- **Diagrama de Casos de Uso**: Mostra a interação entre usuários e sistema, focando nas funcionalidades do sistema.
- **Diagrama de Sequência**: Exibe a interação temporal entre objetos, mostrando as mensagens trocadas.
- **Diagrama de Classes**: Ilustra a estrutura de classes, interfaces e seus relacionamentos.
- **Diagrama de Objetos**: Mostra instâncias de classes e seus links em um momento específico de execução.
- **Diagrama de Estado:** Mostra os estados de um objeto e as transições entre esses estados ao longo do tempo.
- **Diagrama de Atividades:** Representa o fluxo de atividades em um processo ou sistema, destacando as ações e decisões.

Diagrama de Casos de Uso:

Diagrama de Casos de Uso é uma representação visual da iteração entre um sistema e seus usuários. Também chamados de atores. Ele ilustra como um usuário realizará ações e interagirá com um sistema específico, como um site ou aplicativo. O diagrama é composto por diferentes formas e cenários possíveis de uso de um sistema, e é representado por uma forma oval rotulada na notação UML (Unified Modeling Language).

Os atores são representados por bonecos palito, e as associações são representadas por uma linha entre atores e casos de uso. A caixa de limite do sistema define um escopo do sistema para os casos de uso, e todos os casos de uso fora da caixa são considerados fora do escopo do sistema.

O diagrama de caso de uso é ideal para representar as metas de interações entre sistemas e usuários, definir e organizar requisitos funcionais no sistema, e especificar o contexto e os requisitos do sistema. Ele é uma ferramenta importante para a análise e modelagem de sistemas orientados a objetos, e ajuda a identificar funções e como os papéis interagem com eles, fornecendo uma visão de alto nível do sistema.

Os diagramas de caso de uso são uma ótima ferramenta para projetar processos e sistemas, garantindo o desenvolvimento de sistemas corretos e eficientes que atenderão adequadamente aos seus objetivos, capturando as necessidades e objetivos dos usuários.

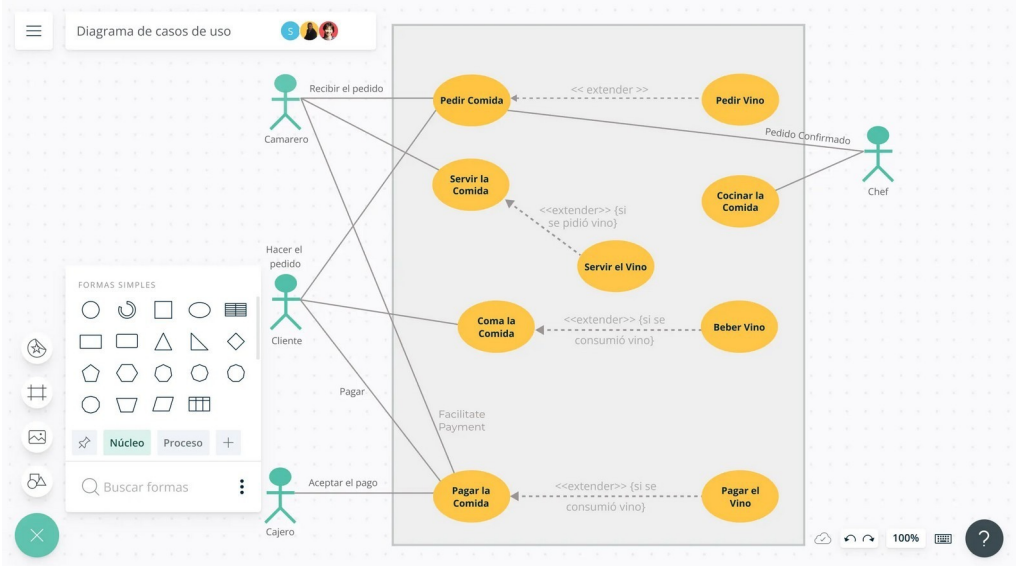


Diagrama de Sequência:

O Diagrama de Sequência é um tipo de diagrama UML usado na engenharia de software para descrever o comportamento dinâmico de um sistema e como diferentes objetos ou componentes interagem entre si. Ele é particularmente útil para modelar o fluxo de interações entre objetos ou componentes, demonstrar o comportamento dinâmico de um sistema, projetar e documentar o comportamento de um sistema ou de um recurso específico, identificar possíveis problemas no projeto do sistema, como gargalos ou problemas de sincronização, e para comunicação e discussão do comportamento do sistema com as partes interessadas.

O diagrama de sequência é ordenado por tempo, indicando que as interações exatas entre os objetos são exibidas passo a passo.

Ele é baseado no diagrama de casos de uso e é comumente usado durante as fases de projeto e documentação do desenvolvimento de software.

O diagrama de sequência é composto por linhas de vida, que representam os objetos ou componentes envolvidos no processo, e mensagens, que representam a comunicação entre esses objetos ou componentes.

As mensagens podem ser síncronas ou assíncronas, indicando se a comunicação é instantânea ou se ocorre um tempo de espera entre a emissão e a recepção da mensagem.

O Diagrama de Sequência é uma ferramenta extremamente útil no processo de desenvolvimento de software. Ele permite visualizar de forma clara e concisa a interação entre os objetos em um sistema, facilitando a compreensão do fluxo de execução e identificando possíveis problemas ou gargalos.

Além disso, o Diagrama de Sequência também pode ser utilizado para documentar e comunicar o funcionamento de um sistema para outras partes interessadas, como desenvolvedores, analistas e clientes.

Para criar um Diagrama de Sequência, é necessário seguir alguns passos, como identificar os objetos ou componentes envolvidos no processo, definir as mensagens que serão trocadas entre esses objetos ou componentes, e organizar as mensagens em ordem temporal.

É importante evitar erros comuns, como adicionar detalhes desnecessários, usar diagramas desatualizados, e não deixar espaço em branco entre o texto do caso de uso e a seta de mensagem.

Em resumo, o Diagrama de Sequência é uma ferramenta poderosa e amplamente utilizada no desenvolvimento de software, que permite visualizar de forma clara e concisa a interação entre os objetos em um sistema, facilitando a compreensão do fluxo de execução e identificando possíveis problemas ou gargalos.

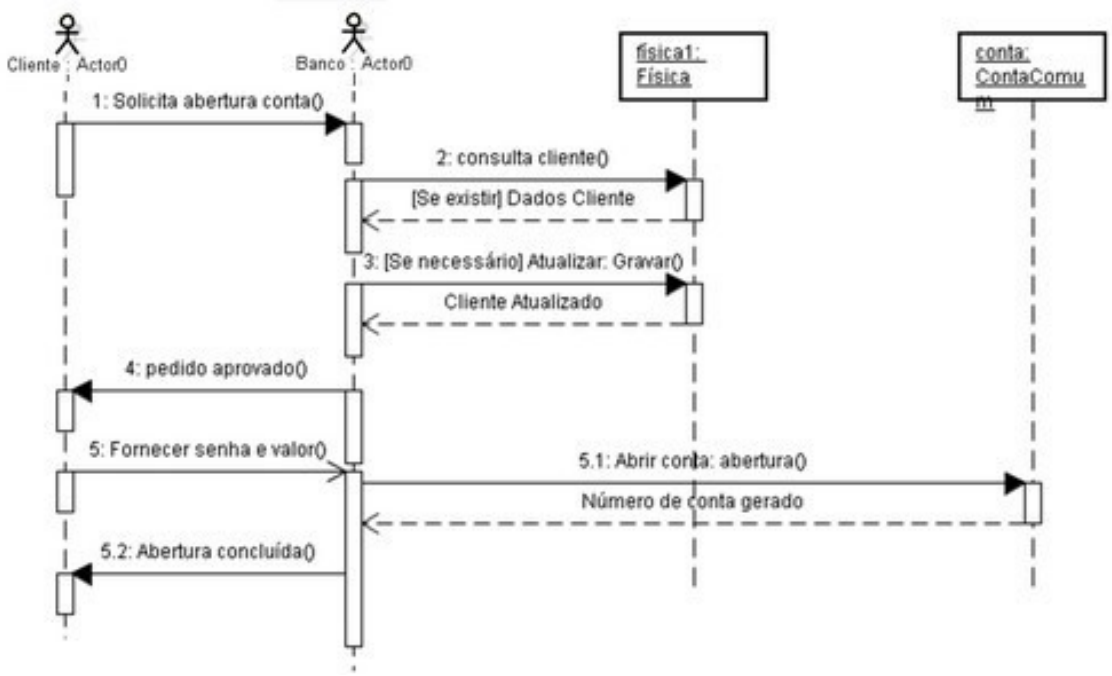


Diagrama de Classes:

Diagrama de Classes são um tipo de diagrama estrutural UML que ilustra as classes, interface e relacionamentos entre elas. Eles são fundamentais para o processo de modelagem de objetos e são os principais diagramas da UML.

Os diagramas de classes descrevem atributos e operações de uma classe, bem como as restrições que descrevem como os objetos podem ser conectados, além de representar os tipos de objetos no sistema e os relacionamentos entre eles.

As classes são representadas por um retângulo dividido em três partes: a primeira linha contém o nome da classe, a segunda linha contém os atributos e a terceira linha contém os métodos ou operações.

Existem diferentes níveis de acesso para as classes, como público, privado, protegido, pacote, derivado e estático, que são representados por diferentes símbolos.

Além disso, os diagramas de classes podem representar diferentes relacionamentos entre classes, como dependência, associação e generalização (ou herança). Os diagramas de classes são uma representação da estrutura e relações das classes que servem de modelo para objetos em programação.

Eles são muito úteis para ilustrar modelos de dados, representar a estrutura de um sistema e ajudar a entender a relação entre diferentes partes do sistema.

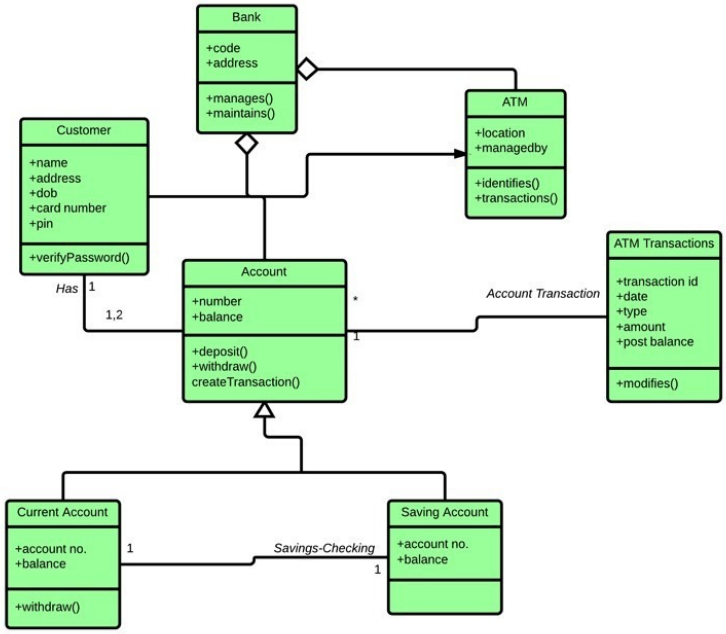


Diagrama de Objetos:

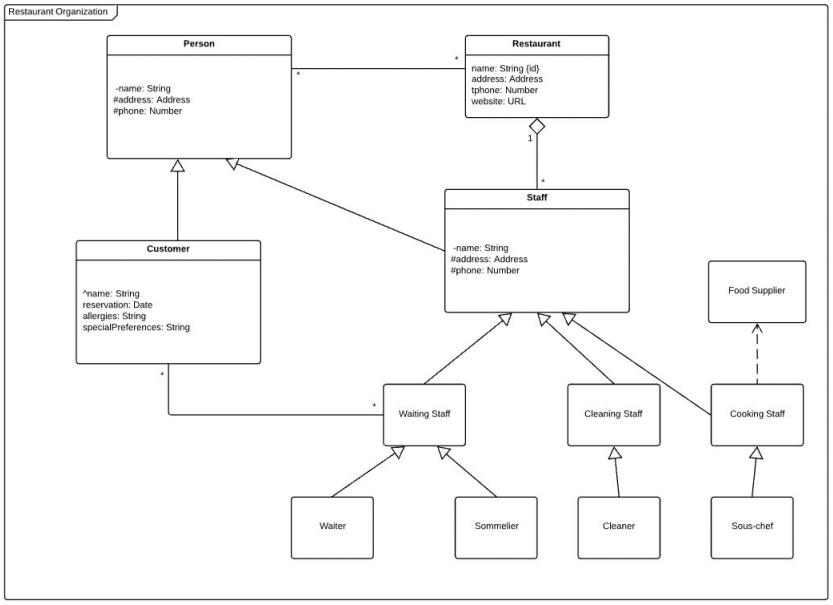
Diagrama de objetos são uma representação visual de um conjunto de instâncias existentes em determinado momento de execução de um programa. Eles são uma “fotografia” das instâncias das classes mostrando os objetos e os links entre eles.

Diagrama de objetos são uma variação de diagramas de classes, mas não são a mesma coisa. Enquanto diagramas de classes representam a estrutura de relações de classes que servem de modelo para objetos, diagramas de objetos mostram instâncias e links entre estas instâncias.

Diagramas de objetos são úteis para depurar uma funcionalidade do sistema, verificar se o sistema foi desenvolvido conforme os requisitos, analisar como a regra de negócio do sistema responde, e exemplificar diagramas complexos para facilitar a compreensão do sistema.

Eles são representados por retângulos divididos em três partes: a primeira linha contém o nome do objeto, a segunda linha contém os atributos e seus valores, e a terceira linha contém os métodos ou operações que podem ser executados no objeto.

Diagramas de objetos são complementares aos diagramas de classes, e podem ser criados utilizando diferentes ferramentas de modelagem, como o Enterprise Architect, Visio, Lucidchart, e Microsoft Visio para a Web.



Resumo:

O Diagrama de Casos de Uso representa a interação entre usuários e um sistema, descrevendo as funcionalidades do sistema do ponto de vista do usuário.

O Diagrama de Sequência mostra a interação entre objetos ao longo do tempo, exibindo as mensagens trocadas entre eles.

O Diagrama de Classes ilustra a estrutura de classes, interfaces e seus relacionamentos em um sistema.

O Diagrama de Objetos exibe instâncias de classes e os links entre elas em um momento específico de execução do programa.

UML introdução - Exercícios - by Gemini - Gean Bandeira - SP3030075

1. Qual das seguintes NÃO é um tipo de diagrama UML?

(a) Diagrama de Classe

(b) Diagrama de Componente

(c) Diagrama de Sequência

(d) **Diagrama de Banco de Dados**

Alternativa Correta: D.

2. Qual é o objetivo de um diagrama de classe?

(a) Mostrar as interações entre objetos em um sistema

(b) Representar a estrutura estática de um sistema

(c) Descrever o comportamento de um sistema ao longo do tempo

(d) Definir a implantação de um sistema em hardware

Alternativa Correta: B.

3. Qual é a diferença entre uma associação e um relacionamento de agregação em UML?

(a) Uma associação é um relacionamento mais forte entre classes do que uma agregação.

(b) Uma agregação é um relacionamento parte-todo entre classes, enquanto uma associação é um relacionamento geral entre classes.

(c) As agregações só podem ser representadas com setas unidirecionais, enquanto as associações podem ser representadas com setas bidirecionais.

(d) As agregações são sempre representadas com diamantes nas extremidades das linhas, enquanto as associações são sempre representadas com linhas sólidas.

Alternativa Correta: B

4. Qual é o objetivo de um diagrama de sequencia em UML?

(a) Mostrar a estrutura geral de um sistema

(b) **Descrever as interações entre objetos em um sistema ao longo do tempo**

(c) Descrever o comportamento de um sistema em resposta a eventos

(d) Definir a implantação de um sistema em hardware

Alternativa Correta: B.

5. Qual é a diferença entre um caso de uso e um ator em UML?

(a) Um caso de uso é uma pessoa que interage com um sistema, enquanto um ator é uma representação de uma função do usuário.

(b) **Um caso de uso é uma tarefa específica que um usuário pode realizar com um sistema, enquanto um ator é uma pessoa que interage com um sistema.**

(c) Os casos de uso são representados por figuras de pau, enquanto os atores são representados por ovais.

(d) Os casos de uso são sempre descritos em texto, enquanto os atores são sempre representados visualmente.

Alternativa Correta: B.

6. Qual é o objetivo de um diagrama de máquina de estado em UML?

(a) Mostrar as interações entre objetos em um sistema

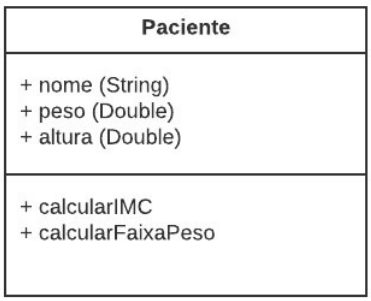
(b) Representar a estrutura estática de um sistema

(c) **Descrever o comportamento de um objeto ao longo do tempo em resposta a eventos**

(d) Definir a implantação de um sistema em hardware

Alternativa Correta: D.

Exercício 1: Paciente

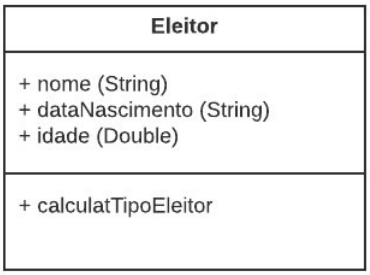



```
import java.util.Scanner;

import java.util.*; class

Paciente{     private String

nome;     public

Paciente(String nome){

this.nome = nome;


   }
```

```
}

public class Main{ public static void

        main(String[] args) {

    Scanner scanner = new Scanner(System.in);

    System.out.println("Nome: ");

    String nome = scanner.nextLine();

    System.out.println("Peso: ");

    Double peso = scanner.nextDouble();

    System.out.println("Altura: ");

    Double altura = scanner.nextDouble();

Double calcularIMC = peso/(altura * altura);

if(calcularIMC < 20){

        System.out.println("\nAbaixo do peso\n");

        System.out.println("IMC: " + calcularIMC);

    }else if(calcularIMC <25){

        System.out.println("\nPeso normal\n");

        System.out.println("IMC: " + calcularIMC);     }else if(calcularIMC < 30){

        System.out.println("\nExcesso de peso\n");

        System.out.println("IMC: " + calcularIMC);
```

```
    }else if(calcularIMC < 35){

        System.out.println("\nObesidade\n");

        System.out.println("IMC: " + calcularIMC);

    }else if(calcularIMC >= 35){

        System.out.println("\nObesidade Morbida\n");

        System.out.println("IMC: " + calcularIMC);

    }else{

        System.out.println("\nFez coisa errada camarada\n");

    }

          }

}
```

Exercício 2: Eleitor




```
import java.util.Scanner;
import java.util.*;

class Paciente{     private String
nome; private String
dataNascimento;
          private Double idade;
```

```
    public Paciente(String nome, String dataNascimento, Double idade)
{       this.nome = nome;
        this.dataNascimento = dataNascimento;
        this.idade = idade;
    }
}

public class Main{ public static void
          main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    System.out.println("Nome: ");
    String nome = scanner.nextLine();
    System.out.println("Data de nascimento: (dd/mm/yyyy)");
    String dataNascimento = scanner.nextLine();
    System.out.println("Idade: ");
    Double idade = scanner.nextDouble();

    if(idade < 16){
        System.out.println("\nNao Eleitor\n");
        System.out.println(idade);
    }else if(idade <18){
        System.out.println("\nEleitor Facultativo\n");
        System.out.println(idade);
    }else if(idade < 65){
        System.out.println("\nEleitor Obrigatorio\n");
        System.out.println(idade);
    }else if(idade > 65){
        System.out.println("\nEleitor Facultativo \n");
        System.out.println(idade);
    }else{
        System.out.println("\nFez coisa errada camarada\n");
    }
          }
}
```

IFSP
DIT-SPO
Tecnologia em Analise e Desenvolvimento de Sistemas

SPOLOG2

Gean Carlos de Sousa Bandeira, SP3030075.

Cenário: Sistema de Biblioteca

Descrição

Você foi contratado para desenvolver um sistema de gerenciamento para uma biblioteca. O sistema deve permitir que os bibliotecários gerenciem livros, membros da biblioteca e empréstimos. O sistema deve ter as seguintes funcionalidades:

1. **Gerenciamento de Livros:**
   Adicionar, editar e remover livros do catálogo.
   Cada livro deve ter um título, autor(es), ISBN, ano de publicação e categoria.

2. **Gerenciamento de Membros:**
   Registrar novos membros, editar e remover informações de membros. Cada membro deve ter um nome, endereço, telefone e número de membro exclusivo.

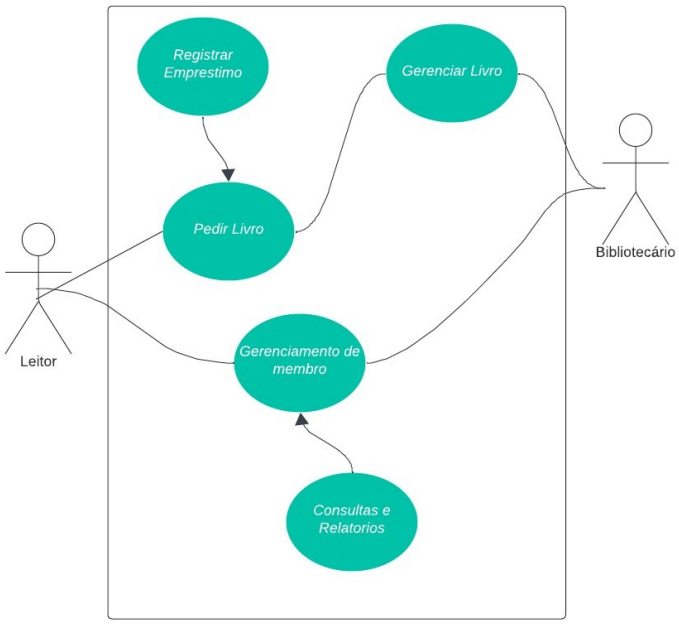
3. **Empréstimos de Livros:**
   Registrar o empréstimo de um livro para um membro.
   Registrar a devolução de livros.
   Cada empréstimo deve ter a data de início, data de devolução prevista e a data de devolução real.

4. **Consultas e Relatórios*:**
   Permitir a busca por livros e membros.
   Gerar relatórios de livros emprestados e membros com livros em atraso.

Exercício

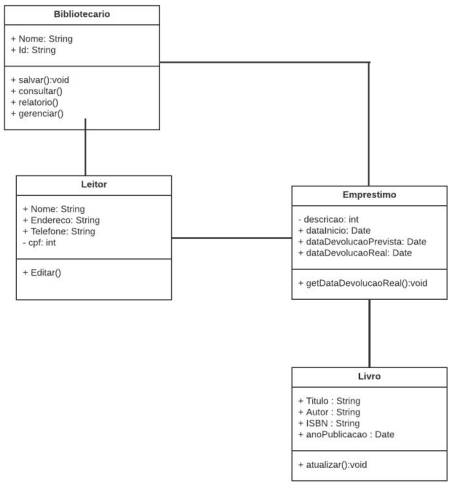
Com base nesse cenário, complete as seguintes tarefas:

1. Diagrama de Casos de Uso
   Identifique e desenhe os principais casos de uso para o sistema de biblioteca.
   Inclua os atores relevantes.



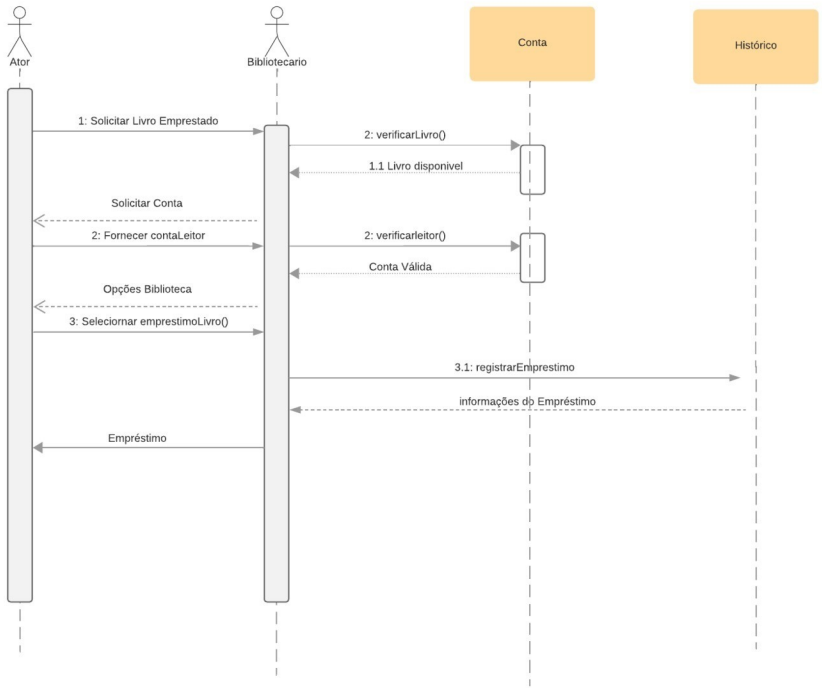
2. Diagrama de Classes

   Desenhe um diagrama de classes que modele as entidades principais do sistema de biblioteca e seus relacionamentos.



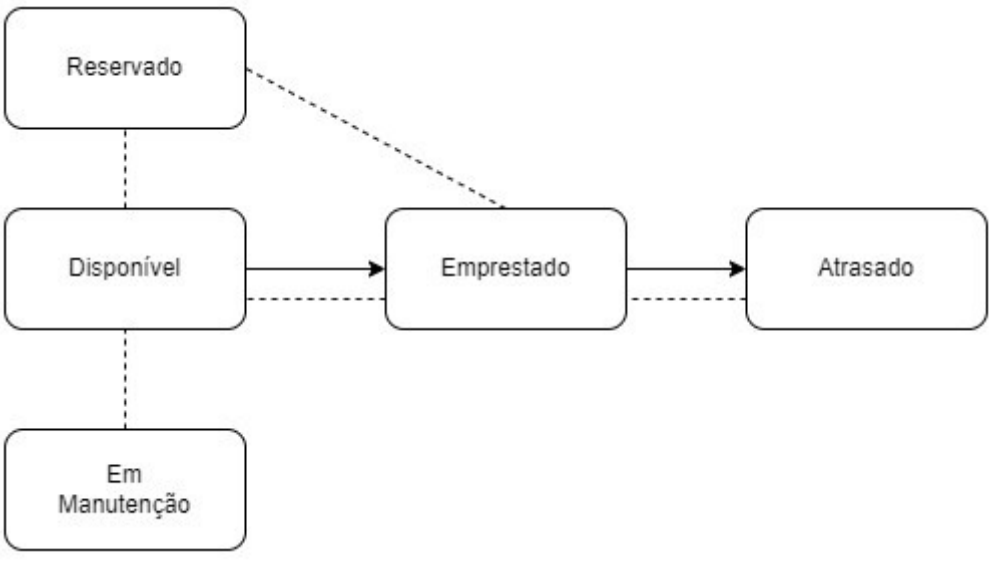
3. Diagrama de Sequência

   Crie um diagrama de sequência para representar o processo de empréstimo de um livro.



4. Diagrama de Estado

   Desenhe um diagrama de estados para a classe "Livro", representando os diferentes estados pelos quais um livro pode passar (disponível, emprestado, atrasado, etc.).



5. Diagrama de Atividades

Crie um diagrama de atividades que descreva o processo de devolução de um livro, incluindo a verificação de atraso e a atualização dos registros.