<!--Estudio Shonos-->

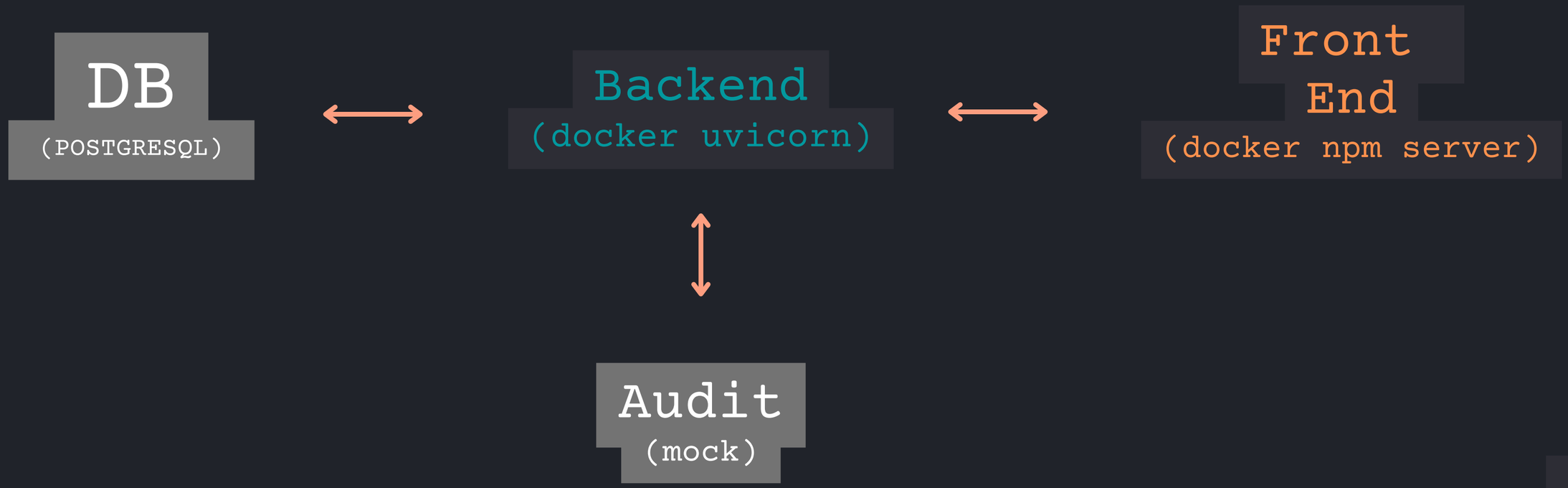
CreditSim {

<Por="Rodrigo García Padilla"/>

}

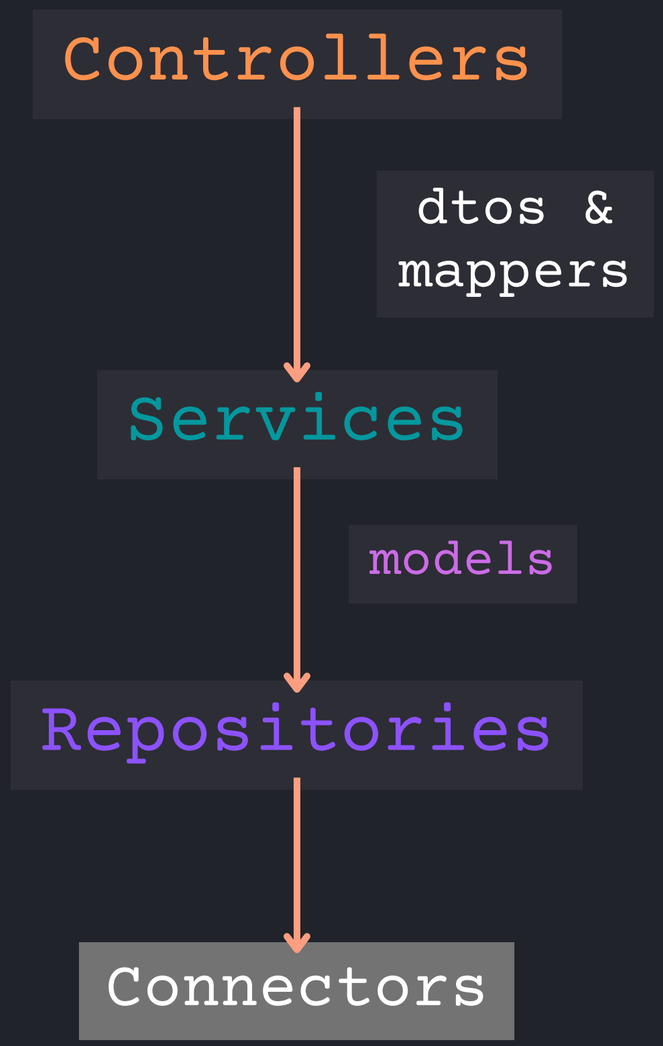
Contenidos

1 Arquitectura:



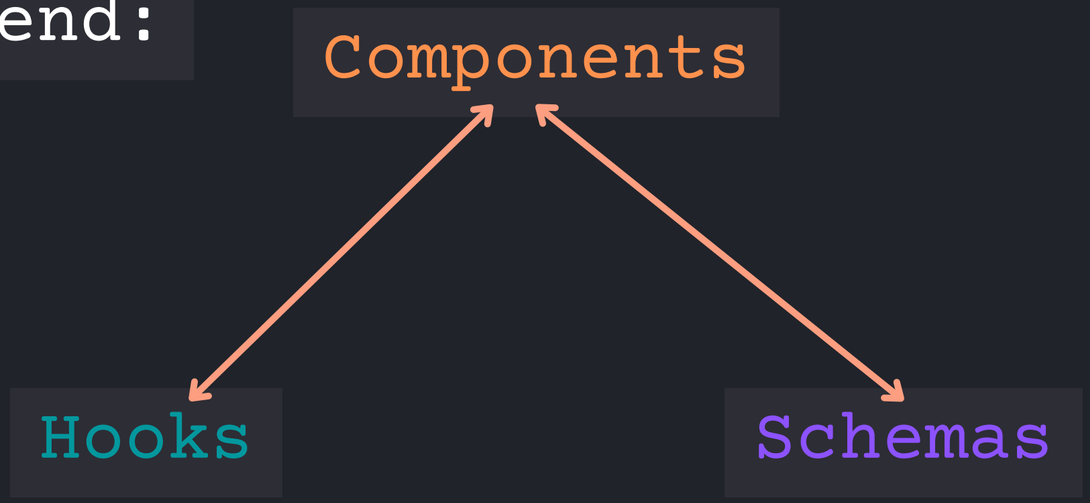
}

2 Diseño Backend:



}

2 Diseño Frontend:



}

3 Requerimientos Backend {

- Cálculo: validación objeto en request pydantic
  - GET /simulate

```
{amortization_periods:[
 {
 "amortization_table": "<simulation_json>",
 "id": 1,
 "annual_rate": 12.0,
 "months_term": 12,
 "amount": 12.0
 },
 {...}
 ]}
```

  - POST /simulate

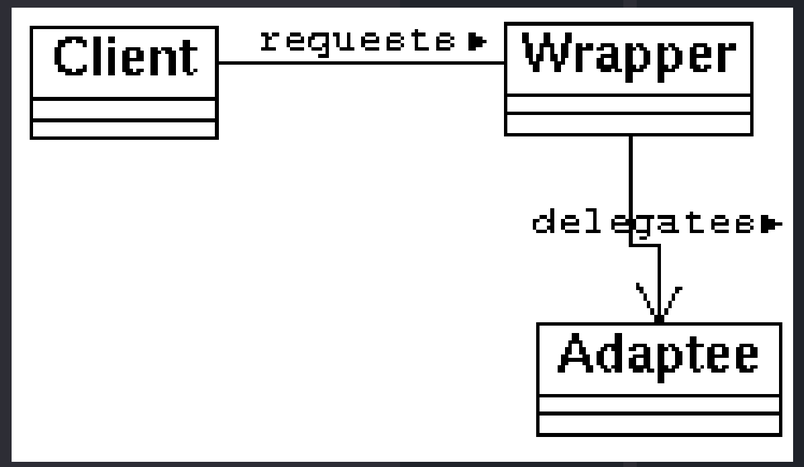
```
{amortization_periods: [
  {
  "actual_period": 0,
  "period_payment_amount": 2183.54,
  "interest_monthly_rate": 0.03,
  "period_interest_amount": 0,
  "capital_period_amortization_amount": 0,
  "remain_balance_amount": 10000.0
  }
  ],
  [...]
```

- Persistencia: En DBMS PostgreSQL ORM (sql model, sqlalchemy)

}

3 Requerimientos
Backend {

Auditoría asíncrona: fastapi BackgroundTasks



}

3 Requerimientos Frontend{

- Formulario de entada:
- Manejo inteligente estado:
  - Local Storage: Hook reutilizable personalizado (useState)
  - Cambio formulario afecta Tabla: useEffect dependiente de state monto, estado monto actualizado en onChange formulario

}

4 Buenas prácticas {

01

Programación defensiva: Validación

02

Componentes reutilizables: generalidad

03

Manejo errores: HHTP y Excepciones

04

Convenciones y formato código

05

Seguridad: CORS y ENV

06

Nombrado significativo

07

08

}

Areas de mejora {

- Pruebas unitarias: permitan refactorizciones eficientes.
- Routing
- Servicio gateway
- Autenticación

}