

## Curitiba Hero Registry

Curitiba Heroes Association

<chá>

Cliente: **CHÁ - Curitiba Hero Association**

Responsáveis no cliente: **Laudelino Cordeiro Bastos**

Projeto: **Curitiba Hero Registry**

Versão: **entrega final (2.6)**

**Curitiba  
2022**

## **Curitiba Hero Registry**

**Responsáveis no cliente:** **Laudelino  
Cordeiro Bastos**

**Responsáveis pelo projeto e  
desenvolvimento:**  
**Bernardo Vendruscolo Mendes**  
**Daniel Augusto Pires de Castro**  
**David Segalle**  
**Thaís Say de Carvalho**

**Curitiba  
2022**

## Histórico de Modificações

Data	Versão	Descrição	Autor
<dd/mm/aa>	<x.x>	<Descrição da modificação>	<nome do autor>
14/09/22	0.1	Inclusão das perguntas de requisitos	Bernardo, Thaís, David, Daniel
05/10/22	0.2	Inclusão das respostas dos requisitos	Bernardo, Thaís, David, Daniel
12/10/22	0.3	Estudo de viabilidade	Bernardo, Thaís, David, Daniel
15/10/22	0.4	Planning poker	Bernardo, Thaís, David, Daniel
15/10/22	0.5	Criação de um UML	Bernardo, Thaís, David, Daniel
18/10/22	0.6	Criação do diagrama de casos de uso	Bernardo, Thaís, David, Daniel
20/10/22	0.7	Especificação dos casos de uso	Bernardo, Thaís, David, Daniel
22/10/22	1.0	Desenvolvimento de um <i>website</i>	Bernardo, Thaís, David, Daniel
22/10/22	1.1	Criação de um diagrama de objetos	Bernardo, Thaís, David, Daniel
23/10/22	1.2	Criação de um diagrama de casos de uso	Bernardo, Thaís, David, Daniel
23/10/22	1.3	Desenvolvimento da página principal	Bernardo, Thaís, David, Daniel
24/10/22	1.4	Desenvolvimento do blog e ações de autenticação	Bernardo, Thaís, David, Daniel
25/10/22	1.5	Complementação do UML	Bernardo, Thaís, David, Daniel
29/10/22	2.0	Desenvolvimento do diagrama de comunicação	Bernardo, Thaís, David, Daniel

05/11/22	2.1	Desenvolvimento do diagrama de entidade-relacionamento	Bernardo, Thaís, David, Daniel
22/11/22	2.2	Atualização do código para corresponder aos diagramas	Bernardo, Thaís, David, Daniel
28/11/22	2.3	Revisão dos diagramas e atualizações mínimas	Bernardo, Thaís, David, Daniel
06/11/22	2.4	Desenvolvimento do diagrama de estados	Bernardo, Thaís, David, Daniel
10/11/22	2.5	Desenvolvimento do diagrama de atividades	Bernardo, Thaís, David, Daniel
13/11/22	2.6	Revisão do relatório	Bernardo, Thaís, David, Daniel

# SUMÁRIO

<b>1 INTRODUÇÃO</b>	<b>8</b>
1.1 Objetivo Geral	8
1.2 Objetivos Específicos	8
1.3 Conteúdo do Plano do Projeto	8
<b>2 LEVANTAMENTO DE REQUISITOS</b>	<b>9</b>
2.1 Questões Organizacionais	9
2.2 Questões Econômicas	10
2.3 Questões Técnicas	11
2.4 Questões Operacionais	12
2.5 Requisitos Funcionais, Não Funcionais, Restrições de Projeto e Requisitos de Experiência do Usuário	13
2.6 Estimativa de esforço de software com planning poker	14
<b>3 ESTUDO DE VIABILIDADE</b>	<b>15</b>
3.1 Viabilidade Organizacional	15
3.2 Viabilidade Econômica	15
3.3 Viabilidade Técnica	15
3.4 Viabilidade Operacional	16
3.5 Recursos a serem utilizados	16
<b>4 RESULTADOS</b>	<b>17</b>
4.1 Conteúdo dos Resultados	17
4.2 Modelagem	17
<b>5 CONCLUSÕES</b>	<b>48</b>
<b>6 REFERÊNCIAS BIBLIOGRÁFICAS</b>	<b>49</b>

## LISTA DE FIGURAS

- Figura 1.** Página de registro de Super-Heróis. 17
- Figura 2.** Diagrama de Casos de Uso até o presente momento de desenvolvimento do projeto. 18
- Figura 3.** Vista diagrama de classes completo. 28
- Figura 4.** Espaço Entities no diagrama de classes. 29
- Figura 5.** Espaço Managers no diagrama de classes. 30
- Figura 6.** Relação de PersonList com outros objetos do sistema. 33
- Figura 7.** Relação de HeroesList com outros objetos do sistema. 34
- Figura 8.** Relação de VillainList com outros objetos do sistema. 35
- Figura 9.** Diagrama de casos de uso para a visualização da lista de heróis. 36
- Figura 10.** Diagrama de casos de uso para a visualização da lista de vilões. 36
- Figura 11.** Diagrama de casos de uso para a ação de registrar um novo usuário. 37
- Figura 12.** Diagrama de casos de uso para a ação de criar um novo vilão. 37
- Figura 13.** Diagrama de casos de uso para a atualização dos dados cadastrais de um herói. 38
- Figura 14.** Diagrama de comunicação para a visualização da lista de heróis. 38
- Figura 15.** Diagrama de comunicação para a visualização da lista de vilões. 39
- Figura 16.** Diagrama de comunicação para a ação de registrar um novo usuário. 39
- Figura 17.** Diagrama de comunicação para a ação de criar um novo vilão. 40
- Figura 18.** Diagrama de comunicação para a ação de criar um novo vilão. 40
- Figura 19.** Conversão da classe Person em UML para DER. 41
- Figura 20.** Conversão da classe Villain em UML para DER. 41
- Figura 21.** Conversão da classe Hero em UML para DER. 42
- Figura 22.** Conversão da herança em UML para DER. 42

**Figura 23.** Diagrama Entidade-Relacionamento completo. 43

**Figura 24.** Diagrama de estado da classe BlogManager para caso de uso de visualização de lista de heróis. 44

**Figura 25.** Diagrama de estado da classe PersonList para caso de uso de visualização de lista de heróis. 44

**Figura 26.1** diagrama de estado da classe BlogManager para caso de uso de visualização de lista de vilões. 44

**Figura 26.2** diagrama de estado da classe DbManager para caso de uso de visualização de lista de vilões. 44

**Figura 27.** diagrama de estado da classe AuthenticationManager para caso de uso de registrar usuário. 45

**Figura 28.** diagrama de estado da classe BlogManager para caso de uso de registrar usuário. 45

**Figura 29.** diagrama de estado da classe BlogManager para caso de uso de castrar vilão. 45

**Figura 30.** diagrama de estado da classe DbManager para caso de uso de castrar vilão. 46

**Figura 31.** diagrama de estado da classe AuthenticationManager para caso de uso atualizar informações do usuário. 46

**Figura 32.** diagrama de estado da classe DbManager para caso de uso atualizar informações de usuário. 46

**Figura 33.** diagrama de atividades para a ação de atualizar os usuários na database. 47

**Figura 34.** diagrama de atividades para a ação de renderizar a lista de pessoas. 47

**Figura 35.** diagrama de atividades para a ação de retornar a database. 47

**Figura 36.** diagrama de atividades para a ação de inserir um Vilão na database. 47

**Figura 37.** diagrama de atividades para a ação de inserir um Herói na database. 47

**Figura 38.** diagrama de atividades para a ação de atualizar a lista de Heróis. 48

**Figura 39.** diagrama de atividades para a ação de atualizar a lista de Vilões. 48

**Figura 40.** diagrama de atividades para a ação de realizar login. 48

**Figura 41.** diagrama de atividades para a ação de remover pessoas da database. 48

**Figura 42.** diagrama de atividades para a ação de logout. 48



## **LISTA DE TABELAS E QUADROS**

**Tabela 1.** compilação de resultados referentes à realização do Planning Poker. 12

**Tabela 2.** Casos de uso identificados. 16

**Tabela 3.** Detalhamento caso de uso Visualizar lista de Heróis - UC001. 17

**Tabela 4.** Detalhamento caso de uso Visualizar lista de Vilões - UC002. 19

**Tabela 5.** Cadastrar-se - UC003. 21

**Tabela 6.** Detalhamento de caso de uso Registrar um Vilão - UC004. 23

**Tabela 7.** Detalhamento de caso de uso Editar informações de usuário - UC005. 25

**Tabela 8.** Dicionário de Informações - Classe Hero. 28

**Tabela 9.** Dicionário de Informações - Classe Hero. 28

**Tabela 10.** Dicionário de Informações - Classe PersonList. 29

**Tabela 11.** Dicionário de Informações - Classe AuthenticationManager. 29

**Tabela 12.** Dicionário de Informações - Classe BlogManager 29

**Tabela 13.** Dicionário de Informações - Init. 29

# 1 INTRODUÇÃO

Na cidade de Curitiba, incluindo regiões centrais e periféricas, os índices de criminalidade apresentam abundância significativa. Devido a isso, urge a necessidade do serviço prestado por super-heróis para combater a bandidagem na zona urbana. Por conseguinte, evidencia-se a demanda de um sistema o qual possibilita organizar o registro e expediente desses heróis a fim de gerenciar o trabalho efetivo conforme suas habilidades e competências.

## 1.1 Objetivo Geral

Implementar um sistema para heróis descobrirem vilões os quais devem ser combatidos, assim como aliados em potencial.

## 1.2 Objetivos Específicos

- Cadastrar super-heróis em um sistema, armazenando suas informações em um banco de dados.
- Permitir a navegação dos usuários cadastrados, podendo acessar informações relativas a elementos pertinentes em suas vidas.
- Possibilitar editar, excluir e consultar os dados a qualquer momento, seja pelo próprio usuário ou pelo administrador.

## 1.3 Conteúdo do Plano do Projeto

Este documento está dividido em cinco partes:

Capítulo 2: Levantamento de Requisitos.

Capítulo 3: Estudo de Viabilidade.

Capítulo 4: Resultados.

Capítulo 5: Conclusões.

Capítulo 6: Referências Bibliográficas.

## 2 LEVANTAMENTO DE REQUISITOS

Para fins de organizar e consolidar as propostas do projeto, foram levantados diversos requisitos os quais abrangem questões organizacionais, econômicas, técnicas e operacionais. Além do mais, houve a inclusão de requisitos funcionais, não funcionais, restrições do projeto e requisitos de experiência do usuário.

### 2.1 Questões Organizacionais

1. Quais os benefícios que o projeto trará para o cliente?

O Sistema possibilitará a socialização do cliente com demais heróis, além de também possibilitar a organização de metas de combate ao crime e possibilitar visualizar vilões.

2. Qual o objetivo do usuário?

Nesse aplicativo o objetivo do usuário será encontrar informações sobre outros super-heróis e encontrar vilões para combater.

3. Como proteger dados mais sigilosos, de modo a evitar vazamento de informações para agentes externos, em especial vilões?

Para evitar o vazamento de informações confidenciais para agentes externos, de modo a evitar o comprometimento da associação de heróis e do bem-estar dos heróis cadastrados, será limitado o acesso a informações confidenciais a administradores do sistema.

4. Quem poderá modificar os cadastros?

O cadastro no sistema e a posterior modificação das informações registradas será limitada ao próprio usuário responsável pelo cadastro. Contudo, o administrador poderá deletar usuários.

5. Quais dados serão protegidos?

Os dados protegidos concernem informações pessoais e de uso exclusivo administrativo do sistema, no caso nome pessoal e endereço.

6. Quais informações são necessárias para realizar o cadastro?

Para realizar o cadastro será necessário, por parte do usuário, o fornecimento do nome pessoal, do nome do herói, a classe (*tier*) de herói e

do endereço de moradia do indivíduo. Também, será requerido o correto preenchimento de um *captcha* de verificação de super-heróis, de modo a garantir que os usuários realizando cadastro próprio sejam de fato super-heróis.

7. Como hierarquizar heróis?

Os heróis serão hierarquizados de acordo com as suas habilidades, cada herói será classificado com uma letra de “C” a “S”, onde “S” representam os mais capazes de derrotar vilões poderosos.

8. Quais informações sobre heróis e vilões serão armazenadas?

Em relação aos heróis, será armazenada a classe, o nome do herói, o nome pessoal e o endereço de moradia do herói. Em relação aos vilões, serão armazenados o nome de vilão, a classe de vilão e área de atividade criminosa, e uma breve descrição de suas habilidades.

## 2.2 Questões Econômicas

1. Quais as demandas do mercado para este produto?

Foi verificado que há uma grande dificuldade de gerenciar os super-heróis e acompanhar os vilões, por essa razão diversos crimes que poderiam ser evitados ocorrem diariamente.

2. Quais os custos do desenvolvimento do sistema?

Para o desenvolvimento do sistema será necessário a contratação de uma equipe de desenvolvedores de software.

3. Qual o hardware necessário para utilizar o sistema?

Para utilizar o sistema será necessário que o usuário possua um computador pessoal, o qual será utilizado tanto para o cadastro quanto para a navegação usual no sistema.

4. Qual o custo contínuo do sistema?

Para a manutenção a longo prazo do sistema será necessário manter os servidores operando, assim como contratar uma equipe de administradores para realizar a manutenção do sistema e o cadastro de super-vilões.

5. Como arrecadar fundos para a manutenção do projeto?

Os fundos necessários para manter o sistema ativo serão adquiridos por meio da CHÁ, através de doações de usuários e membros da sociedade civil.

6. Quais serviços serão terceirizados?

Os serviços terceirizados serão os servidores utilizados para manter os dados, assim como rodar o sistema.

7. Como gerenciar as informações do usuário de modo a manter uma boa imagem da empresa com o público?

A administração responsável pelo sistema garante o correto gerenciamento de dados sigilosos dos usuários, de modo a manter informações confidenciais limitadas para acesso pelos administradores. Também garante-se que todo e qualquer herói cadastrado no sistema é de fato um super-herói de bem, e, portanto, que o acesso a informações de super-vilões está mantido em pessoal autorizado.

## 2.3 Questões Técnicas

1. Para que tipo de hardware o sistema é projetado?

O sistema é projetado para o uso em *personal computers*.

2. Qual o hardware necessário para manter o sistema?

Para manter o sistema operante é necessário um sistema de servidores, assim como *personal computers* para os administradores poderem realizar a manutenção do sistema.

3. Como administrar o sistema universalmente?

A administração do sistema será realizada de modo remoto, os administradores terão em seus computadores pessoais um login capaz de acessar o servidor com direitos administrativos.

4. Quais os direitos do administrador?

Enquanto administrador o indivíduo tem como direito a manutenção dos usuários cadastrados, de modo a poder remover heróis e vilões registrados. No entanto, o administrador não pode interferir nas informações pessoais dos heróis.

5. Como será realizada a manutenção do sistema?

A manutenção do sistema será realizada pelos administradores contratados.

6. Como armazenar os dados dos heróis?

Os dados sobre os heróis serão armazenados em um banco de dados.

7. Como transmitir informações sobre os vilões?

Os administradores responsáveis pelos sistemas podem coletar dados fornecidos pela sociedade e cadastrar super-vilões.

8. Quais serão as tecnologias utilizadas para o desenvolvimento do sistema?

Com o intuito de desenvolver o sistema será utilizado o Flask, um pequeno framework em python o qual facilita a criação de um site. Além desta, serão utilizadas outras ferramentas como SQLite para manusear o banco de dados, html e css para controlar a aparência e o formato da página e um Template Engine para determinar os dados a serem mostrados.

## 2.4 Questões Operacionais

1. Quais são os usuários do sistema?

O sistema será utilizado por super-heróis.

2. Quais dados o usuário pode acessar?

O usuário pode acessar seus dados pessoais, os dados públicos sobre os heróis e os dados sobre os vilões.

3. De que maneira o usuário interage com o sistema?

O usuário pode navegar por listas de heróis e de vilões cadastrados no sistema, acessar informações extras sobre cada um deles e alterar as suas informações pessoais.

4. Quem serão os administradores?

Os administradores serão membros da CHÁ.

5. Como encontrar vilões de interesse?

Os vilões de interesse serão encontrados por meio de uma aba navegável a qual mostra a classe e o nome dos vilões, ao selecionar o vilão desejado serão mostrados dados adicionais sobre ele.

6. Como visualizar outros heróis?

Há uma aba de super-heróis, em formato de lista, na qual é possível acessar perfis de mais heróis cadastrados.

7. Como interagir com outros heróis?

É possível visualizar os perfis de outros heróis, incluindo nomes de herói e classe.

8. Como os usuários poderão requisitar suporte?

Heróis cadastrados podem requisitar suporte.

## **2.5 Requisitos Funcionais, Não Funcionais, Restrições de Projeto e Requisitos de Experiência do Usuário**

- **Requisitos Funcionais**

RF01: O herói deve poder se cadastrar no sistema.

RF02: O usuário deve poder acessar informações públicas de outros heróis.

RF03: Os heróis devem poder acessar informações dos vilões.

RF04: O usuário deve poder alterar seus dados cadastrais.

- **Requisitos não Funcionais**

RNF01: As consultas ao banco de dados não podem ultrapassar 10 segundos.

RNF02: Manter informações confidenciais referentes a cada herói para acesso exclusivo de administradores.

RNF03: A navegação se dará por um gerenciador de blogs.

- **Restrições de Projeto**

RP01: O projeto deve ser desenvolvido utilizando a linguagem de programação python.

RP02: A navegação do banco de dados deve ser implementada utilizando-se SQLite.

- **Requisitos de experiência do Usuário**

RE01: O sistema deve ter um design simples e claro.

RE02: O herói deve se sentir seguro no ambiente.

## 2.6 Estimativa de Esforço de Software com Planning Poker

**Tabela 1.** compilação de resultados referentes à realização do Planning Poker.

Tarefa:	Bernardo	Daniel	David	Thaís	Tempo real
(RF01) Herói pode se cadastrar	8	13	13	5	10
(RF02) Usuário pode acessar informações públicas	8	3	3	5	7
(RF03) Os heróis devem poder acessar dados dos vilões	3	2	3	2	3π
(RF04) Usuário deve poder alterar seus dados cadastrais	8	5	13	2	2



## **3 ESTUDO DE VIABILIDADE**

Com o intuito de investigar a competência necessária para conceber o projeto, foi realizado um estudo de viabilidade do potencial dos desenvolvedores e dos recursos próprios para tanto. Assim se determinam os requisitos de recursos, relações de custo e benefícios que interferem em tal ponto.

### **3.1 Viabilidade Organizacional**

Enquanto sistema para cadastro de heróis, o Curitiba Hero Registry possibilita o atendimento à função veementemente quintessencial da Curitiba Hero Association: o auxílio da atuação dos usuários na profissão de super-herói. Isso se dá por meio do cadastro de heróis e vilões no sistema.

### **3.2 Viabilidade Econômica**

Enquanto organização sem fins lucrativos, os custos do sistema, relativos a manter o servidor operando e à contratação de administradores, devem ser mantidos por doações. As doações serão efetivadas diretamente com a gerência da CHÁ.

### **3.3 Viabilidade Técnica**

Visto a simplicidade do equipamento necessário para o desenvolvimento do sistema, não há nenhum integrante o qual não possui acesso ao hardware e software necessários. As tecnologias utilizadas no sistema não foram amplamente utilizadas por nenhum membro do grupo, porém, alguns membros possuem conhecimentos limitados quando concerne Python, SQL. Os membros com experiência em python são: Bernardo, Daniel e David. SQL é dominado apenas por Daniel. O framework Flask não é dominado por nenhum membro previamente à implementação do código, no entanto, por caracterizar-se como um modelo intuitivo, mostra-se viável implementar o projeto.

### **3.4 Viabilidade Operacional**

Conforme o regulamento de controle de super-heróis em colaboração com o sindicato dos trabalhadores da justiça, os direitos de dados que concernem à organização atuam em compatibilidade com o que é previsto em ata. Portanto, é viável descrever o sistema como coerente às normas éticas em prol do bom uso de *software*.

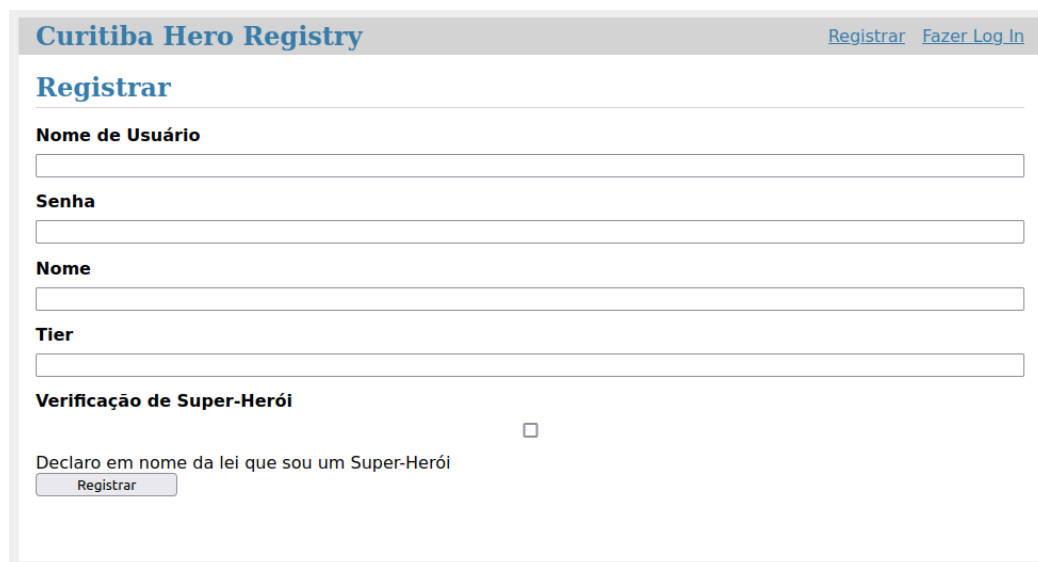
### **3.5 Recursos a serem utilizados**

Para o projeto, será contratada uma pequena equipe de desenvolvedores, no caso 4 engenheiros de software. Será providenciado um auxílio monetário, no valor de 25 reais por hora, e computadores para desenvolvimento do sistema, aos membros contratados. Espera-se 40 horas de trabalho para cada membro da equipe. Portanto, será gasto até um limite de 4000 reais em salário e 10000 em computadores e hardware.

## 4 RESULTADOS

### 4.1 Conteúdo dos Resultados

Enquanto site, o Curitiba Hero Registry atualmente opera em servidor local, possibilitando o registro, o login, a edição de cadastro, a inclusão de vilões no sistema, e a visualização de heróis e vilões cadastrados (em abas distintas). O site opera com conexão a um banco de dados implementado pelo grupo.



Curitiba Hero Registry [Registrar](#) [Fazer Log In](#)

### Registrar

**Nome de Usuário**

**Senha**

**Nome**

**Tier**

**Verificação de Super-Herói** ☐

Declaro em nome da lei que sou um Super-Herói

**Figura 1.** Página de registro de Super-Heróis.

### 4.2 Modelagem

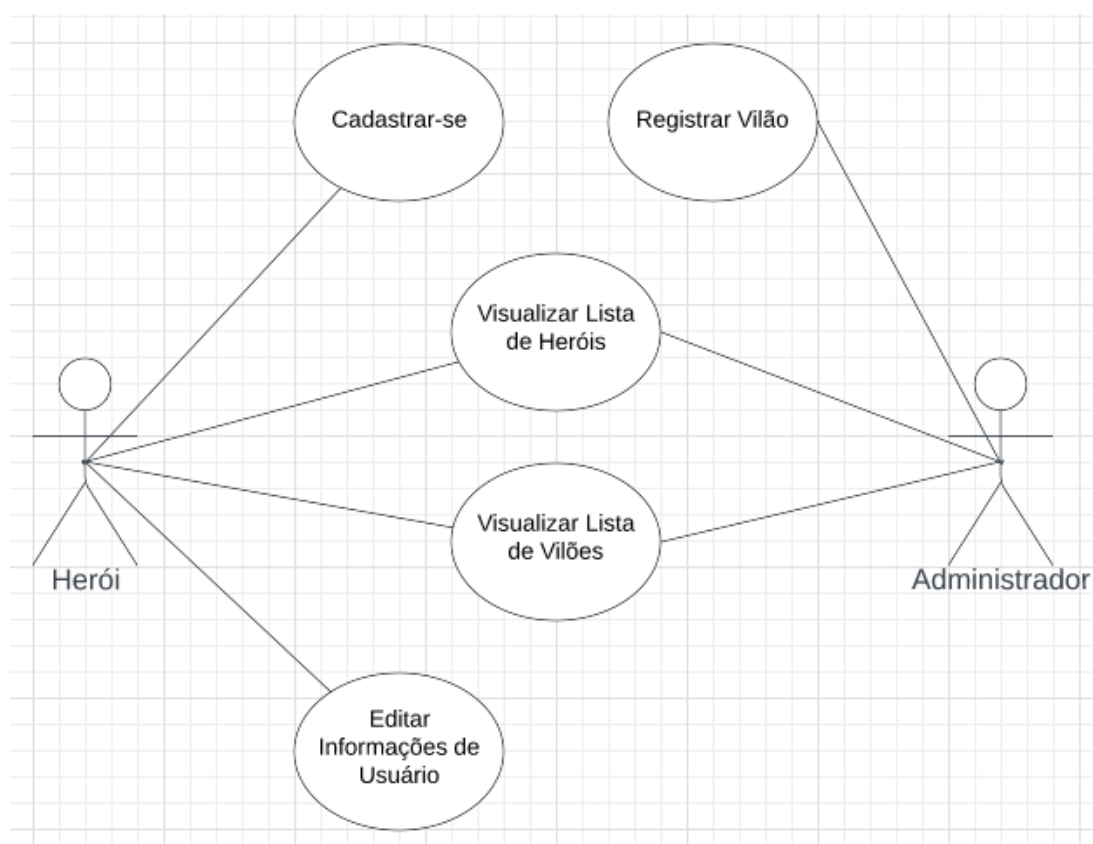
A partir dos requisitos funcionais levantados, tornou-se possível compilar casos de uso referentes ao programa.

**Tabela 2.** Casos de uso identificados.

Código	Descrição
Visualizar lista de Heróis - UC001	Visualizar dados públicos de heróis cadastrados no sistema, em formato

	de lista.
Visualizar lista de Vilões - UC002	Visualizar vilões cadastrados no sistema, em formato de lista.
Cadastrar-se - UC003	Registrar-se no sistema na condição de herói, fornecendo os dados necessários.
Registrar um vilão - UC004	Enquanto admin, adicionar um vilão e suas respectivas informações à database.
Editar informações de usuário - UC005	Editar informações pessoais cadastradas na database.

E desse modo, elaborou-se um Diagrama de Casos de Uso, conforme modelo apresentado a seguir.



**Figura 2.** Diagrama de Casos de Uso até o presente momento de desenvolvimento do projeto.

Elaborou-se então uma descrição detalhada referente a cada caso de uso identificado.

**Tabela 3.** Detalhamento caso de uso Visualizar lista de Heróis - UC001.

Nome	UC001: Visualizar lista de Heróis.
Atores	Ator Principal: Herói. Ator Principal: Administrador.
Descrição	Caso de uso executado quando o ator deseja visualizar a lista de Heróis cadastrados no sistema e os dados referentes a cada Herói.
Pré-condições	<ul style="list-style-type: none"><li>- O ator deve estar logado.</li><li>- A lista de Heróis deve ser carregada da Database</li></ul>
Pós-condições	Deve-se exibir informações não confidenciais de usuários cadastrados na condição de Herói.
Fluxo Básico	
Ações dos atores	Ações do sistema
1 - Ator seleciona a opção Lista de Heróis no menu principal.	
	2 - O sistema redireciona o ator para a página de exibição da lista de Heróis.
	3 - O sistema percorre a lista de pessoas cadastradas na database e exibe informações referentes àquelas cadastradas na condição de Herói.
Regras de Negócio	
[RN001] O sistema deve conseguir acessar a database e carregar pessoas cadastradas.	
Fluxo alternativo 1	

<b>Ações dos atores</b>	<b>Ações do sistema</b>
1 - O usuário seleciona a opção de retornar para a página anterior.	
	2 - O sistema redireciona o usuário para o menu principal.

**Tabela 4.** Detalhamento caso de uso Visualizar lista de Vilões - UC002.

Nome	UC002: Visualizar lista de Vilões		
Atores	Ator Principal: Herói. Ator Principal: Administrador.		
Descrição	Caso de uso executado quando o ator deseja visualizar a lista de vilões cadastrados no sistema e os dados referentes a cada Vilão.		
Pré-condições	<ul style="list-style-type: none"><li>- O ator deve estar logado.</li><li>- A lista de Vilões deve ser carregada da Database</li></ul>		
Pós-condições	Deve-se exibir informações de usuários cadastrados na condição de Vilão.		
Fluxo Básico			
Ações dos atores		Ações do sistema	
1 - Ator seleciona a opção Lista de Vilões no menu principal.			
		2 - O sistema redireciona o ator para a página de exibição da lista de Vilões.	
		3 - O sistema percorre a lista de pessoas cadastradas na database e exibe informações referentes àquelas cadastradas na condição de Vilão.	
Regras de Negócio			
[RN002] O sistema deve conseguir acessar a database e carregar pessoas cadastradas.			
Fluxo alternativo 1			
Ações dos atores		Ações do sistema	
1 - O usuário seleciona a opção de retornar para a página anterior.			

	2 - O sistema redireciona o usuário para o menu principal.
--	--



Tabela 5. Cadastrar-se - UC003.

<b>Nome</b>	UC003: Cadastrar-se
<b>Atores</b>	Ator Principal: Herói.
<b>Descrição</b>	Caso de uso executado quando o ator deseja criar um usuário para adquirir acesso às partes privadas do sistema
<b>Pré-condições</b>	<ul style="list-style-type: none"> <li>- O ator não pode estar logado</li> <li>- O ator precisa ser um herói</li> </ul>
<b>Pós-condições</b>	O ator deve estar logado no sistema com suas informações já registradas
<b>Fluxo Básico</b>	
<b>Ações dos atores</b>	<b>Ações do sistema</b>
1 - Ator seleciona a opção Registrar-se no canto superior direito.	
	2 - O sistema redireciona o autor para a página da criação de usuário.
3 - O ator preenche os campos apresentados com suas informações.	
4 - O ator pressiona o botão registrar na parte inferior da tela	
	5 - O sistema registra o usuário no sistema e redireciona o autor para a página principal como um usuário logado.
<b>Regras de Negócio</b>	
[RN003] O sistema deve conseguir acessar a database e adicionar novos dados. [RN004] O sistema deve ser capaz de detectar registros duplicados.	
<b>Fluxo alternativo 1</b>	
<b>Ações dos atores</b>	<b>Ações do sistema</b>

1 - O ator tenta criar uma conta já existente.	
	2 - O sistema informa o usuário sobre a existência de um usuário duplicado
	3 - O sistema solicita que o ator corrija as informações aprensetadas.
<b>Fluxo alternativo 1</b>	
1 - O ator seleciona o menu principal.	
	2 - O sistema redireciona o usuário para o menu principal.
<b>Fluxo alternativo 2</b>	
1 - O ator seleciona a opção fazer login	
	2 - O sistema redireciona o autor para a página de login.
<b>Fluxo alternativo 3</b>	
1 - O autor não clica em “confirmar que sou um super herói”	
	2 - O sistema impede o usuário de registrar-se.

Tabela 6. Detalhamento de caso de uso Registrar um Vilão - UC004.

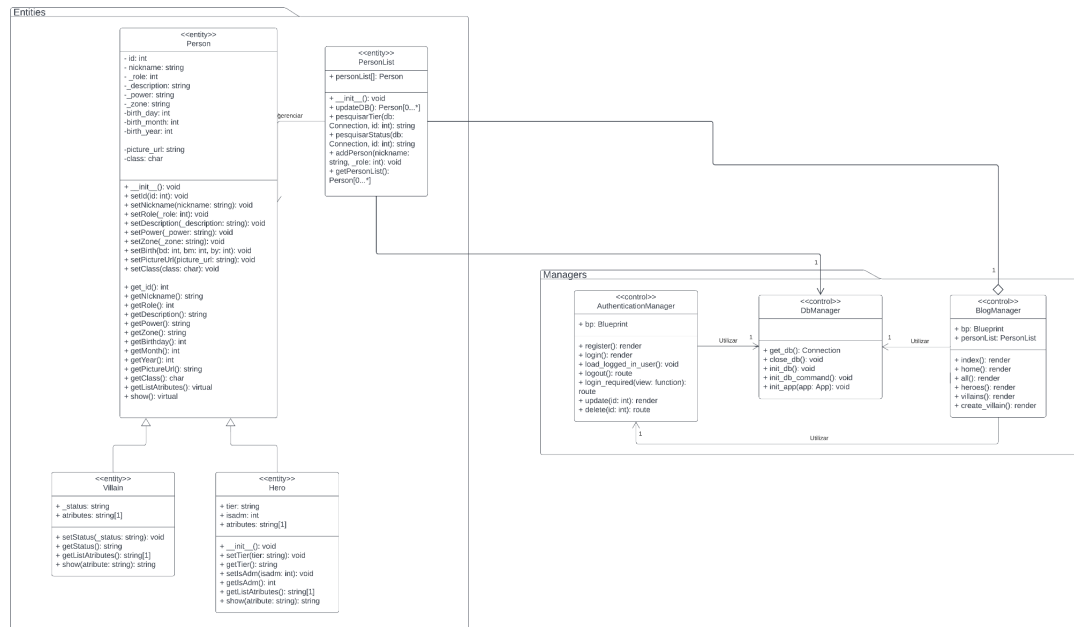
<b>Nome</b>	UC004: Registrar um vilão
<b>Atores</b>	Ator Principal: Administrador.
<b>Descrição</b>	Caso de uso executado quando o ator deseja inserir os dados de um novo vilão no sistema.
<b>Pré-condições</b>	<ul style="list-style-type: none"> <li>- O ator deve estar logado como um administrador.</li> <li>-</li> </ul>
<b>Pós-condições</b>	O vilão registrado deve estar presente na <i>database</i> de vilões.
<b>Fluxo Básico</b>	
<b>Ações dos atores</b>	<b>Ações do sistema</b>
1 - Ator seleciona a opção Registrar um vilão.	
	2 - O sistema redireciona o ator para a página de registro de vilões.
3 - O ator preenche os campos apresentados com os dados correspondentes do vilão.	
4 - O ator confirma os dados inseridos.	
	5 - O sistema insere o novo vilão na <i>database</i> .
	6 - O sistema redireciona o autor para a lista de vilões
<b>Regras de Negócio</b>	
[RN005] O sistema deve conseguir acessar a <i>database</i> e carregar vilões cadastrados. [RN006] O sistema deve ser capaz de detectar a inserção de um vilão já existente.	
<b>Fluxo alternativo 1</b>	
<b>Ações dos atores</b>	<b>Ações do sistema</b>

1 - O usuário seleciona a opção de retornar para a página anterior.	
	2 - O sistema redireciona o usuário para o menu principal.
<b>Fluxo alternativo 2</b>	
1 - O ator insere dados já existentes.	
	2 - O sistema avisa o usuário sobre a duplicação dos dados e impede o registro.
<b>Fluxo alternativo 3</b>	
1 - O autor seleciona a opção fazer logout.	
	2 - O sistema remove a autenticação do usuário.
	3 - O sistema redireciona o usuário para o menu principal

**Tabela 7.** Detalhamento de caso de uso Editar informações de usuário - UC005.

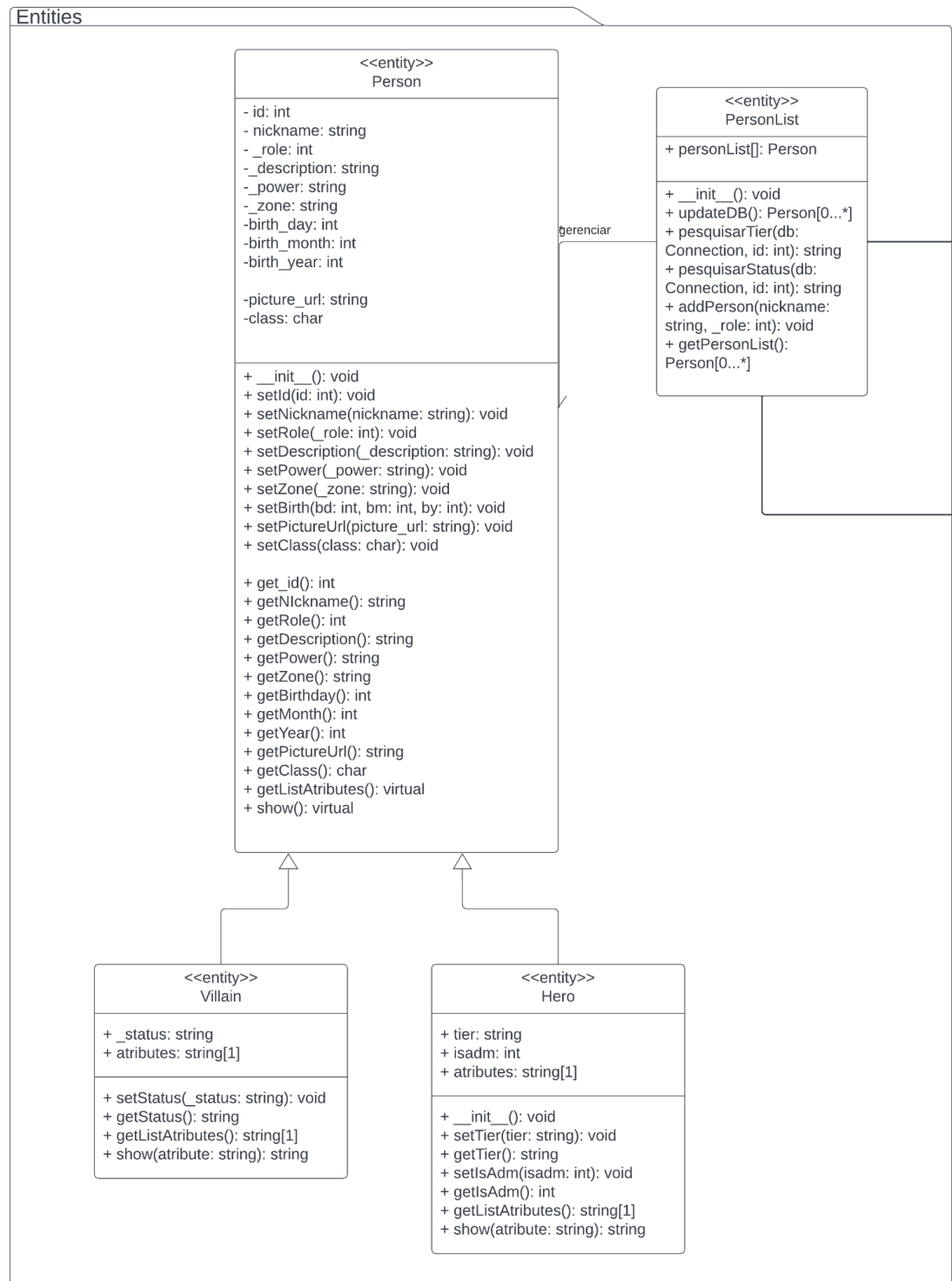
<b>Nome</b>	UC005: Editar informações de usuário
<b>Atores</b>	Ator Principal: Herói.
<b>Descrição</b>	Caso de uso executado quando o ator deseja editar seus respectivos dados cadastrados no sistema
<b>Pré-condições</b>	<ul style="list-style-type: none"> <li>- O ator deve estar logado.</li> <li>- Suas informações devem ser carregada da Database</li> </ul>
<b>Pós-condições</b>	Deve-se modificar os dados do usuário na database.
<b>Fluxo Básico</b>	
<b>Ações dos atores</b>	<b>Ações do sistema</b>
1 - Ator seleciona a opção Editar Registro no menu principal.	
	2 - O sistema redireciona o ator para a página de modificar cadastro.
3 - O ator preenche os campos disponibilizados com informações atualizadas e clica em salvar.	
<b>Regras de Negócio</b>	
[RN007] O sistema deve conseguir acessar a database e atualizar os dados informados.	
<b>Fluxo alternativo 1</b>	
<b>Ações dos atores</b>	<b>Ações do sistema</b>
1 - O usuário seleciona a opção delete e encerra seu perfil.	
	2 - O sistema elimina o cadastro do usuário da database.
	3 - O sistema redireciona o usuário para a página home

Tendo em vista a implementação do projeto, modelou-se um diagrama de classes.



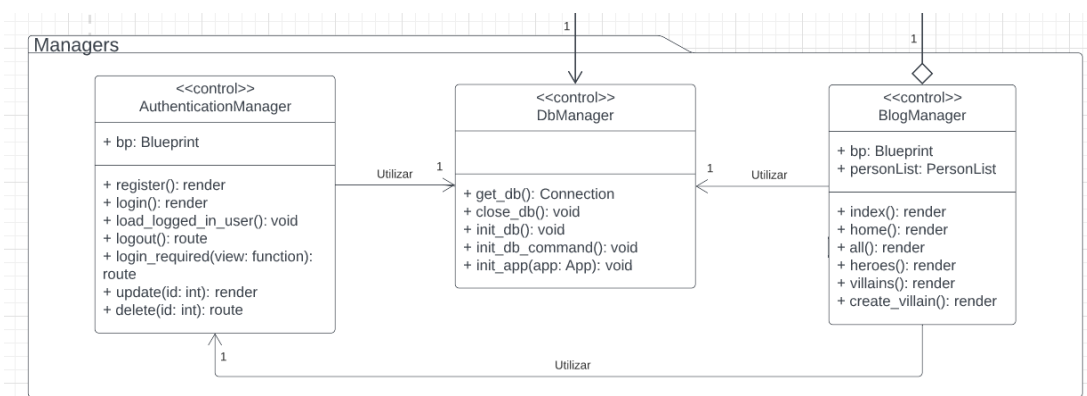
**Figura 3.** Vista diagrama de classes completo.

Esse modelo divide-se em dois espaços, Entities e Managers.



**Figura 4.** Espaço Entities no diagrama de classes.

Em que as classes em Entities concentram informações carregadas a partir da database e as classes em Managers gerenciam a navegação no site.



**Figura 5.** Espaço Managers no diagrama de classes.

A partir do diagrama de classes, pode-se então elaborar o dicionário de informações.

**Tabela 8.** Dicionário de Informações - Classe Hero.

Hero					
Atributo	Descrição	Tamanho	Tipo	Formato	Domínio
_role	Refere-se à qual subclasse a “Pessoa” pertence	1	Numérico	9{20}	Discreto 0 = Admin 1 = Herói 2 = Vilão
Nickname	Nome de usuário		Alfanumérico	x{100}	Contínuo
ID	Número de registro		Numérico		Contínuo
tier	Nível de força do indivíduo	1	Alfanumérico	A	Discreto
Atributos	Poderes dos indivíduos	200	Alfanumérico	X{200}	Contínuo

**Tabela 9.** Dicionário de Informações - Classe Hero.

Villain
---------



Atributo	Descrição	Tamanho	Tipo	Formato	Domínio
_role	Refere-se à qual subclasse a "Pessoa" pertence	1	Numérico	9{20}	Discreto 0 = Admin 1 = Herói 2 = Vilão
Nickname	Nome de usuário		Alfanumérico	x{100}	Contínuo
ID	Número de registro		Numérico		Contínuo
_status	Status de liberdade	5	Alfanumérico	XXXXX	Discreto
Attributes	Poderes dos indivíduos	200	Alfanumérico	X{200}	Contínuo

Tabela 10. Dicionário de Informações - Classe PersonList.

PersonList					
Atributo	Descrição	Tamanho	Tipo	Formato	Domínio
PersonList	Lista de pessoas		Person		Contínuo

Tabela 11. Dicionário de Informações - Classe AuthenticationManager.

AuthenticationManager					
Atributo	Descrição	Tamanho	Tipo	Formato	Domínio
Bp	Blueprint da página		Blueprint		Contínuo

Tabela 12. Dicionário de Informações - Classe BlogManager

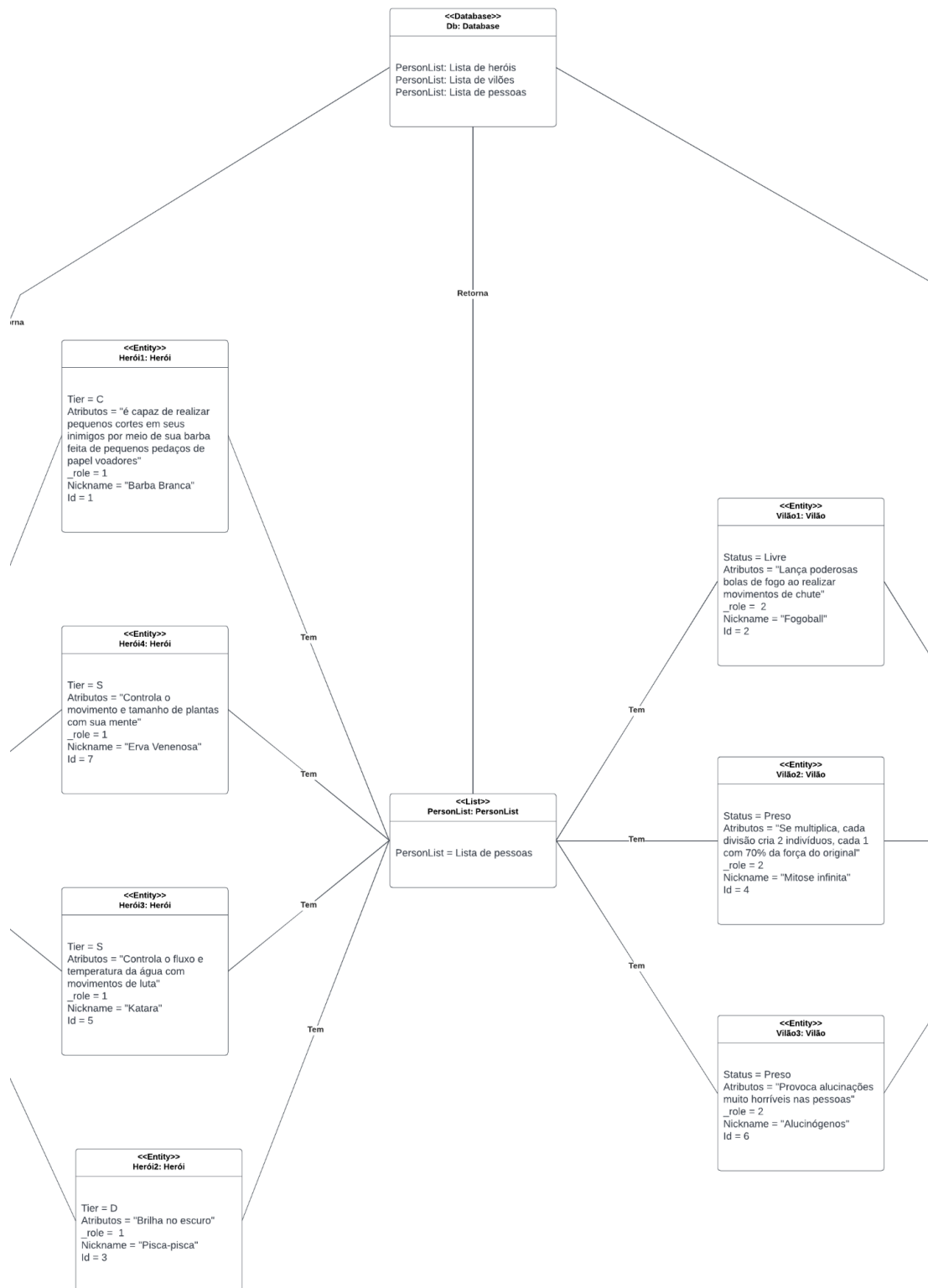
BlogManager					
Atributo	Descrição	Tamanho	Tipo	Formato	Domínio

PersonList	Lista de pessoas		Person		Contínuo
------------	------------------	--	--------	--	----------

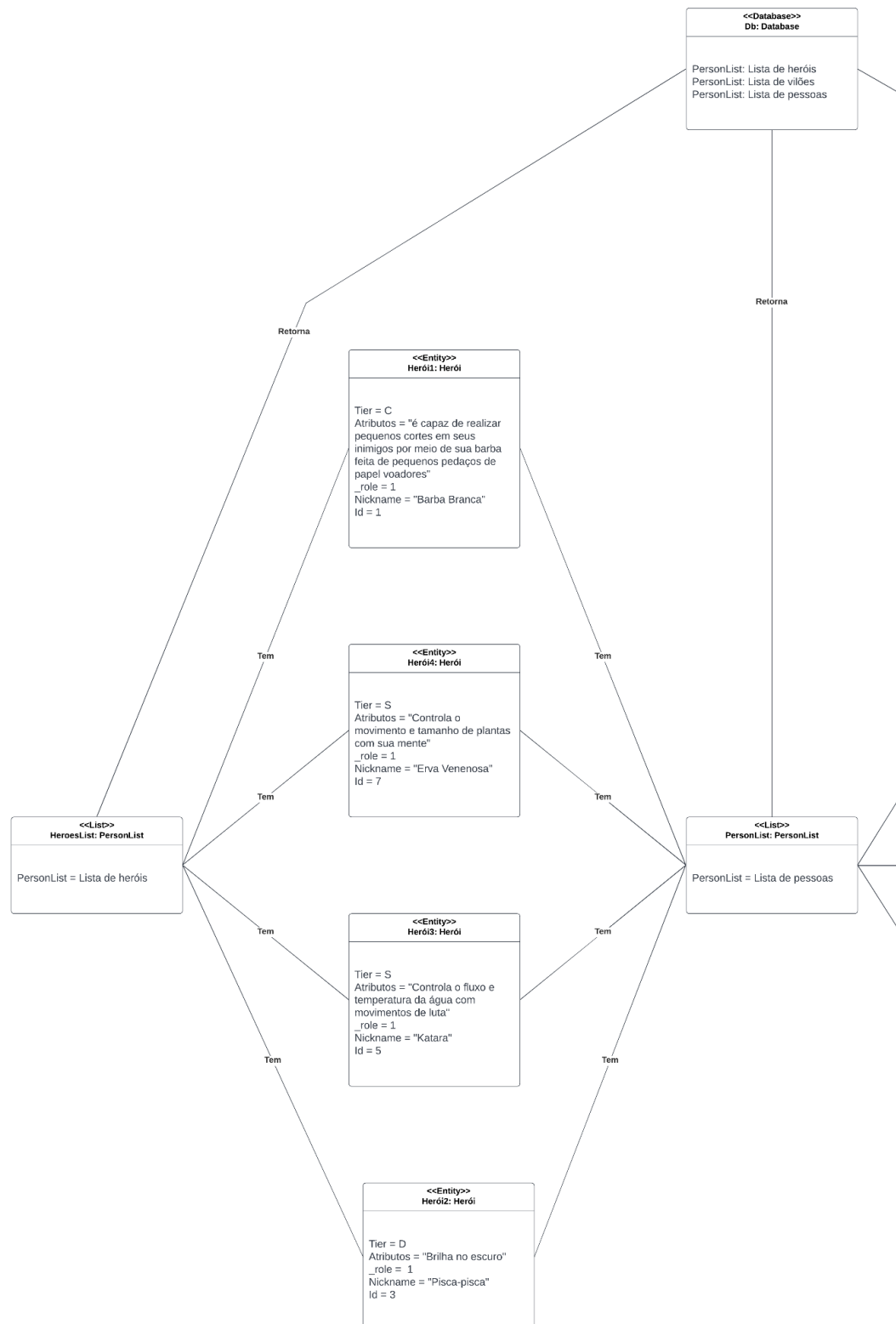
Tabela 13. Dicionário de Informações - Init.

Init					
Atributo	Descrição	Tamanho	Tipo	Formato	Domínio
app	Aplicativo o qual roda o website		Flask app		Contínuo

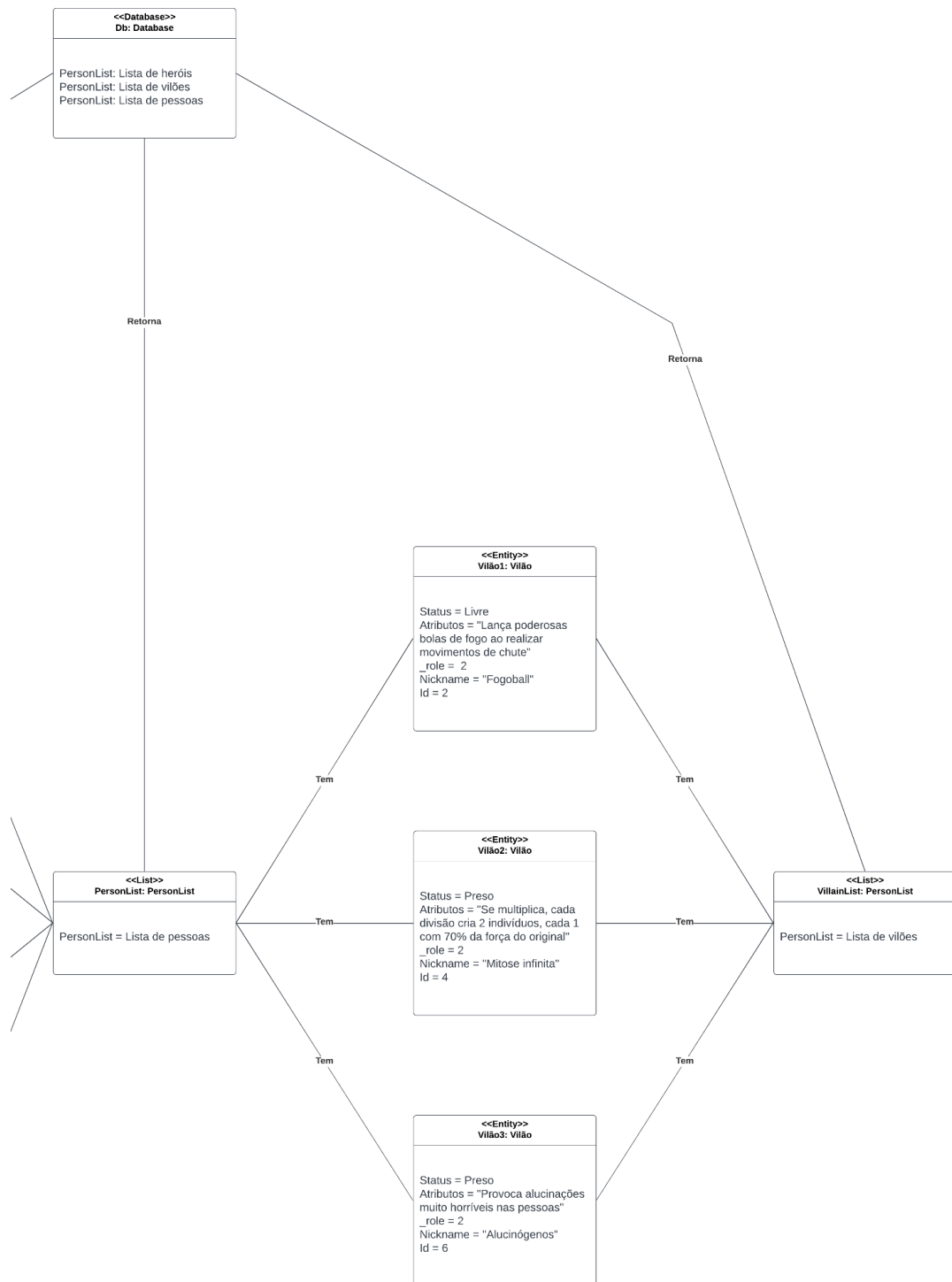
Já com os diagramas de caso de uso e o Uml em andamento foi desenvolvido o diagrama de objetos, este tem como objetivo demonstrar como objetos do sistema se relacionam de maneira concreta.



**Figura 6.** Relação de PersonList com outros objetos do sistema.



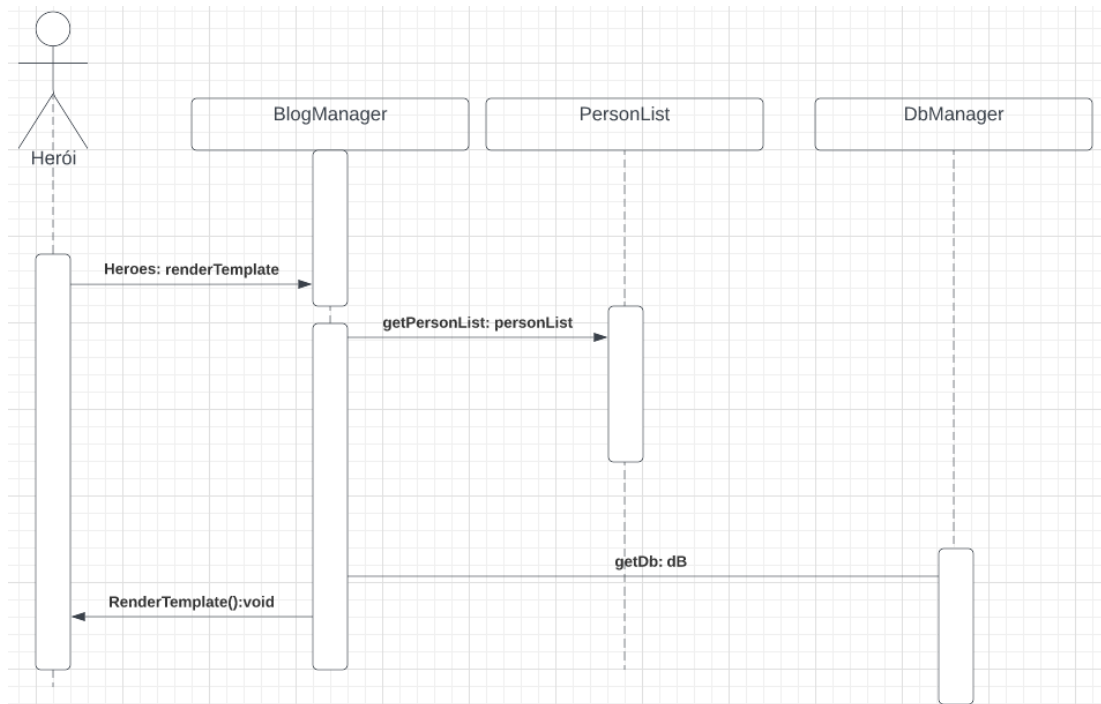
**Figura 7.** Relação de HeroesList com outros objetos do sistema.



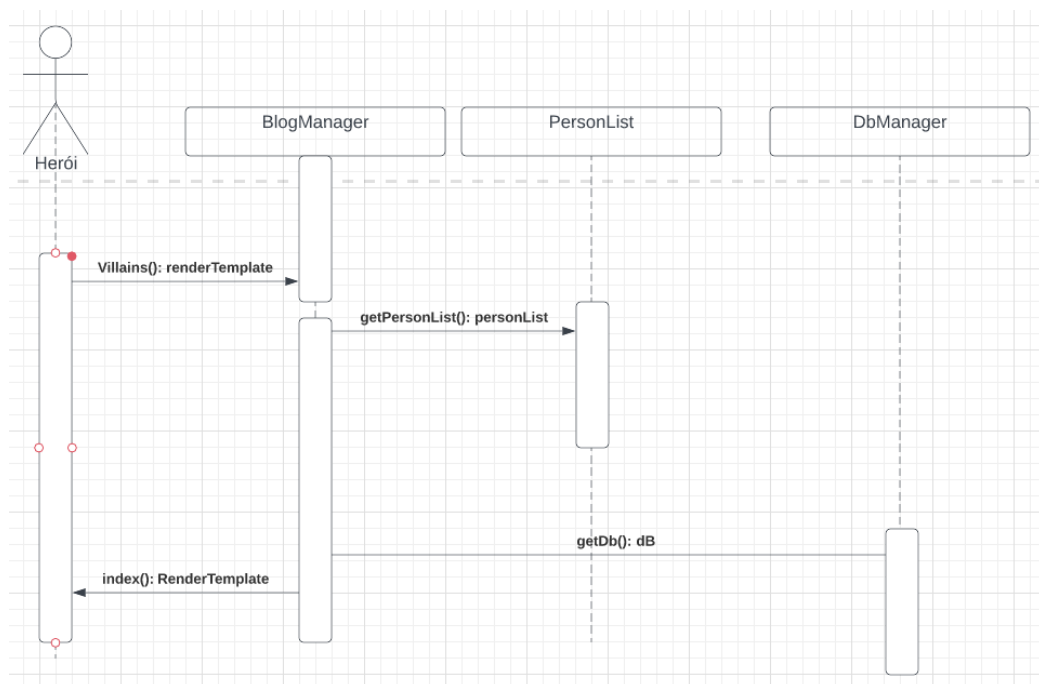
**Figura 8.** Relação de VillainList com outros objetos do sistema.

Para o melhor entendimento dos casos de uso, tornou-se necessário o desenvolvimento de um diagrama o qual demonstra as relações entre os

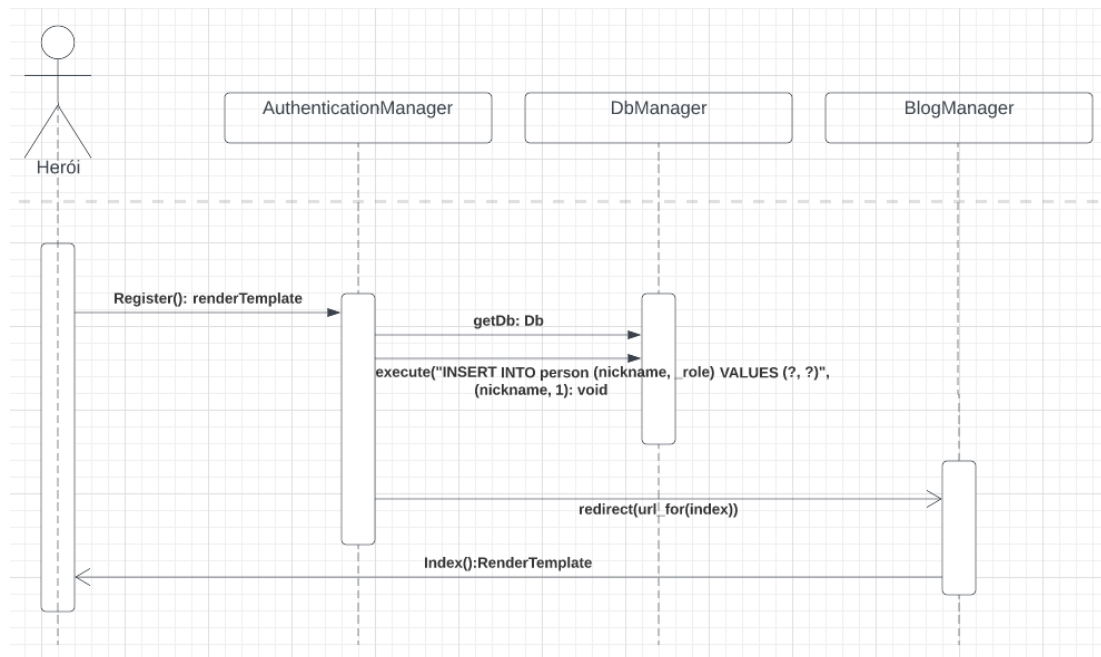
objetos e como eles são chamados durante o processo de utilização do sistema. A solução utilizada foi o diagrama de casos de uso mostrado a seguir:



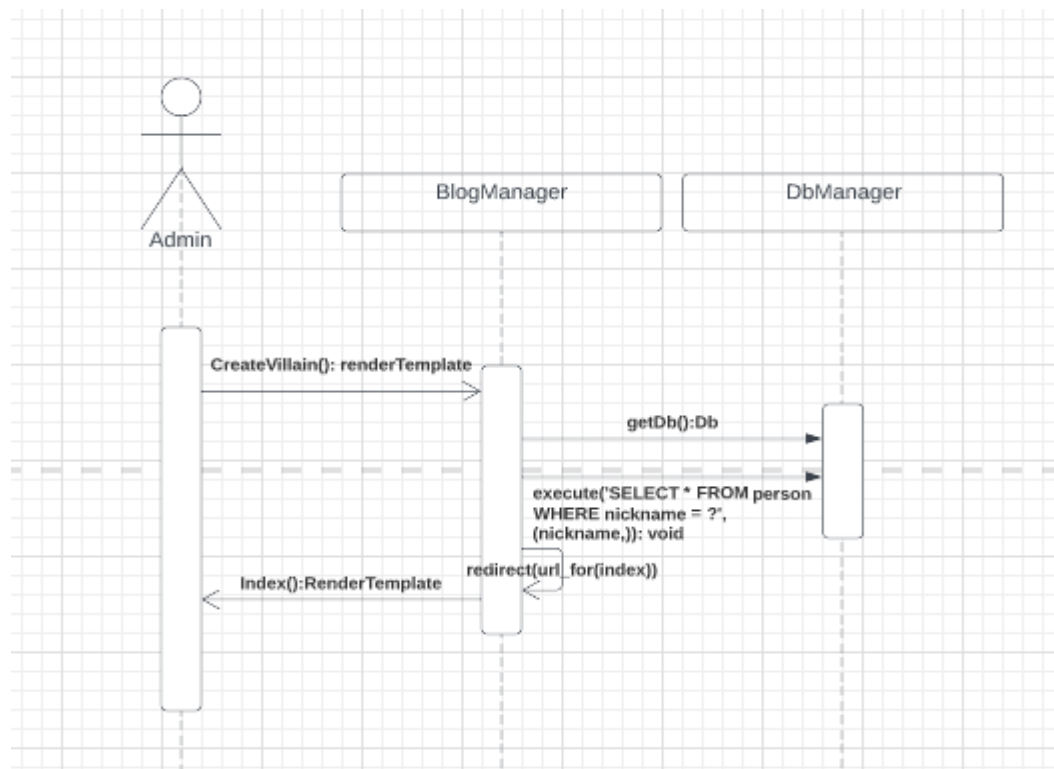
**Figura 9.** Diagrama de casos de uso para a visualização da lista de heróis.



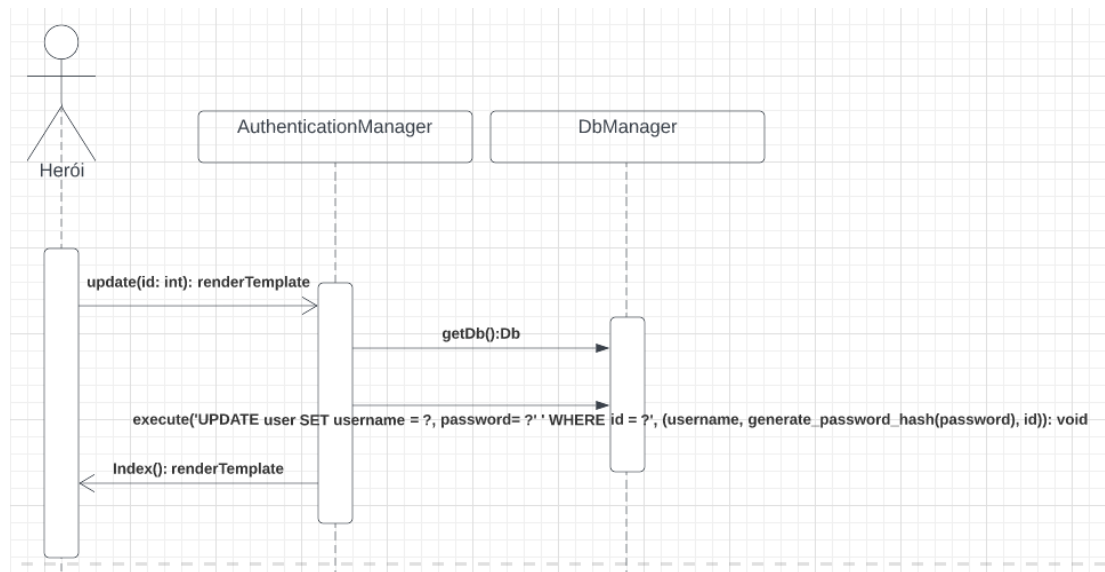
**Figura 10.** Diagrama de casos de uso para a visualização da lista de vilões.



**Figura 11.** Diagrama de casos de uso para a ação de registrar um novo usuário.

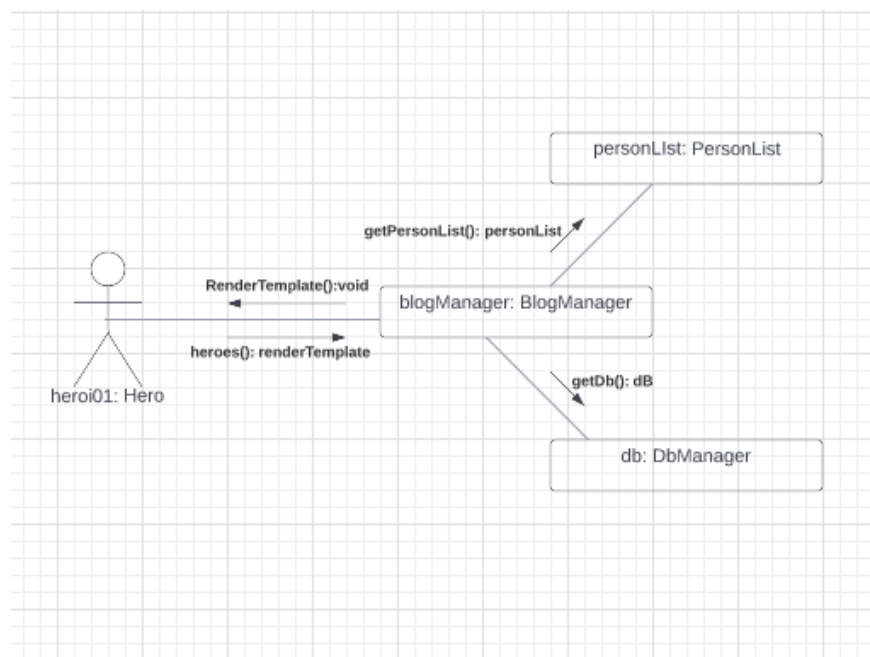


**Figura 12.** Diagrama de casos de uso para a ação de criar um novo vilão.



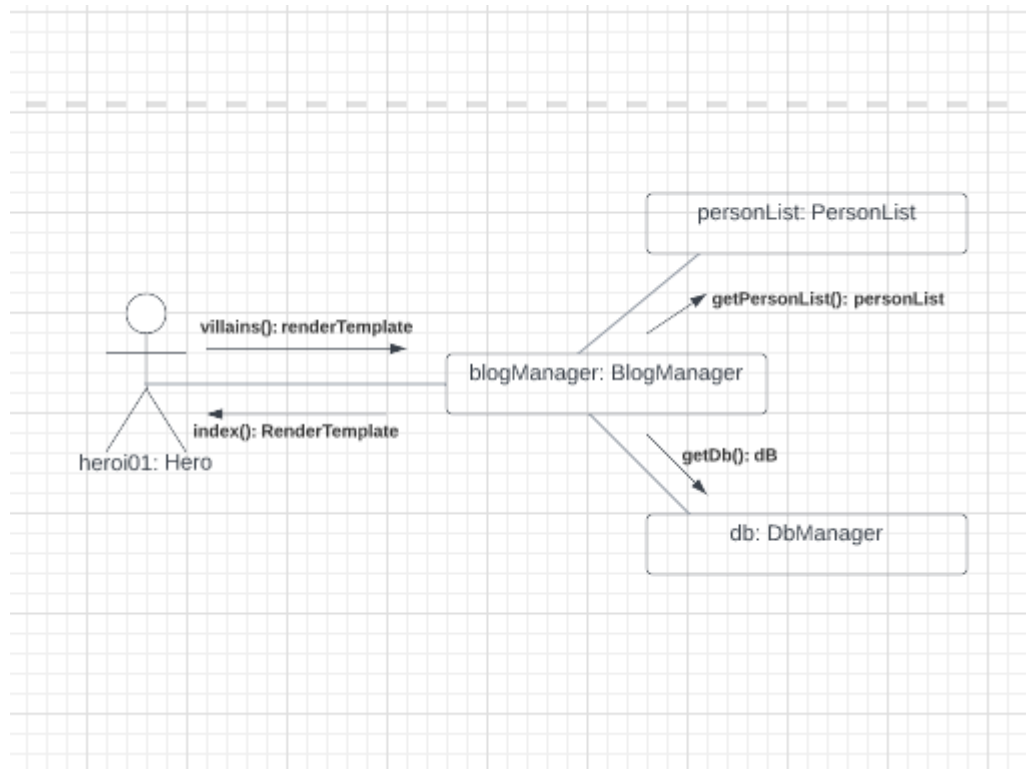
**Figura 13.** Diagrama de casos de uso para a atualização dos dados cadastrais de um herói.

Para cada diagrama de caso de uso elaborou-se, então, um respectivo diagrama de comunicação, de modo a representar as interações entre os objetos no sistema.

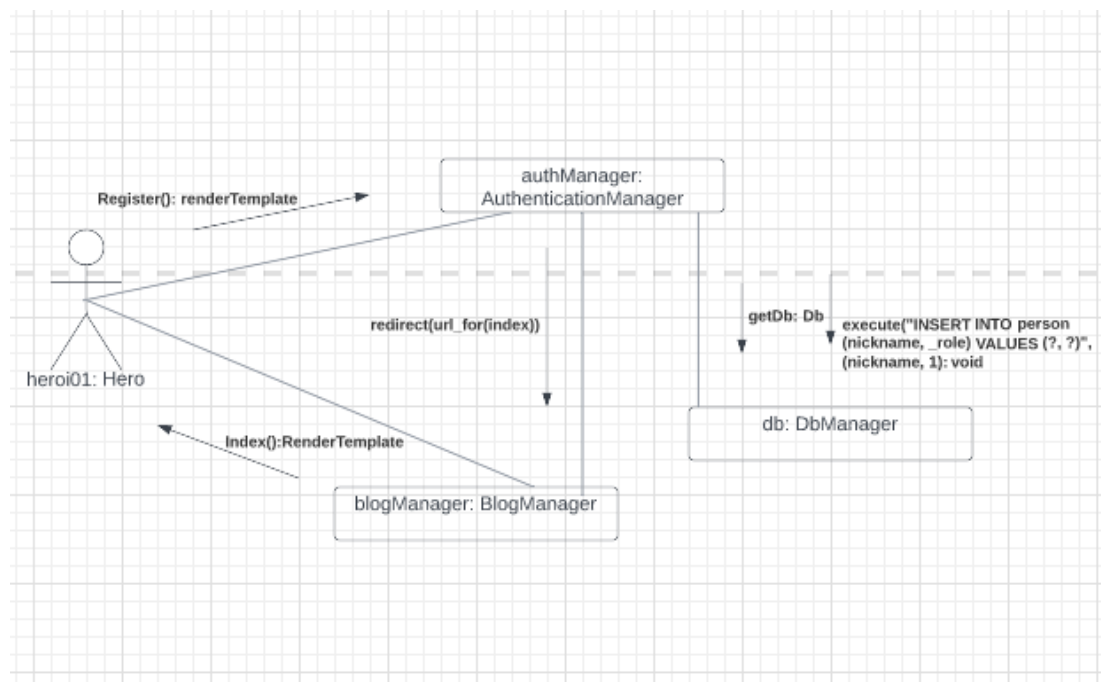


**Figura 14.** Diagrama de comunicação para a visualização da lista de heróis.

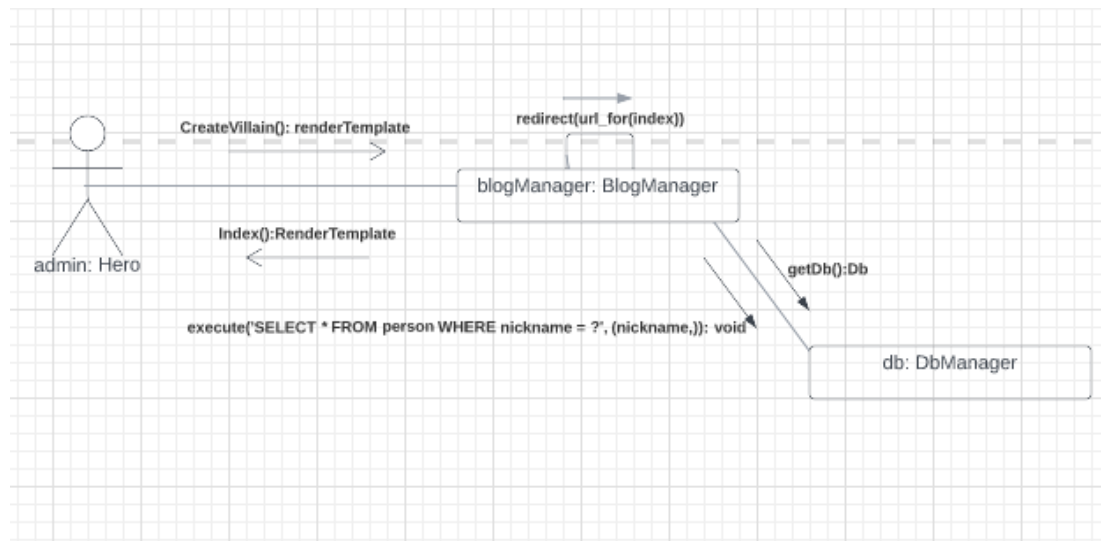




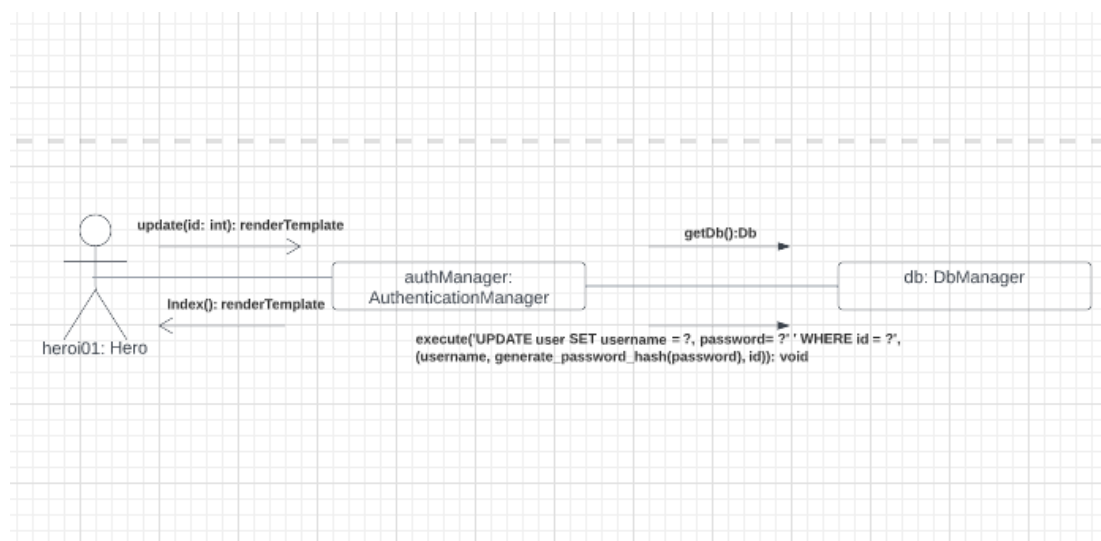
**Figura 15.** Diagrama de comunicação para a visualização da lista de vilões.



**Figura 16.** Diagrama de comunicação para a ação de registrar um novo usuário.

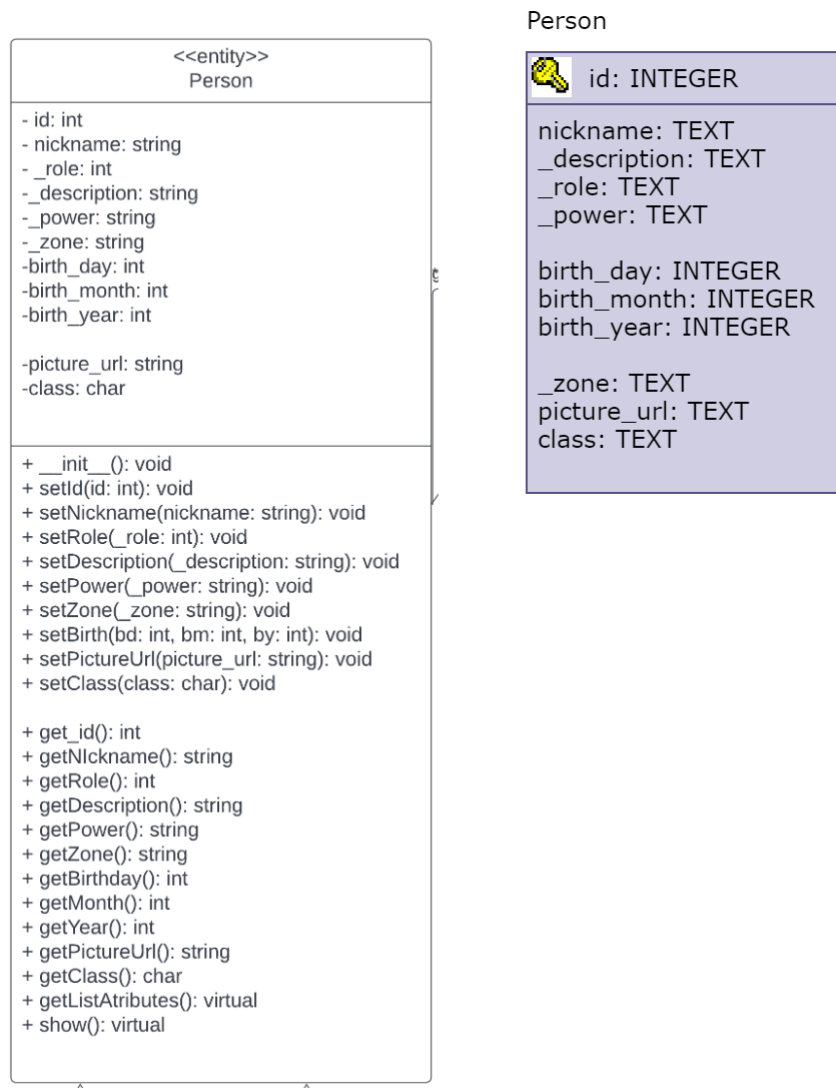


**Figura 17.** Diagrama de comunicação para a ação de criar um novo vilão.

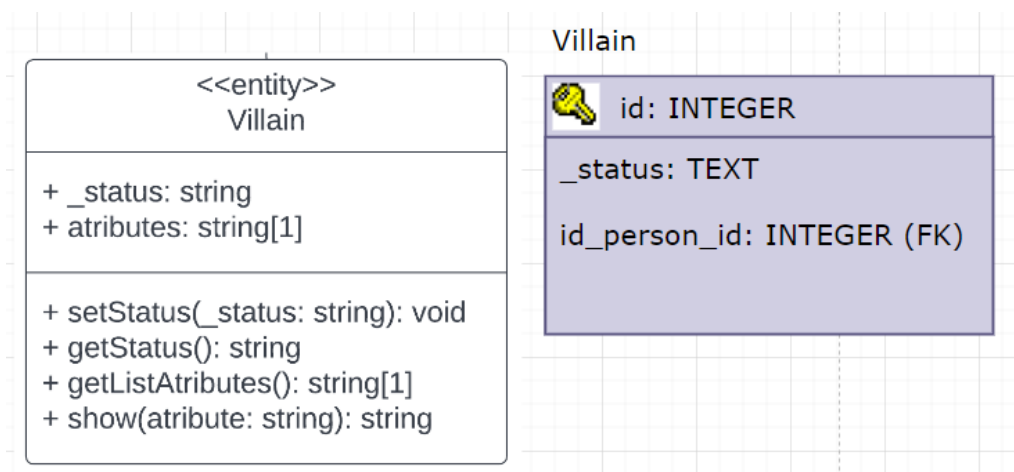


**Figura 18.** Diagrama de comunicação para a ação de criar um novo vilão.

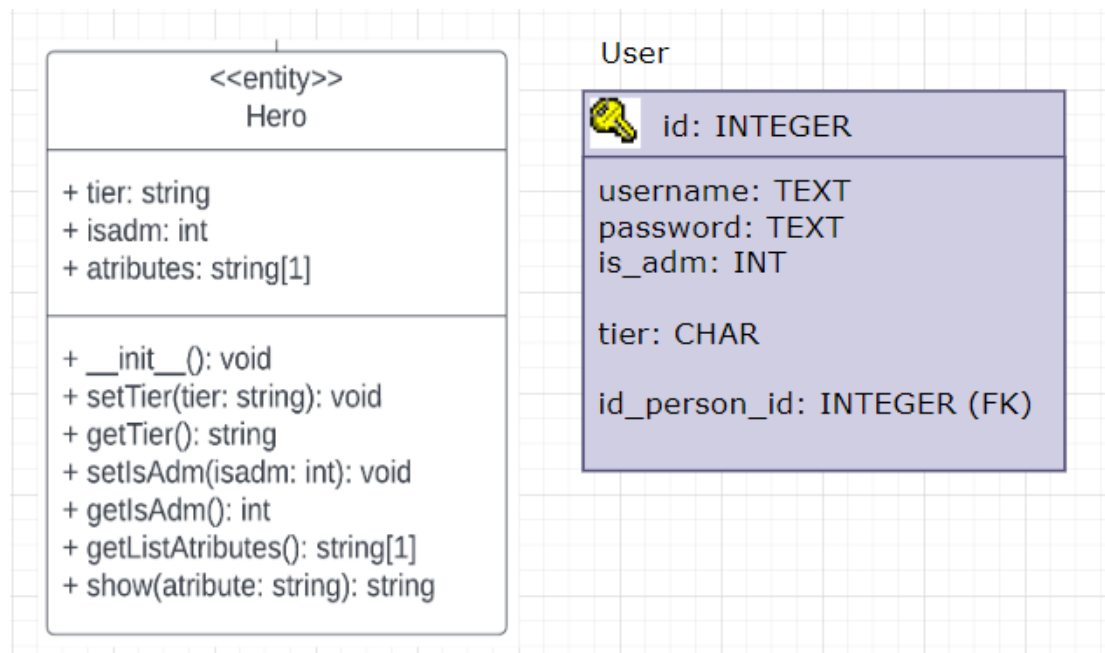
A fim de realizar o mapeamento do Diagrama de Classes UML para o Diagrama de Entidade-Relacionamento (DER), foi utilizado apenas o espaço Entities, tal como tinha sido mostrado na Figura 4, mas especificamente as classes que não formam listas, uma vez que a classe *PersonList* representa uma abstração das tabelas do banco de dados. Com esse processo, nas Figuras 19 a 21 são mostradas as conversões do design de cada classe UML em design DER.



**Figura 19.** Conversão da classe Person em UML para DER.



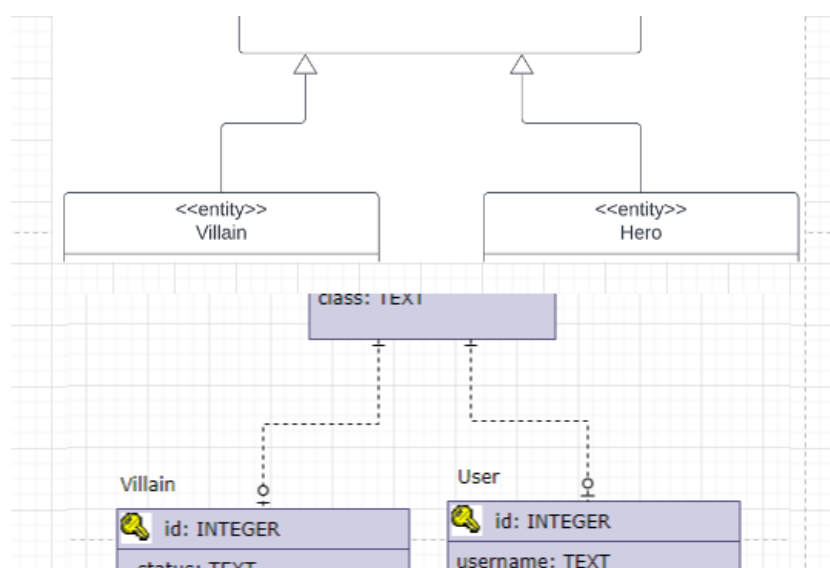
**Figura 20.** Conversão da classe Villain em UML para DER.



**Figura 21.** Conversão da classe Hero em UML para DER.

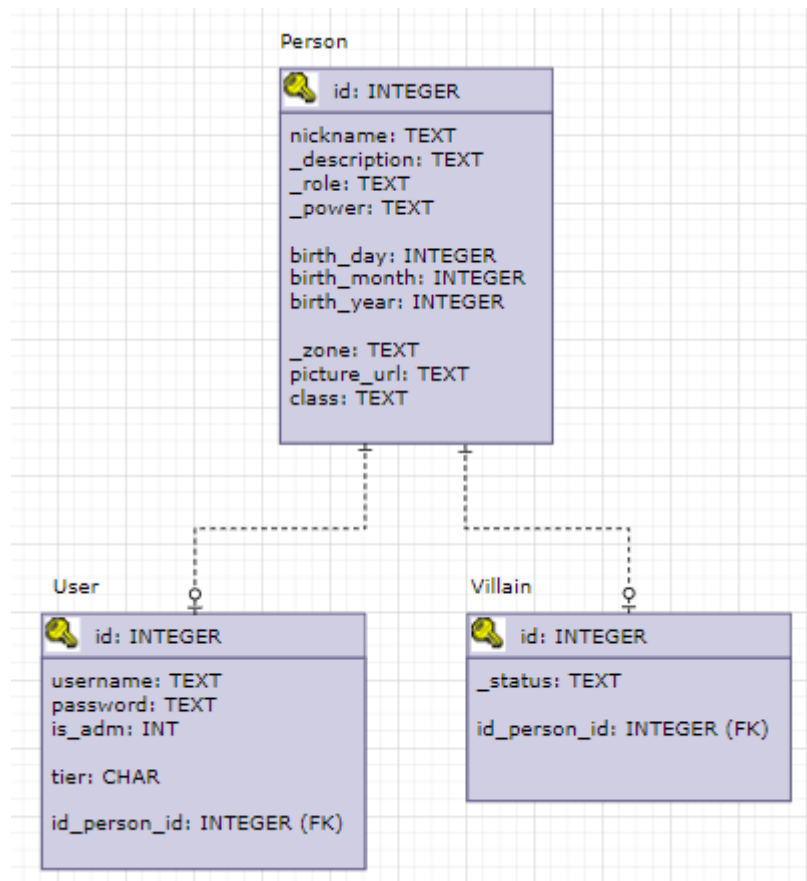
Como pôde-se notar, no Diagrama de Entidade-Relacionamento, os métodos são ocultados e alguns atributos foram adicionados, tais como o ID para cada entidade, estabelecidos como chave-primária, e os IDs da pessoa herdada, correspondendo-se à chave estrangeira.

Já na Figura 22 é visualizado o relacionamento de herança que as entidades possuem com Person, comparando o design entre os modelos.



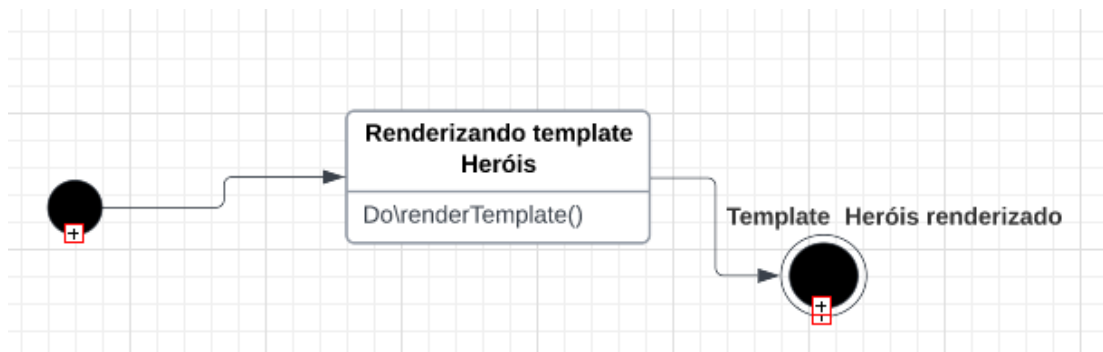
**Figura 22.** Conversão da herança em UML para DER.

Isso posto, gerou-se o diagrama completo das Entidades, tal como consta na Figura 23. Considera-se que o herói é um usuário justamente pela caracterização intrínseca do Framework desenvolvido, na qual é preciso um nome de usuário e senha.



**Figura 23.** Diagrama Entidade-Relacionamento completo.

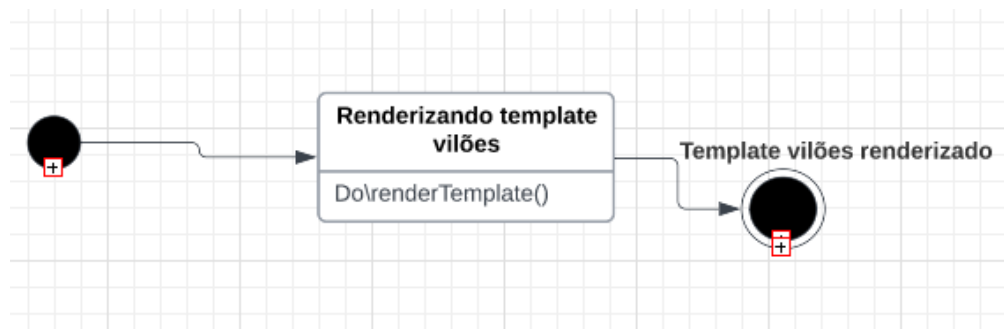
Para cada um dos diagramas de sequência foram selecionadas duas classes e desenvolvido seus respectivos diagramas de estado.



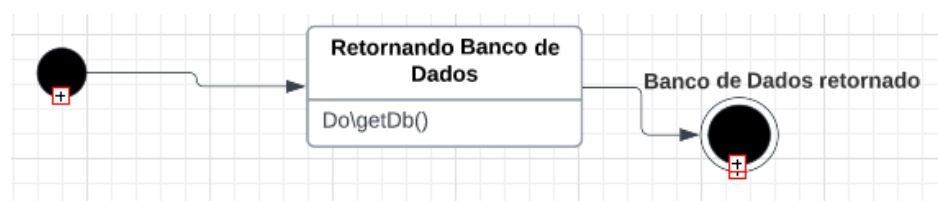
**Figura 24.** Diagrama de estado da classe BlogManager para caso de uso de visualização de lista de heróis.



**Figura 25.** Diagrama de estado da classe PersonList para caso de uso de visualização de lista de heróis.

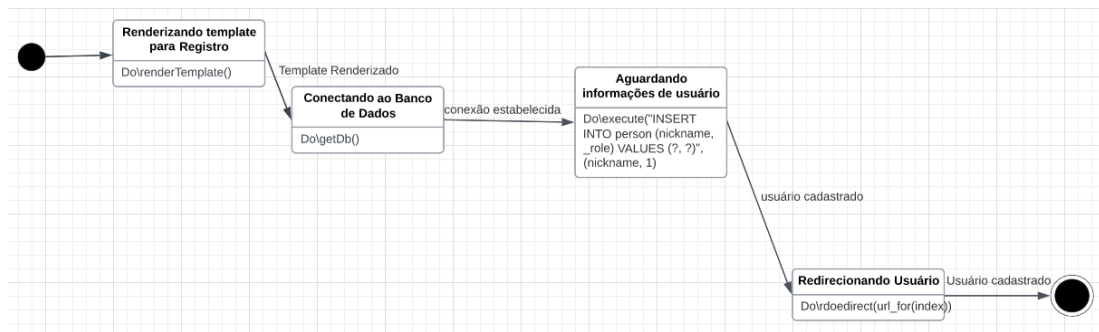


**Figura 26.1** diagrama de estado da classe BlogManager para caso de uso de visualização de lista de vilões.

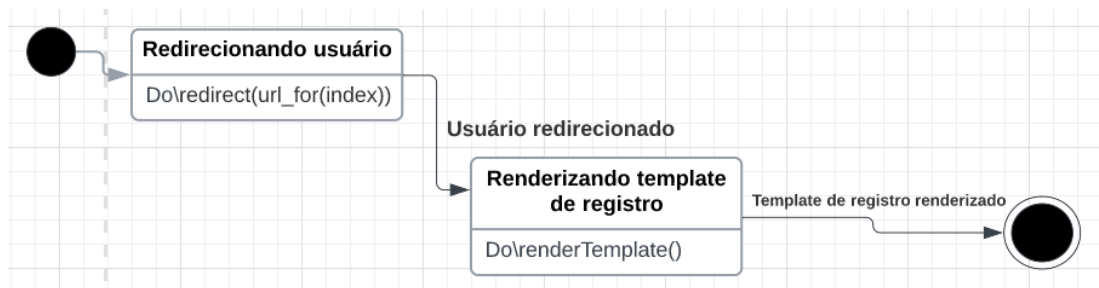


**Figura 26.2** diagrama de estado da classe DbManager

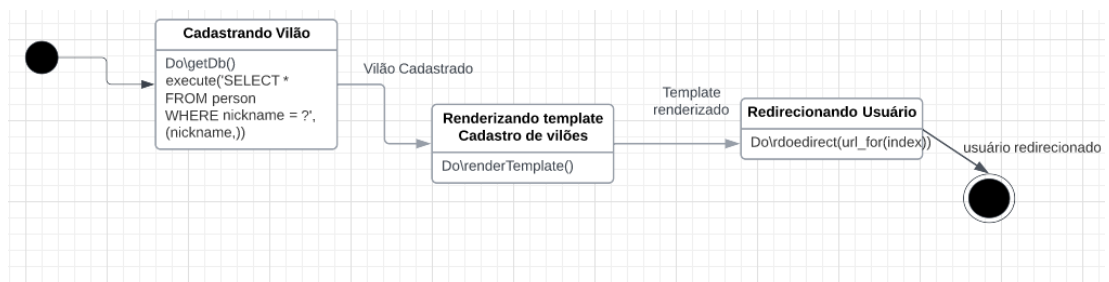
para caso de uso de visualização de lista de vilões.



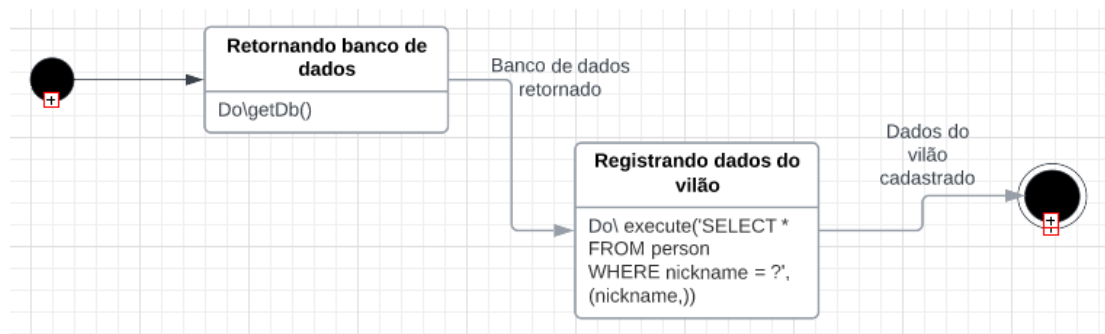
**Figura 27.** diagrama de estado da classe AuthenticationManager para caso de uso de registrar usuário.



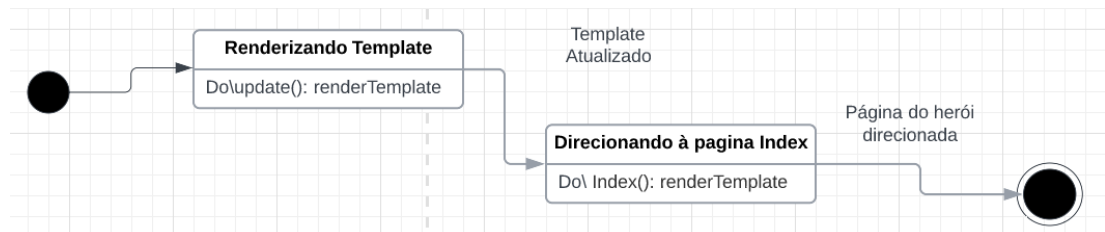
**Figura 28.** diagrama de estado da classe BlogManager para caso de uso de registrar usuário.



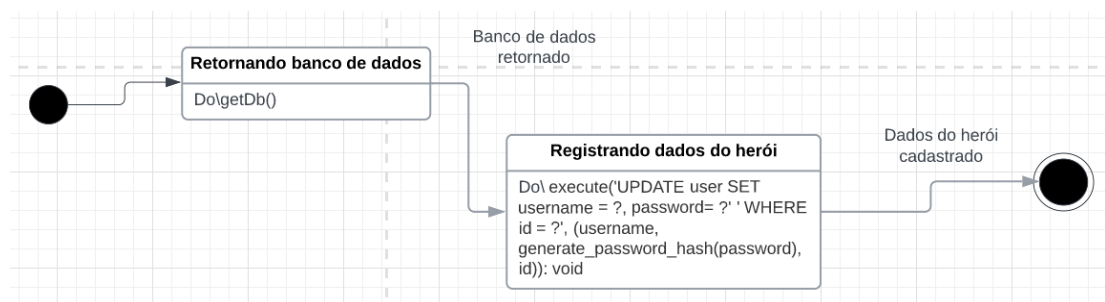
**Figura 29.** diagrama de estado da classe BlogManager para caso de uso de castrar vilão.



**Figura 30.** diagrama de estado da classe **DbManager** para caso de uso de castrar vilão.



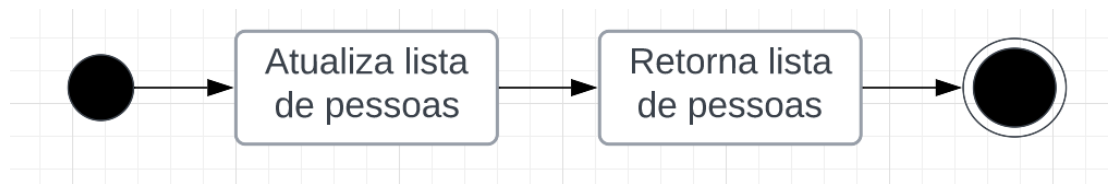
**Figura 31.** diagrama de estado da classe **AuthenticationManager** para caso de uso atualizar informações do usuário.



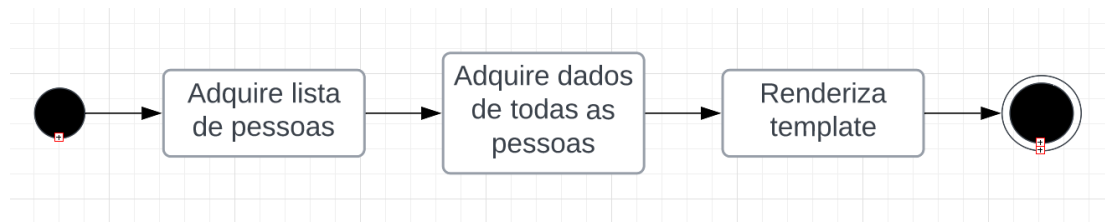
**Figura 32.** diagrama de estado da classe **DbManager** para caso de uso atualizar informações de usuário.

Em função dos estados de ação, foram selecionadas algumas atividades para desenvolver os diagramas de atividades.

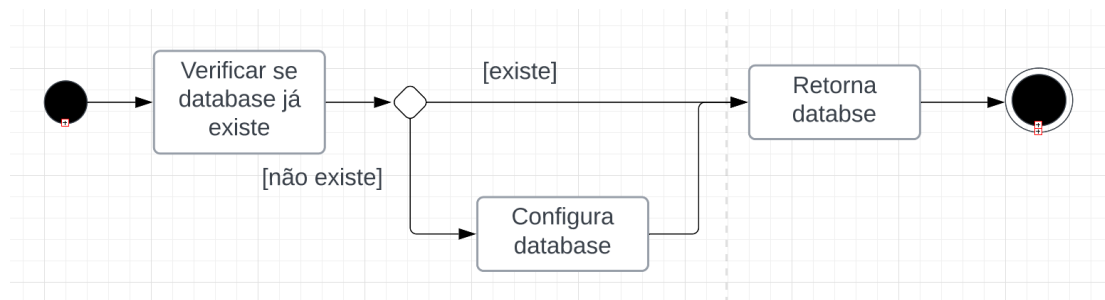




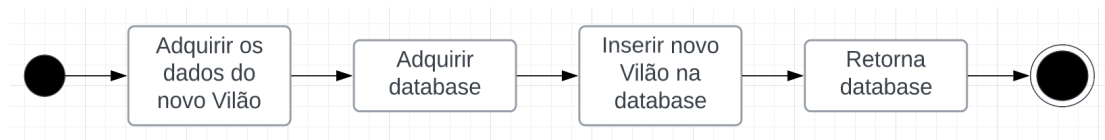
**Figura 33.** diagrama de atividades para a ação de atualizar os usuários na database.



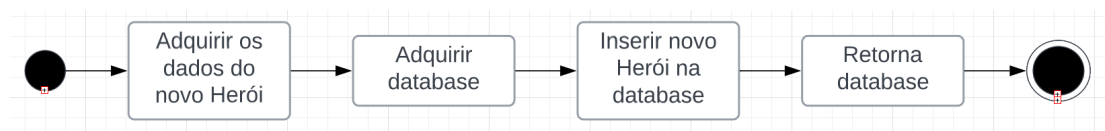
**Figura 34.** diagrama de atividades para a ação de renderizar a lista de pessoas.



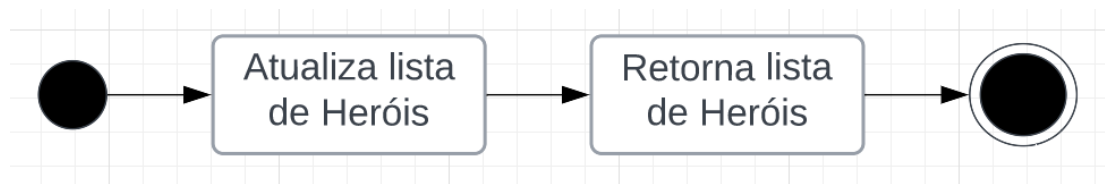
**Figura 35.** diagrama de atividades para a ação de retornar a database.



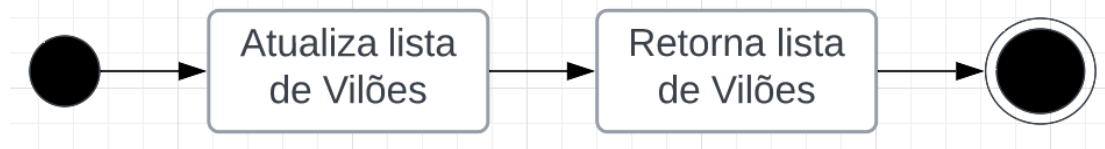
**Figura 36.** diagrama de atividades para a ação de inserir um Vilão na database.



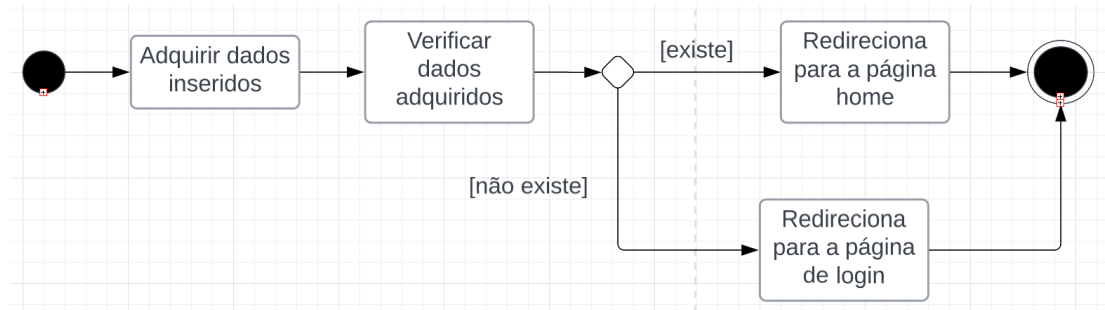
**Figura 37.** diagrama de atividades para a ação de inserir um Herói na database.



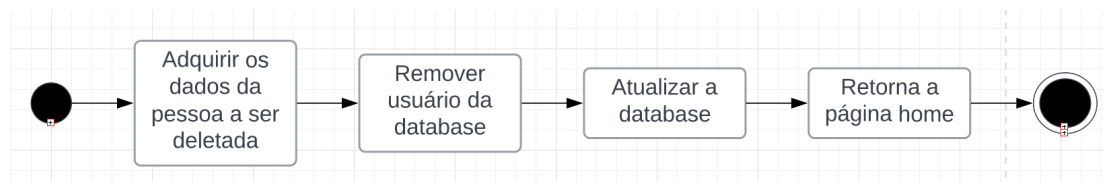
**Figura 38.** diagrama de atividades para a ação de atualizar a lista de Heróis.



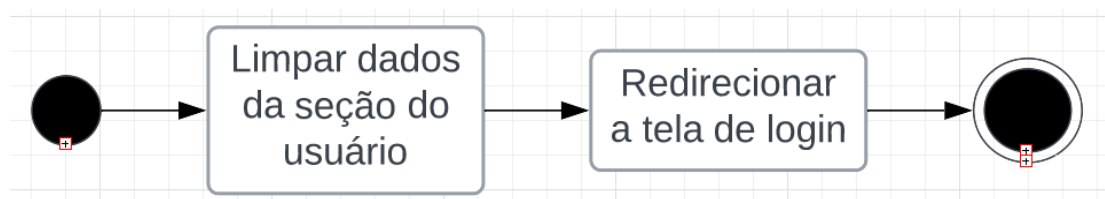
**Figura 39.** diagrama de atividades para a ação de atualizar a lista de Vilões.



**Figura 40.** diagrama de atividades para a ação de realizar login.



**Figura 41.** diagrama de atividades para a ação de remover pessoas da database.



**Figura 42.** diagrama de atividades para a ação de logout.

## 5 CONCLUSÕES

Enquanto ferramenta, o flask demonstra-se um framework poderoso para gerenciar a navegação de usuários na aplicação. No entanto, sob a perspectiva de comunicação de dados, em especial em relação ao banco de dados, a utilização desse framework resulta na abstração de diversos conceitos de gerenciamento de eventos, rompendo a linearidade do programa e dificultando o rastreamento dos acessos à database.

Tendo isso em vista, uma vez superadas as dificuldades e implementados métodos efetivos de consulta ao banco de dados, torna-se possível registrar e carregar informações de modo desacoplado, conforme os princípios de programação orientadas a objetos. Logo, obtém-se um software de fácil manutenção a longo prazo e de fácil atualização.

### 5.1 Trabalhos Futuros

Além de todos os benefícios que esse trabalho pôde trazer, tanto no quesito técnico quanto no intelectual, salienta-se que existe a possibilidade de transcender os limites acadêmicos estabelecidos pela disciplina de Análise e Projeto de Sistemas, elevando os conhecimentos a níveis externos. Isso diz respeito às bagagens de aprendizado adquiridas ao longo desse percurso, as quais são aplicáveis aos futuros projetos que podem ser criados pelos desenvolvedores do sistema.

Um exemplo claro disso é a implementação de redes sociais como ferramentas de conexões interpessoais, bem como o uso de Frameworks e modelos de trabalho Backend/Frontend. Com essas experiências, são estruturadas novas competências para o planejamento de Softwares de boa qualidade, respeitando cordialmente o ciclo de Engenharia em todos os seus aspectos. Portanto, futuros trabalhos serão encarados com muita confiança, de tal modo que o desenvolvimento se torne o mais dinâmico e prático possível.

## 6 REFERÊNCIAS BIBLIOGRÁFICAS

LAUDELINO CORDEIRO BASTOS. Professor Laudelino Cordeiro Bastos, 2022. Disponível em: <<http://laudelinobastos.com.br/>>. Acesso em 17 de set. de 2022.

PYTHON SOFTWARE FOUNDATION. Python 3.10.8 documentation. Disponível em: <<https://docs.python.org/3/>>. Acesso em 1 de out. de 2022.

PALLETS. Flask Documentation, 2022. Disponível em: <<https://flask.palletsprojects.com/en/2.2.x/>>. Acesso em 5 de out. de 2022.

GEEKSFORGEEKS. GeeksforGeeks, 2022. Disponível em: <<https://www.geeksforgeeks.org/>>. Acesso em 1 de out. de 2022.

JEAN MARCELO SIMÃO. Página de Internet do Prof. Simão, 2022. Disponível em: <<https://pessoal.dainf.ct.utfpr.edu.br/jeansimao/index.html>>. Acesso em 5 de out. de 2022.