



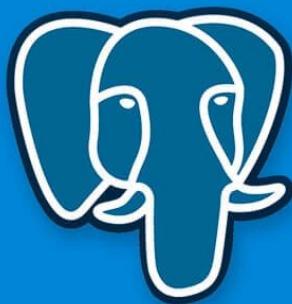
# Presentacion Final

**Espitia Pizza Juan Manuel** (jespitiap@unal.edu.co)  
**Prieto Mendoza Andrew Nicolay** (aprietome@unal.edu.co)  
**Rodriguez Ortiz Sebastian Steeven** (serodriguezor@unal.edu.co)  
**Gonzalez Tapia Yeswah** (yegonzalezt@unal.edu.co)  
**Cortes Salazar David Camilo** (dacortess@unal.edu.co)



# Arquitectura del proyecto

A lo largo del proyecto se utilizó la estructura por capas

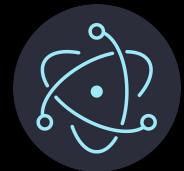


```
✓ repositories
  JS AlertRepository.js
  JS BalanceRepository.js
  JS CajaRepository.js
  JS ClosingRepository.js
  JS LogsRepository.js
  JS MovementRepository.js
  JS MovementUserRepository.j
  JS PaymentMethodRepositor.
  JS RoleRepository.js
  JS UserRepository.js
```



Capa de base de datos

Capa de persistencia



# Arquitectura del proyecto

A lo largo del proyecto se utilizó la estructura por capas

```
✓ ipc
  JS alert.handler.js
  JS auth.handler.js
  JS closing.handler.js
  JS export.handler.js
  JS movement.handler.js
  JS MovimientosHandler.js
  JS paymentMethod.handler.js
  JS role.handler.js
  JS user.handler.js
  JS main.js
  JS preload.cjs
```

```
✓ services
  JS AlertService.js
  JS AuthService.js
  JS BalanceService.js
  JS ClosingService.js
  JS ClosingServiceC.js
  JS ExportService.js
  JS LogsService.js
  JS MovementService.js
  JS PaymentMethodService.js
  JS RoleService.js
  JS UserService.js
```

```
✓ dto
  JS CreateClosingDTO.js
  JS CreateMovementDTO.js
  JS CreatePaymentMethodDT...
  JS CreateUserDTO.js
  JS LoginDTO.js
  JS UpdatePaymentMethodDT...
  JS UpdateUserDTO.js
  ✓ exceptions
    JS BusinessLogicException.js
    JS DuplicateException.js
    JS NotFoundException.js
    JS UnauthorizedException.js
    JS ValidationException.js
```

Capa de Negocios

# Arquitectura del proyecto

A lo largo del proyecto se utilizó la estructura por capas

```
✓ renderer/components
  ✓ auth
    ✎ AdministrarRoles.jsx
    ✎ AuthContext.jsx
    ✎ Login.jsx
    ✎ UserRegister.jsx
  ✓ auxiliary
    ✎ PermisosTabla.jsx
    ✎ UserTable.jsx
  ✓ layout
    ✎ Dashboard.jsx
    ✎ LeftPanel.jsx
    ✎ RightContent.jsx
  ✓ layout
    ✎ Dashboard.jsx
    ✎ LeftPanel.jsx
    ✎ RightContent.jsx
  ✓ ui
    ✎ button.jsx
    ✎ card.jsx
    ✎ input.jsx
    ✎ label.jsx
    ✎ select.jsx
    ✎ textarea.jsx
    ✎ utils.jsx
  # App.css
  # App.jsx
  # index.css
  # main.jsx
```



Capa de Presentación

# Desafíos Técnicos

- Adaptar el uso de un nuevo patrón de diseño (Data Transfer Object)
- Refactorizar funciones de acuerdo un nuevo estándar y a otras funciones añadidas
- Tener una concordancia en la base de datos y modificarla de acuerdo a necesidades que se pasaron por alto
- Usar Git de manera disciplinada, evitando problemas comunes relacionados con rebase y merge.
- Realizar siempre un pull antes de comenzar a trabajar, para garantizar que el repositorio local esté actualizado.
- Mejorar el uso de ramas, manteniendo una estructura clara y aplicando buenas prácticas para facilitar la colaboración.
- No enviar archivos sensibles (ej. .env, seeds, archivos generados)
- Coordinar mejor las añadiduras al programa
- No especificar bien el backend y el frontend

# Reflexiones finales



# Lecciones Aprendidas

- Organizar adecuadamente el trabajo: definir tareas, responsables y tiempos para evitar retrabajos o confusiones.
- Evitar comenzar proyectos con tecnologías desconocidas: priorizar lenguajes y herramientas que el equipo domine para garantizar eficiencia y calidad.
- Documentar las funcionalidades desarrolladas: registrar cambios, decisiones técnicas y uso de cada módulo para facilitar mantenimiento y escalabilidad.
- Trabajar en equipos para el desarrollo de software requirió más sincronización de lo esperado y puede llegar a ser complejo
- El diseño de software y la documentación es una base necesaria.

# **Opiniones Acerca Del Curso**

## **Limitaciones:**

- El horario en la noche
- La restricción a un solo lenguaje

## **Sugerencias:**

- Más puntos de referencias e información para ahondar de forma autónoma.