

# Logo - Modeligado

Renaldo Silva Santino - 119210334

Portfólio Behance: <[Renaldo Silva on Behance](#)>

**Objetivo:** o projeto tem foco no desenvolvimento de uma logo tomando como base as características do sistema Modeligado. Com isso, serão apresentados alguns protótipos acompanhados de uma breve exemplificação dos elementos que os compõem. Ademais, vale salientar que todos foram pensados de forma a manter a identidade visual minimalista do sistema.

**Processo:** parte do desenvolvimento se deu de forma manual a partir de esboços (essa etapa não está presente neste documento), posteriormente alguns deles foram selecionados e refeitos de forma digital com o intuito de refinar a ideia, com isso, passou-se para a última etapa na qual os protótipos foram modelados em sua versão final.

## Softwares utilizados:

- Medibang paint Pro <[MediBang Paint](#)>;
- Inkscape <[Desenhe Livrementemente | Inkscape](#)>.

## Cores base em RGBA:

- Creme: fffe0ff (cor utilizada nas classes do Modeligado);
- Preto: 000000ff.

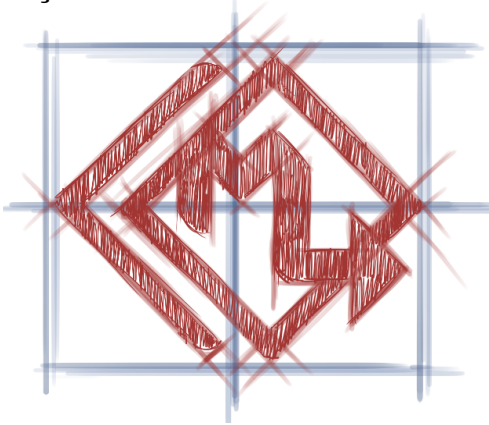
## Palavras-chave:

- Modeligado / ML;
- Classes / Camadas;
- Relações (relacionamento entre classes);
- Orgânico (representa a mutabilidade que um sistema pode ter).

Obs.: foram utilizadas formas arredondadas em algumas partes dos protótipos com o objetivo de representar a “parte orgânica dos sistemas”.

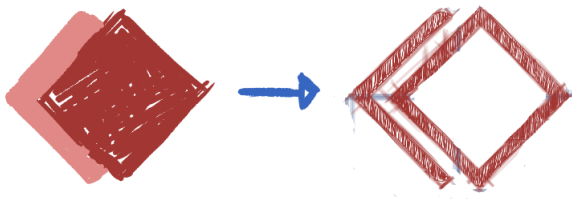
## Protótipo 1

### Esboço:



### Exemplificação:

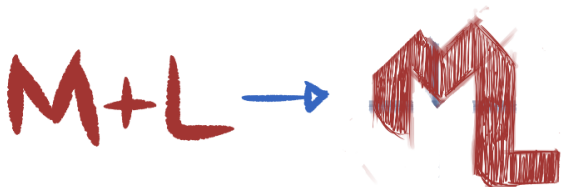
- 1) Representação das camadas de um sistema.



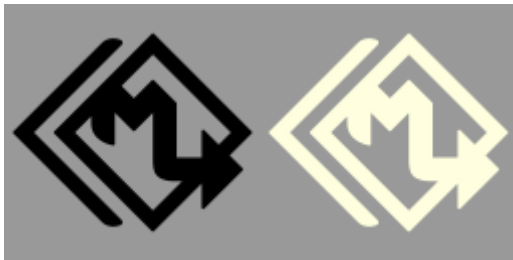
- 2) Representação de uma relação entre classes.



- 3) ML - Modeligado.



## Versões finais:



obs.: fundo cinza aplicado somente por questões de visualização.



## Exemplo de aplicação:

← → ↻ <https://matheusgr.github.io/modeligado/edit.html>

 **ODELIGADO**

☒ Auto Update - ☐ Trace - No errors detected... UML (F2) -

```
1 // diagrama de classes de exemplo
2 // duas barras definem comentários
3
4 Main
5 association UsuarioController
6 directionalAssociation SistemaController // A ser implementado depois
7 ---
8 ---
9 ---
10
11 UsuarioController
12 composes UsuarioRepository // Outro tipo de associação: Aggregates
13 ---
14 - usuarioRepository: UsuarioRepository
15 ---
16 + UsuarioController()
17 + adicionaTrabalhador(nome: str, code: int): void
18 + adicionaAssociado(nome: str, code: int, empresa: str): void
```

GoJS 2.1 evaluation

(c) 1996-2021 Northwoods Software

Not for distribution or reproduction

gojs not

**Usuar**

- usuarioRepository: Usuar

+ UsuarioController()

+ adicionaTrabalhador(nome: str, code: int): void

+ adicionaAssociado(nome: str, code: int, empresa: str): void

+ localizaUsuarios(nome: str, code: int): void

+ adicionaBonus(code: int, valor: int): void

+ totalizaBonus(code: int): void

**Main**

- usuarios: Map<id, Usuario>

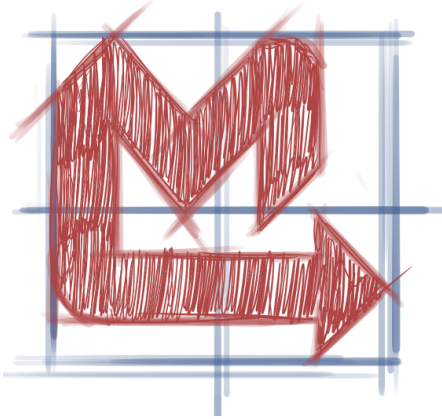
+ UsuarioRepository()

+ adicionaTrabalhador(nome: str, code: int): void

+ adicionaAssociado(nome: str, code: int, empresa: str): void

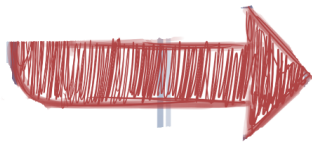
## Protótipo 2

### Esboço:



### Exemplificação:

- 1) Representação de uma relação entre classes.



- 2) ML - Modeligado.



## Versões finais:



obs.: fundo cinza aplicado somente por questões de visualização.



## Exemplo de aplicação:

The screenshot shows the Modeligado web application interface. The browser address bar displays <https://matheusgr.github.io/modeligado/edit.html>. The application header features the Modeligado logo and a status bar with the following elements: ☒ Auto Update, ☐ Trace, No errors detected..., and UML (F2).

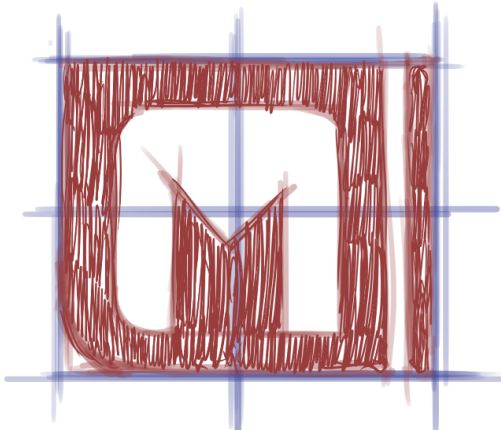
The main content area is divided into two panels. The left panel displays a UML class diagram in text format:

```
1 // diagrama de classes de exemplo
2 // duas barras definem comentários
3
4 Main
5 association UsuarioController
6 directionalAssociation SistemaController // A ser implementado depois
7 ---
8 ---
9 ---
10
11 UsuarioController
12 composes UsuarioRepository // Outro tipo de associação: Aggregates
13 ---
14 - usuarioRepository: UsuarioRepository
15 ---
16 + UsuarioController()
17 + adicionaTrabalhador(nome: str, code: int): void
18 + adicionaAssociado(nome: str, code: int, empresa: str): void
```

The right panel displays a graphical UML diagram. It shows a class named **Usuar** (likely **Usuario**) with the following methods: `+ adicionaTrabalhador(nome: str, code: int): void`, `+ adicionaAssociado(nome: str, code: int, empresa: str): void`, `+ localizaUsuario(code: int): void`, `+ adicionaBonus(code: int, valor: float): void`, and `+ totalizaBonus(code: int): void`. Below this, a **Main** class is shown with a `- usuarios: Map<id, Usuario>` attribute and a `+ UsuarioRepository()` method. A line connects the **Usuar** class to the **Main** class.

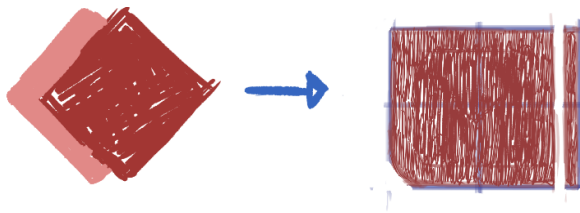
## Protótipo 3

### Esboço:

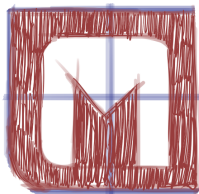


### Exemplificação:

- 1) Representação das camadas de um sistema.



- 2) M - Modeligado.



## Versões finais:



obs.: fundo cinza aplicado somente por questões de visualização.



## Exemplo de aplicação:

The screenshot shows the Modeligado web application interface. The browser address bar displays <https://matheusgr.github.io/modeligado/edit.html>. The application header features the Modeligado logo and a status bar with options: ☒ Auto Update, ☐ Trace, and No errors detected... On the right, it shows 'UML (F2)'.

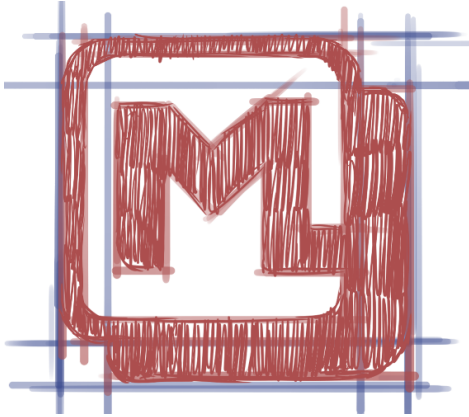
The main area is divided into two panels. The left panel contains a code editor with the following UML class diagram code:

```
1 // diagrama de classes de exemplo
2 // duas barras definem comentários
3
4 Main
5 association UsuarioController
6 directionalAssociation SistemaController // A ser implementado depois
7 ---
8 ---
9 ---
10
11 UsuarioController
12 composes UsuarioRepository // Outro tipo de associação: Aggregates
13 ---
14 - usuarioRepository: UsuarioRepository
15 ---
16 + UsuarioController()
17 + adicionaTrabalhador(nome: str, code: int): void
18 + adicionaAssociado(nome: str, code: int, empresa: str): void
```

The right panel displays the generated UML class diagram. It shows two classes: **UsuarioController** and **UsuarioRepository**. **UsuarioController** has methods: `+ adicionaTrabalhador(nome: str, code: int): void`, `+ adicionaAssociado(nome: str, code: int, empresa: str): void`, `+ localizaUsuarios(nome: str): void`, `+ localizaUsuario(code: int): void`, `+ adicionaBonus(code: int, valor: float): void`, and `+ totalizaBonus(code: int): void`. **UsuarioRepository** has a method: `+ adicionaTrabalhador(nome: str, code: int): void`. The diagram also shows a **Main** class with a `- usuarios: Map<id, Usuario>` attribute and a `+ UsuarioRepository()` method.

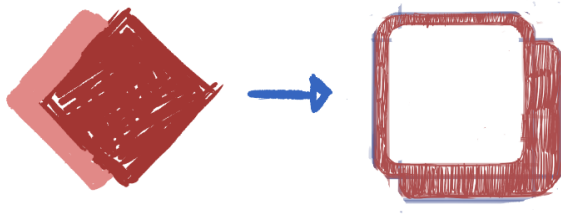
## Protótipo 4

**Esboço:**

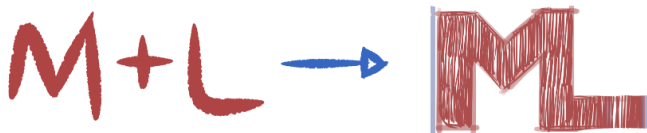


**Exemplificação:**

- 1) Representação das camadas de um sistema.



- 2) M - Modeligado.





## Versões finais:



obs.: fundo cinza aplicado somente por questões de visualização.



## Exemplo de aplicação:

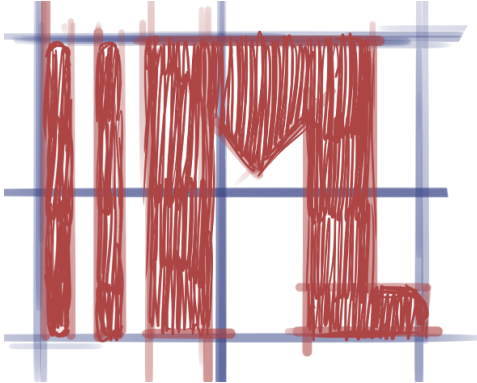
The screenshot shows the Modeligado web application interface. At the top, there's a browser address bar with the URL <https://matheusgr.github.io/modeligado/edit.html>. Below the address bar is the Modeligado logo and a navigation bar with options: ☒ Auto Update, ☐ Trace, and No errors detected... On the right of the navigation bar is a button labeled UML (F2). The main area is split into two panels. The left panel is a code editor showing a GoJS diagram definition. The right panel is a UML diagram viewer showing the generated diagram. The code in the left panel is as follows:

```
1 // diagrama de classes de exemplo
2 // duas barras definem comentários
3
4 Main
5 association UsuarioController
6 directionalAssociation SistemaController // A ser implementado depois
7 ---
8 ---
9 ---
10
11 UsuarioController
12 composes UsuarioRepository // Outro tipo de associação: Aggregates
13 ---
14 - usuarioRepository: UsuarioRepository
15 ---
16 + UsuarioController()
17 + adicionaTrabalhador(nome: str, code: int): void
18 + adicionaAssociado(nome: str, code: int, empresa: str): void
```

The UML diagram in the right panel shows a class hierarchy with 'Main' at the top, 'UsuarioController' below it, and 'UsuarioRepository' below 'UsuarioController'. There is an association between 'Main' and 'UsuarioController', and another between 'UsuarioController' and 'UsuarioRepository'. The 'UsuarioController' class has methods: `+ adicionaTrabalhador(nome: str, code: int): void`, `+ adicionaAssociado(nome: str, code: int, empresa: str): void`, `+ localizaUsuarios(nome: str): void`, `+ localizaUsuario(code: int): void`, `+ adicionaBonus(code: int, valor: float): void`, and `+ totalizaBonus(code: int): void`. The 'UsuarioRepository' class has a property `- usuarios: Map<id, Usuario>` and a method `+ UsuarioRepository()`.

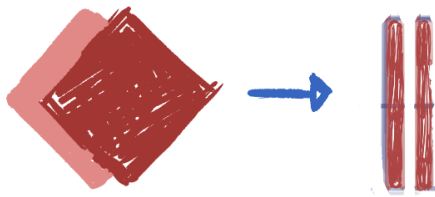
## Protótipo 5

### Esboço:



### Exemplificação:

- 1) Representação das camadas de um sistema.



- 2) ML - Modeligado.



## Versões finais:



obs.: fundo cinza aplicado somente por questões de visualização.



## Exemplo de aplicação:

← → ↻ 🔒 <https://matheusgr.github.io/modeligado/edit.html>

# MODELIGADO

☒ Auto Update - ☐ Trace - No errors detected... UML (F2) -

```
1 // diagrama de classes de exemplo
2 // duas barras definem comentários
3
4 Main
5 association UsuarioController
6 directionalAssociation SistemaController // A ser implementado depois
7 ---
8 ---
9 ---
10
11 UsuarioController
12 composes UsuarioRepository // Outro tipo de associação: Aggregates
13 ---
14 - usuarioRepository: UsuarioRepository
15 ---
16 + UsuarioController()
17 + adicionaTrabalhador(nome: str, code: int): void
18 + adicionaAssociado(nome: str, code: int, empresa: str): void
```

GoJS 2.1 evaluation  
(c) 1996-2016 by Microsoft Corporation  
Not for redistribution or reproduction in any form  
gojs.rtf

### Usuar

- usuarioRepository: UsuarioRepository
- + UsuarioController()
  - + adicionaTrabalhador(nome: str, code: int): void
  - + adicionaAssociado(nome: str, code: int, empresa: str): void
  - + localizaUsuario(code: int): void
  - + adicionaBonus(code: int, valor: int): void
  - + totalizaBonus(code: int): void

Main

Us

- usuarios: Map<id, Usuario>
- + UsuarioRepository()
- + adicionaTrabalhador(nome: str, code: int): void