

## 1. Planificación Scrum

### 1.1. Definición clara de los roles y responsabilidades del equipo.

- Daniel Felipe Mendez Güetocue: Asignación de tareas para los miembros del grupo, elaboración de documentación de planeación del aplicativo, elaboración de frontend y Backend.
- Jhonathan Arley Silva Piñacue: Elaboración de documentación de planeación del aplicativo y backend.
- Juan Camilo Cubides Solano: Elaboración de documentación y creación de base de datos.
- Sebastian Camilo Rojas Polania:

La asignación a detalle de las tareas que debe realizar cada miembro según su rol lo manejamos por medio de la plataforma trello.

Enlace de trello:  
<https://trello.com/invite/b/PBLbfe4v/ATTIc79c8be35b38b30205246dd8b7de3a63D0003029/grupo-3-sistemas-distribuidos>

## 2. Actas de reuniones

El control de actas se maneja

La evidencia de dicha labor se esta llevando el control por medio de un control de actas la cual se registran semanalmente la labor o la participación que tiene cada miembro del grupo para dicho control del seguimiento de lo que se realiza semanalmente.

Las actas del proyecto las encontramos en el siguiente enlace:

<https://github.com/jucacuso96/Grupo-3/tree/main/Actas>

- 2.1. Documentación clara de los puntos de acción y responsables asignados.
- 2.2. Inclusión de los acuerdos y compromisos alcanzados durante la reunión.
- 2.3. Distribución oportuna de las actas a los participantes.

## 3. Código

### 3.1. Uso de estándares de codificación y convenciones de nomenclatura.

### 3.2. Nomenclatura acordada para el proyecto:

En este proyecto, es crucial establecer una nomenclatura coherente y comprensible para garantizar la claridad y la consistencia en el desarrollo. La nomenclatura es una parte fundamental de la organización del código, los archivos y otros elementos del proyecto, lo que facilita la colaboración entre los miembros del equipo y el mantenimiento a largo plazo.

## Convenciones de Nomenclatura:

Para mantener una estructura uniforme y fácilmente comprensible, hemos decidido adoptar las siguientes convenciones de nomenclatura:

**snake\_case:** Utilizaremos el estilo snake\_case para nombrar archivos, carpetas, clases y variables. Esto implica escribir todas las letras en minúsculas y separar las palabras con guiones bajos.

2. Ejemplos de Nomenclatura:

### Clases y Archivos:

- **Ejemplo de clase:** user\_profile

```
class producto_model extends Model
> { ...
}
```

- **Ejemplo de Funciones:** obtener productos

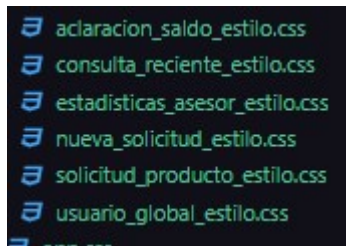
```
public static function obtener_productos()
{
    $productos = DB::select("SELECT * FROM producto");
    return $productos;
}
```

- **Ejemplo de archivo:** user\_profile\_controller.php

```
acleracion_saldo.blade.php
acuerdo_pago.blade.php
consulta_reciente.blade.php
detalle_caso.blade.php
escalar_caso.blade.php
estadistica_asesor.blade.php
nueva_solicitud.blade.php
solicitud_producto.blade.php
```

### Archivos de Estilo (CSS):

Ejemplo de archivo CSS: main\_styles.css

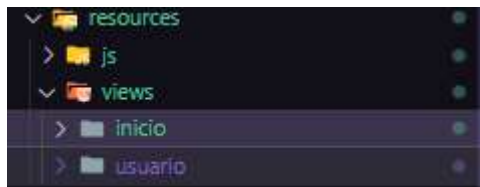


```

aclaracion_saldo_estilo.css
consulta_reciente_estilo.css
estadisticas_asesor_estilo.css
nueva_solicitud_estilo.css
solicitud_producto_estilo.css
usuario_global_estilo.css

```

Carpetas: la creación de carpeta será todo en minúscula y si existe una segunda palabra en adelante se utilizare el método de escritura snake\_case



```

resources
├── js
├── views
│   ├── inicio
│   └── usuario

```

### Archivos de JavaScript:

- Ejemplo de archivo JS: utility\_functions.js



```

js
├── estilo
│   ├── grafica_1.js
│   ├── grafica_2.js
│   └── menu.js

```

### Base de datos

- Ejemplo entidad: tipo\_usuario



Tabla
<input type="checkbox"/> caso
<input type="checkbox"/> cliente
<input type="checkbox"/> grupo_especialidad
<input type="checkbox"/> grupo_especialidad_pais
<input type="checkbox"/> manual_producto

- Ejemplo atributos: nombre\_producto

#	Nombre	Tipo
1	id 	int(11)
2	numero_caso	int(11)
3	asunto	varchar(100)
4	estado_caso	varchar(11)
5	id_producto 	int(11)
6	id_usuario 	int(11)
7	id_cliente 	int(11)
8	observacion	text

### Razones para la Elección de esta Nomenclatura:

Claridad: El estilo snake\_case es fácil de leer y entender, lo que facilita la comprensión del código para todos los miembros del equipo.

Consistencia: Mantener una convención de nomenclatura consistente en todo el proyecto ayuda a evitar confusiones y errores.

Compatibilidad con Laravel: Adaptaremos nuestras convenciones de nomenclatura para que sean compatibles con Laravel, lo que facilitará la integración y el mantenimiento del proyecto en el marco de trabajo.

### Responsabilidades del Equipo:

Cada miembro del equipo es responsable de adherirse a estas convenciones de nomenclatura y de asegurarse de que cualquier nueva adición al proyecto siga estas pautas. Además, se alienta a todos los miembros del equipo a proporcionar retroalimentación y sugerencias para mejorar nuestra nomenclatura si es necesario.

## 4. Diseño

- 4.1. Diseño de interfaz y experiencia de usuario.
- 4.2. Modularidad y reutilización del código.
- 4.3. Usabilidad y accesibilidad para diferentes tipos de usuarios.
- 4.4. Correcto funcionamiento de todas las características y elementos interactivos.