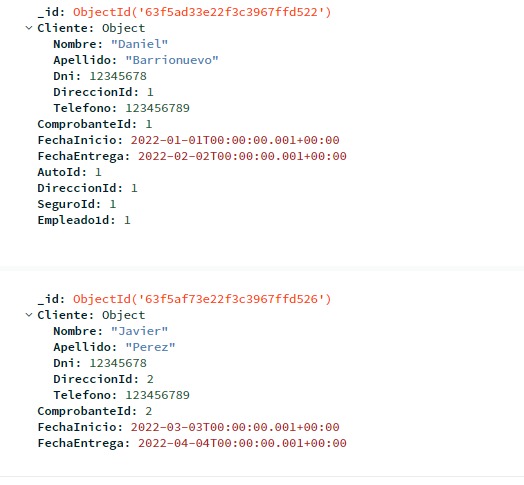
**Autos**



**Empleados**

****

**Contrato**

****

Las tablas de Autos, contrato, cliente y vendedores son fundamentales para el funcionamiento de la aplicación de alquiler de autos. Estas tablas contienen información clave sobre los vehículos, los clientes, los vendedores y los contratos, y son esenciales para llevar un registro preciso y actualizado de las operaciones de la empresa.

Al elegir MongoDB como base de datos NoSQL, podemos aprovechar sus ventajas, como su capacidad de manejar grandes volúmenes de datos no estructurados, su escalabilidad y su flexibilidad en el manejo de datos complejos. Además, MongoDB ofrece una gran facilidad de uso y una amplia variedad de herramientas y aplicaciones, lo que facilita su implementación y mantenimiento.

**5.2. Database Schema Validation**

**Autos**

$jsonSchema: {

bsonType: "object",

required: [ "AutoId","Placa", "Color", "Modelo", "Marca" ],

properties: {

AutoId: {

bsonType: "int",

description: "Can only be a number"

},

Placa: {

bsonType: "string",

description: "must be a string"

},

Color: {

bsonType: "string",

description: "must be a string"

},

Modelo: {

bsonType: "string",

description: "Can only be a number"

},

Marca: {

bsonType: "string",

description: "must be a double"

}

}

}

}

**Empleado**

, {

$jsonSchema: {

bsonType: "object",

required: [ "VendedorId","Nombre", "Apellido", "Telefono", "Dni" ],

properties: {

VendedorId: {

bsonType: "int",

description: "Can only be a number"

},

Nombre: {

bsonType: "string",

description: "must be a string"

},

Apellido: {

bsonType: "string",

description: "must be a string"

},

Telefono: {

bsonType: "int",

description: "Can only be a number"

},

Dni: {

bsonType: "int",

minimum: 0,

description: "must be a double"

}

}

}

}

**Contrato**

, {

$jsonSchema: {

bsonType: "object",

required: [ "ComprobanteId","FechaInicio", "FechaEntrega", "AutoId", "DireccionId", "Seguro", "EmpleadoId", "Cliente" ],

properties: {

ComprobanteId: {

bsonType: "int",

description: "Can only be a number"

},

FechaInicio: {

bsonType: "date",

description: "must be a date"

},

FechaEntrega: {

bsonType: "date",

description: "must be a date"

},

AutoId: {

bsonType: "int",

description: "Can only be a number"

},

DireccionId: {

bsonType: "int",

description: "must be a number"

},

Seguro: {

bsonType: "int",

description: "must be a number"

},

EmpleadoId: {

bsonType: "int",

description: "must be a number"

}, Cliente: {

bsonType: "object",

description: "Cliente is required",

properties: {

Nombre: {bsonType:"string"},

Apellido: {bsonType:"string"},

Dni: {bsonType:"int"},

DireccionId: {bsonType:"int"},

Telefono: {bsonType:"int"}

}

}

}

}

}

**5.3. Database Data Model Patterns**

Empleados: Este esquema está relacionado con Contrato mediante el subset pattern, esto debido a la cantidad de datos y poco uso solo generaría una carga adicional al momento de la lectura.

Contrato: En este esquema se uso el modelado de datos embedded document pattern ya que la relación de cliente a contrato es de uno a uno y esto ayuda a que se reduzca la cantidad de operaciones necesarias para poder visualizar los datos del cliente sin necesidad de acceder a otro esquema.

Autos: Del mismo modo que el esquema de Empleados, se usó el subset pattern ya que esto hace más ágil la lectura de los esquemas y evita que se sobrecarguen los campos del esquema Contrato haciéndola más sencilla de leer.

**5.4 Scripts MongoDb**

Estos son los scripts que se utilización para crear las colecciones en MongoDB

db.createCollection(Autos, {

$jsonSchema: {

bsonType: "object",

required: [ "AutoId","Placa", "Color", "Modelo", "Marca" ],

properties: {

AutoId: {

bsonType: "int",

description: "Can only be a number"

},

Placa: {

bsonType: "string",

description: "must be a string"

},

Color: {

bsonType: "string",

description: "must be a string"

},

Modelo: {

bsonType: "string",

description: "Can only be a number"

},

Marca: {

bsonType: "string",

description: "must be a double"

}

}

}

})

db.createCollection(Empleado, {

$jsonSchema: {

bsonType: "object",

required: [ "VendedorId","Nombre", "Apellido", "Telefono", "Dni" ],

properties: {

VendedorId: {

bsonType: "int",

description: "Can only be a number"

},

Nombre: {

bsonType: "string",

description: "must be a string"

},

Apellido: {

bsonType: "string",

description: "must be a string"

},

Telefono: {

bsonType: "int",

description: "Can only be a number"

},

Dni: {

bsonType: "int",

minimum: 0,

description: "must be a double"

}

}

}

})

db.createCollection(Contrato, {

$jsonSchema: {

bsonType: "object",

required: [ "ComprobanteId","FechaInicio", "FechaEntrega", "AutoId", "DireccionId", "Seguro", "EmpleadoId", "Cliente" ],

properties: {

ComprobanteId: {

bsonType: "int",

description: "Can only be a number"

},

FechaInicio: {

bsonType: "date",

description: "must be a date"

},

FechaEntrega: {

bsonType: "date",

description: "must be a date"

},

AutoId: {

bsonType: "int",

description: "Can only be a number"

},

DireccionId: {

bsonType: "int",

description: "must be a number"

},

Seguro: {

bsonType: "int",

description: "must be a number"

},

EmpleadoId: {

bsonType: "int",

description: "must be a number"

}, Cliente: {

bsonType: "object",

description: "Cliente is required",

properties: {

Nombre: {bsonType:"string"},

Apellido: {bsonType:"string"},

Dni: {bsonType:"int"},

DireccionId: {bsonType:"int"},

Telefono: {bsonType:"int"}

}

}

}

}

} )

**6 Relational Database Queries**

**6.1. SQLserver Database queries**

**Queries Fáciles – 3**

* La empresa desea ver la informacion de todos los Vendedores en un Local y el monto que vendieron desde que trabajan en la empresa.

SELECT Empleados.nombre, Empleados.apellidos, empleados.dni, Empleados.telefono, vendedor.MontoVentas from Local

JOIN Local-Empleado ON Local.localId = Local-Empleado.localId

JOIN Empleados ON Local-Empleado.empleadoId = Empleados.EmpleadoID

JOIN Vendedor ON Empleados.VendedorID = Vendedor.VendedorID WHERE VendedorId IS NOT NULL

 ORDER BY Vendedor.MontoVentas DESC;

* Se desea buscar un auto en especifico

Select placa, Modelo

From modelos

join Autos on Autos.ModeloID = modelos.ModeloId

Where Placa = 'ASD123'

* Brinda informacion del personal que trabaja en los locales

select Empleados.nombre, Empleados.apellidos, empleados.dni, Empleados.telefono from Local

JOIN Local\_Empleado ON Local.localId = Local\_Empleado.localId

JOIN Empleados ON Local\_Empleado.empleadoId = Empleados.EmpleadoID

JOIN Personal ON Empleados.PersonaID = Personal.PersonalID

**Queries Intemedios - 6**

1. Una funcion que te permite buscar autos por marcas

 CREATE FUNCTION dbo.AutosPorMarca(

@marcaAuto varchar(30)

)RETURNS int

BEGIN

DECLARE @cantidadAutosPorMarca int

SELECT @cantidadAutosPorMarca = COUNT(Autos.MarcaId) FROM Autos

JOIN MarcaAuto ON Autos.MarcaId = MarcaAuto.MarcaId

WHERE MarcaAuto.Marca = @marcaAuto

GROUP BY  autos.marcaId

return @cantidadAutosPorMarca

END

go

SELECT dbo.AutosPorMarca('Audi')

1. Actualizar la información de un cliente usando su id

CREATE OR ALTER PROCEDURE actualizarCliente(

@nombre varchar(50)

@apellidos  varchar(50)

@telefono int

@ClienteId int

)

AS

BEGIN

UPDATE Cliente SET Nombre = @nombre, Apellidos = @apellidos, Telefono = @telefono

WHERE  ClienteId = @ClienteId;

END

EXEC actualizarCliente ‘Pedro’, ‘Picapiedra’, 123123123

1. Permite mostrar el monto que pago el cliente en el contrato

Create function dbo.MontoDelCliente(@ClienteId int)

returns Money

as

begin

declare @Monto Money

select @Monto = ComprobantePago.Monto

From Cliente join Contrato on Cliente.clienteID = Contrato.clienteID

join ComprobantePago on Contrato.ComprobanteId = ComprobantePago.ComprobanteId

where ClienteId= @ClienteId

return @Monto

end

go

SELECT dbo.MontoDelCliente(1) as Monto,Nombre from Cliente where clienteID=1

1. La empresa desea hallar la duración de un contrato

Create function dbo.DuracionDeAlquiler(@VendedorId int) returns date

as

begin

declare @Duracion date

select @Duracion = (FechaEntrega - FechaInicio) from Contrato where VendedorId = @VendedorId

return @Duracion

end

1. Nos da la cantidad de autos existente de la empresa

Create function dbo.CantidadAutos()

returns int

as

begin

declare @cont int

select @cont=COUNT(AutoID)from Autos

return @cont

end

select dbo.CantidadAutos()

1. Nos permite insertar un auto de acuerdo a la cantidad permitida

CREATE OR ALTER PROCEDURE InsertarAuto(

@autodi int,

@placa varchar(10),

@colorid int,

@modeloid int,

@marcaid int,

@cantidad int

)

AS

BEGIN

if  @cantidad<5

begin

Insert into Autos

(AutoID,

Placa,

ColorID,

ModeloID,

MarcaID)

Values(

@autodi,

@placa,

@colorid,

@modeloid,

@marcaid)

end

END

Exec InsertarAuto 4,'b3v694',5,1,3,3

**6.2. Mongodb Database queries**

Conseguir todos contratos con los autos de ID = 3

Db.getCollection(‘Contrato’).find({ ‘AutiId’: 3 })

Conseguir todos los empleados con nombre igual a Pedro

Db.getCollection(‘Empleados’).find({ ‘Nombre‘ : “Pedro” })

Conseguir todos los registros de lso autos con marca toyota

Db.getCollection(‘Autos’).find({ ‘Marca’: “Toyota” })

Conseguir el registro del auto con la placa XXX-XXX

Db.getCollection(‘Autos’).find({ ‘Placa’: “XXX-XXX” })

Conseguir los contratos del cliente con el DNI 12312312

Db.getCollection(‘Contrato’).find({ ‘Cliente’: { ‘DNI’ : 12312312 }” })

Actualizar el nombre de un usuario

Db.getCollection(‘Contrato’).update({ ‘nombre’: {$eq : ‘Carlos’ }}, { $set: {‘nombre’: “Alberto”}} )

5.2