**Sistema de gerenciamento para serpentário**

**Rodrigo Cabral Bock, Hylson Vescovi Netto (orientador)**

Instituto Federal Catarinense - Campus Blumenau (IFC)

Blumenau, SC - Brasil

rodrigocabralbock@gmail.com, hylson.vescovi@ifc.edu.br

# **1.Introdução**

Este documento tem como objetivo dissertar sobre o desenvolvimento de um sistema que pode ser usado para gerenciar um serpentário.

Um serpentário é um viveiro de serpentes que pode ser usado para fazer experiências com venenos e também preparação de soros antiofídicos (Porto Editora, 2021).

Como base para o desenvolvimento desde documento, está sendo usado o Serpentário Científico da Universidade Regional de Blumenau, localizado no Campus 5 cujo endereço é Rua Samuel Morse, 768 - Fortaleza Alta, Blumenau - SC, 89058-010.

## 1.1. Tema/Problema

O Serpentário/FURB armazena os dados somente em papel, com cópia única e suscetíveis a perda, assim, tais documentos precisam obrigatoriamente estar no campus para acesso de todos, portanto, o funcionário não pode levar os arquivos para casa com a finalidade de atualizar as informações das serpentes. Em caso de algum incidente em que o veterinário precise ver o histórico da serpente, o mesmo não estará disponível, podendo afetar o tratamento de saúde.

Por conta de os dados não estarem disponibilizados de forma digital, é necessário se deslocar até o serpentário para ter acesso aos documentos e passar as informações aos demais funcionários. Um outro ponto que deve ser levado em consideração é o acúmulo de papel.

## 1.2. Objetivos Propostos/Solução dos Problemas

A solução proposta é um site que terá todos os dados do Serpentário, servindo como um *backup* e solucionando o problema de deslocamento e possível perca. Além disso, com os dados publicados no site, qualquer um com acesso poderá visualizar as informações sobre as serpentes.

O diferencial é a criação de gráficos, que têm como foco acompanhar o desenvolvimento das serpentes. Isso é muito útil no caso da chegada de serpentes novas ou nascimento de filhotes, pois torna possível o acompanhamento do crescimento, peso e alimentação.

## 1.3 Escopo

O cadastro de serpentes conterá vários campos, todos de relevância para a identificação da serpente. Esses campos serão salvos no banco de dados, podendo ser utilizados em outras partes do sistema.

O cadastro de usuários é importante para manter um registro dos funcionários, sendo que o cadastro de funcionários salvará informações básicas destes, como nome, telefone, e-mail, usuário e senha. Logo, quando o usuário acessar o sistema, fará *login* com as informações de usuário e senha, e em seguida, poderá navegar livremente pelo sistema. Caso o funcionário faça a inserção de novos dados, seu nome aparecerá automaticamente junto com a informação adicionada.

Uma funcionalidade que o sistema terá é a geração de gráficos sobre o desenvolvimento das serpentes, sendo possível visualizar melhor as informações mais relevantes.

## 1.4 Viabilidade do projeto

O desenvolvimento do sistema será dividido em 2 partes, o *back-end* e o *front-end*. Para o desenvolvimento *front-end* e do *back-end* não há demais custos, pois serão utilizadas ferramentas gratuitas para tal. Para o possível *deploy* da aplicação não há muitos custos tendo em vista a gama de serviços de hospedagem a preços favoráveis disponíveis no tempo atual.

Sobre a viabilidade operacional, o sistema apresenta vantagens por ser acessado por qualquer sistema operacional ou dispositivo, graças ao seu formato de *web*, eliminando a necessidade de instalações e evitando qualquer alteração na infraestrutura tecnológica da organização. Entretanto, o sistema apresenta a necessidade de conexão com a *internet*, o que resulta na indisponibilidade do sistema quando houver queda ou falta de conexão.

## 1.5 Estrutura do trabalho

O *front-end* foi desenvolvido em HTML e CSS, utilizando Bootstrap para criar as tabelas de consulta das páginas. O Bootstrap é uma ferramenta gratuita para desenvolvimento HTML, CSS e JS. Podendo criar protótipos rápidamente ou aplicações completas com variáveis e mixins Sass, sistemas de grid responsivo, componentes pré-construídos e *plugins* com jQuery. (BOOTSTRAP, 2022).

O back-end desenvolvido com a linguagem Java, em conjunto com o banco de dados PostgreSQL. Para o desenvolvimento dos casos de uso, foi utilizado o *software* StarUML.

A IDE utilizada no desenvolvimento foi o NetBeans versão 12, ferramenta de desenvolvimento Java gratuita e de código aberto, o que evita custos para o desenvolvimento do sistema. (WIKIPEDIA, 2021).

O SGBD utilizado no sistema é o PostgreSQL, ferramenta gratuita e também de código aberto. (DEVMEDIA, 2021).

Para o deploy do sistema foi utilizado o Heroku. O Heroku é uma plataforma para dados e aplicativos - fornecendo um banco de dados como serviço seguro e escalável com toneladas de ferramentas de desenvolvedores, como seguidores de banco de dados, bifurcação, clipes de dados e verificações de integridade automatizadas. (HEROKU, 2022).

# **2. Trabalhos Correlatos Existentes**

Não foram encontrados trabalhos similares, ou outros sistemas que façam o gerenciamento de um serpentário.

# **3. Requisitos**

Apresentação dos requisitos do sistema gerenciador de serpentário.

3.1 Requisitos funcionais

|  |  |
| --- | --- |
| Requisito Funcional 001 | Salvar novas serpentes |
|  | Para adicionar novas serpentes, todos os campos do cadastro devem ser preenchidos. O código deve ser um campo único |

Tabela 01 – RF 01

|  |  |
| --- | --- |
| Requisito Funcional 002 | Atualização de informações |
|  | Só podem ser atualizados os dados de serpentes já cadastradas. |

Tabela 02 – RF 02

|  |  |
| --- | --- |
| Requisito Funcional 003 | Gráficos |
|  | Para gerar os gráficos, é necessário que haja ao menos 2 registros de cada serpente.  Exemplo: 2 entradas de peso, 2 entradas de alimentação. |

Tabela 03 – RF 03

|  |  |
| --- | --- |
| Requisito Funcional 004 | Exportação de gráficos |
|  | O gráfico pode ser exportado para PDF ou impresso. |

Tabela 04 – RF 04

3.1 Requisitos não funcionais

|  |  |
| --- | --- |
| Requisito não funcional 001 | Acesso ao sistema |
|  | Para adicionar serpentes e informações, é necessário que o usuário tenha permissão de administrador ao sistema.  Para acessar gráficos e fazer consultas, qualquer usuário tem acesso. |

Tabela 05 – RNF 01

|  |  |
| --- | --- |
| Requisito não funcional 002 | Acesso à *Internet* |
|  | O sistema será feito em plataforma WEB, e para conseguir se conectar é necessário que haja conexão à internet. A velocidade da internet impactará diretamente no desempenho do sistema. |

Tabela 06 – RNF 02

|  |  |
| --- | --- |
| Requisito não funcional 003 | Impressão de gráfico |
|  | Necessário que o funcionário tenha uma impressora para que o gráfico seja impresso. |

Tabela 07 – RNF 03

# **4. Diagramas UML**

4.1 Diagrama de atividades:

Ao fazer *login* no sistema, o funcionário deverá escolher qual ação irá executar, podendo escolher entre cadastrar uma nova serpente, inserir um funcionário, atualizar as informações de serpente e funcionário, atualizar o estoque de camundongos caso tenha sido recebido uma nova remessa e também é possível gerar gráficos. (IMAGEM 01)

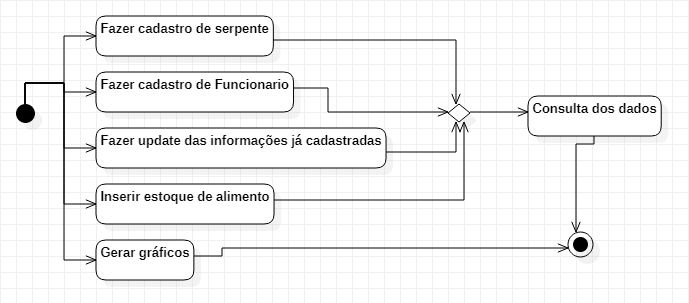


Imagem 01 – Diagrama de atividades

4.2 Casos de uso

O sistema possui somente um agente, que possui acesso irrestrito ao sistema. (IMAGEM 02)

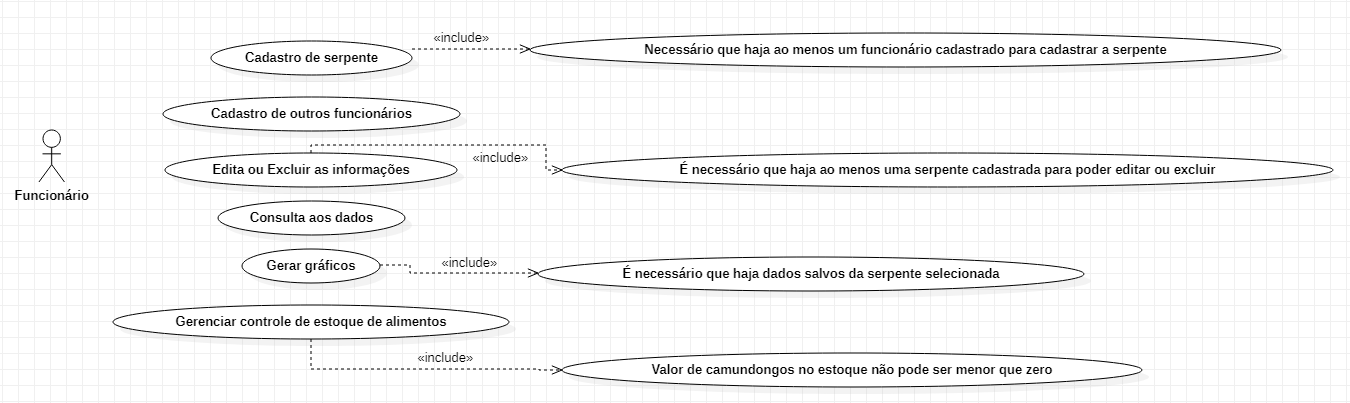


Imagem 02 – Diagrama de caso de uso

Cadastro das serpentes:

Ao fazer o cadastro das serpentes, todos os dados solicitados devem ser preenchidos. Para que seja possível realizar o cadastro, é necessário associar o cadastro a um funcionário.

Cadastro de funcionários:

Ao fazer o cadastro de funcionário, devem ser inseridos todos os dados solicitados.

Editar ou excluir informações:

Edição ou exclusão dos itens da tabela.

Consulta dos dados:

É necessário selecionar a serpente desejada para verificar as informações do animal.

Gerar Gráficos:

Será gerado o gráfico desejado pelo funcionário com base nas informações já salvas.

Gerenciar controle de estoque de alimentos:

É necessário manter o estoque de alimentos atualizado, inserindo novos camundongos quando receber uma nova remessa. O valor de camundongos não pode ser menor que zero.

4.3 Diagrama de Classes

A principal classe do projeto é a classe serpente. Nela são gerenciados todos os dados e métodos das serpentes, anexo a esta classe está a classe do funcionário, é necessário a ligação entre a classe funcionário e serpente pois deve ser armazenado o nome do funcionário que realizou o cadastro ou alteração dos dados das serpentes.

A classe camundongos serve somente para armazenar a quantidade de camundongos presentes no estoque. (IMAGEM 03).

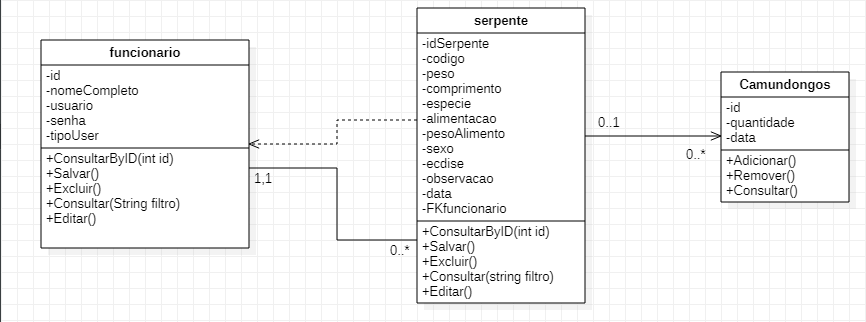


Imagem 03 – Diagrama de classes

# **5. Modelagem de Dados**

O banco de dados é constituído de 3 tabelas. Uma tabela responsável por armazenar os as informações sobre os funcionários do serpentário, onde apenas algumas informações básicas são solicitadas, como Nome completo, usuário e senha.

A próxima tabela é a de serpentes, onde devem ser cadastrados o máximo de informações sobre cada serpente para se ter o diagnóstico mais completo sobre cada serpente. Essa tabela possui um campo para inserir o ID do funcionário, devido a necessidade de registrar qual foi o funcionário que realizou o cadastro ou alterou as informações sobre a serpente.

A última tabela é a tabela de camundongos, o alimento dado para as serpentes. Como esses camundongos são disponibilizados por um terceirizado, o que nos interessa é unicamente um controle de estoque, onde é necessário apenas saber apenas a quantidade de camundongos no estoque.

# **6. Projeto da Interface**

A interface do projeto está sendo desenvolvida com paginas JSF, onde a codificação é feita com linguagem de marcação HTML e o visual do site é feito com linguagem CSS. O projeto tem o objetivo de não possuir inúmeras telas, onde o usuário necessita clicar em vários botões até chegar na página desejada.

Após fazer o login, a página de menu é aberta, e nela é possível escolher se queremos acessar as informações do funcionário onde é possível pesquisar, editar e excluir ou ir para outra página e inserir um novo registro. O mesmo acontece para as serpentes. Há ainda a página responsável pelo controle de estoque e a página de geração de gráficos. Porém essa última tela não foi desenvolvida.

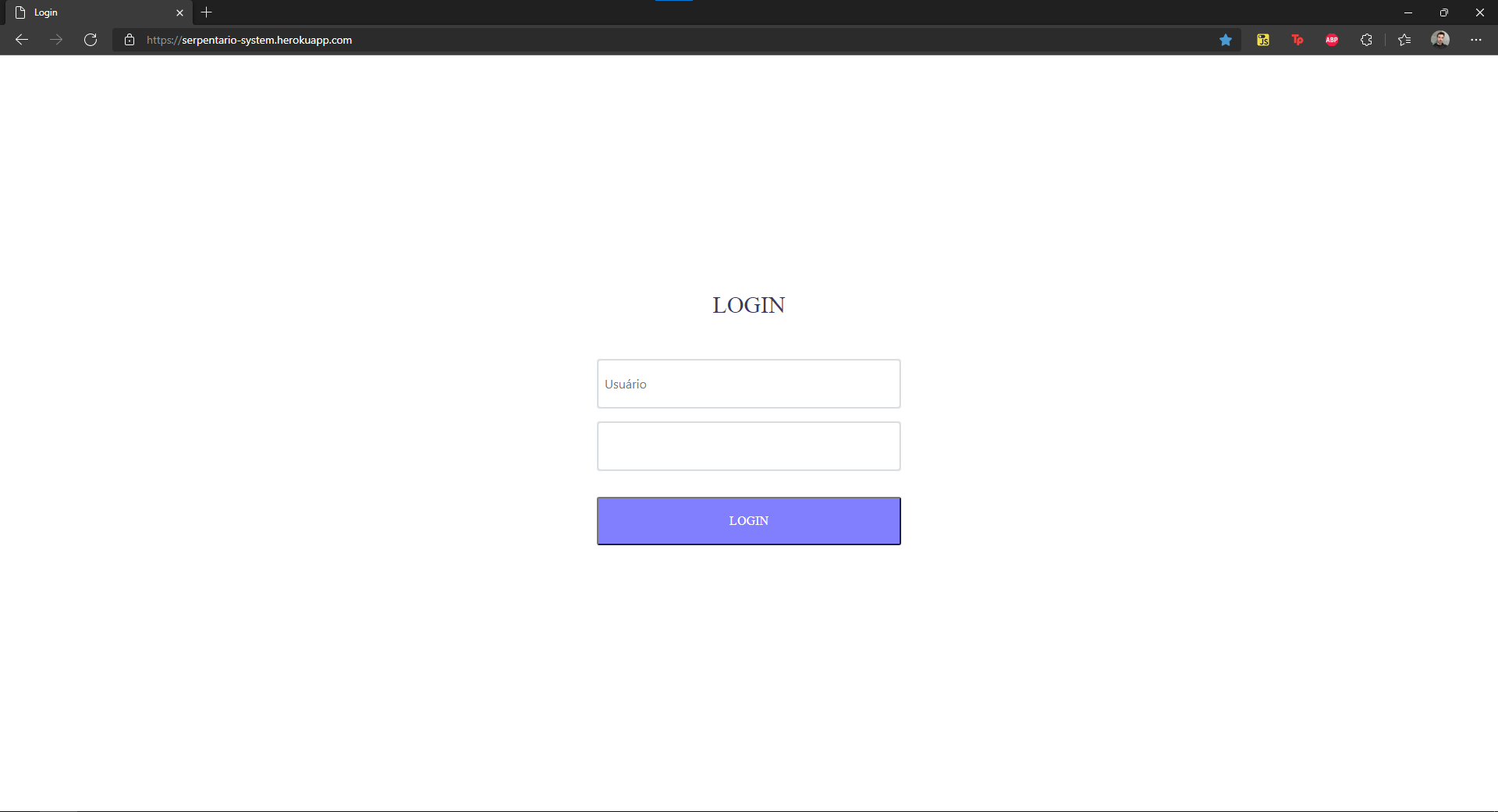


Imagem 04 – Tela de *Login*

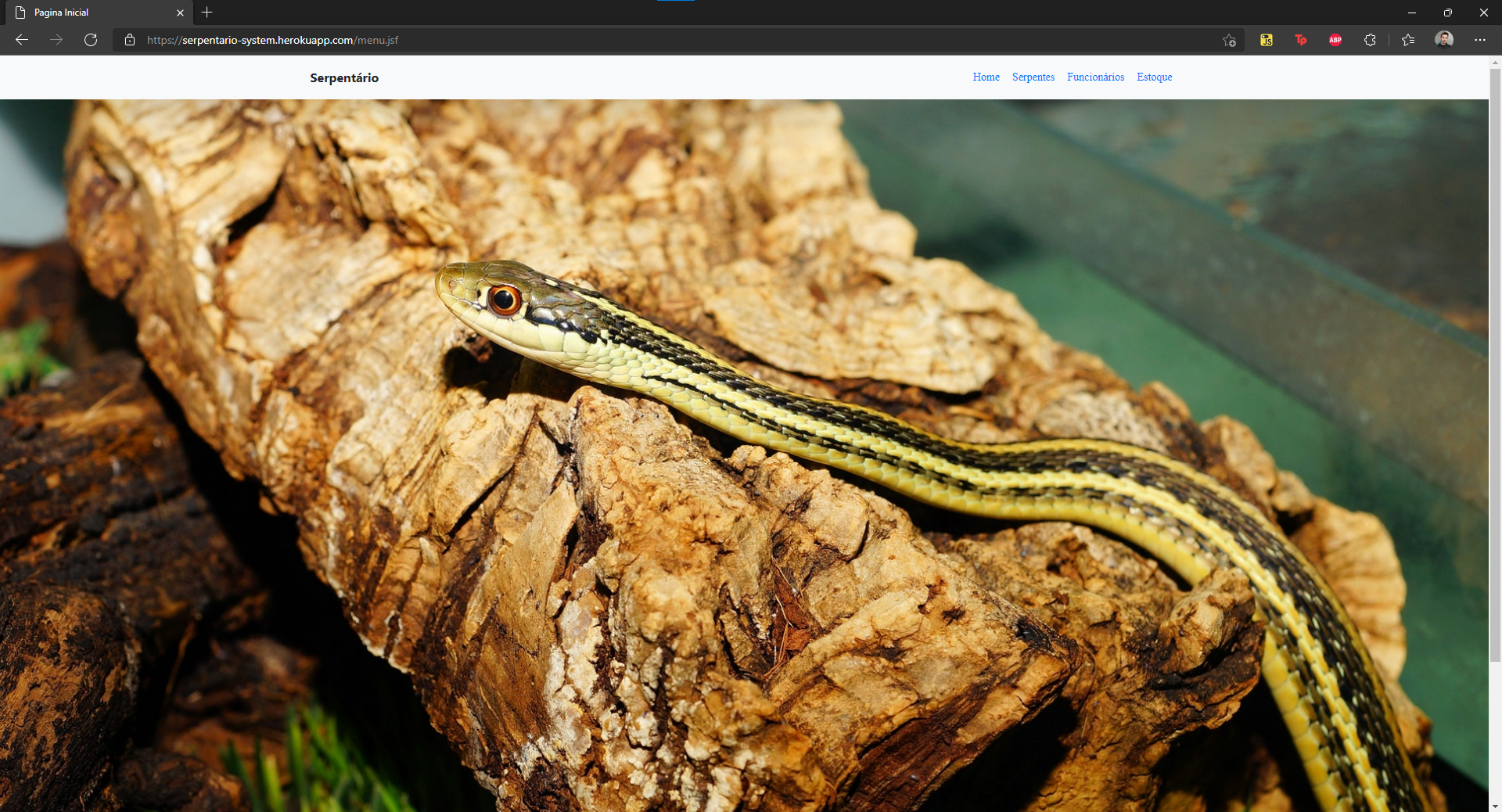


Imagem 05 – Menu Principal

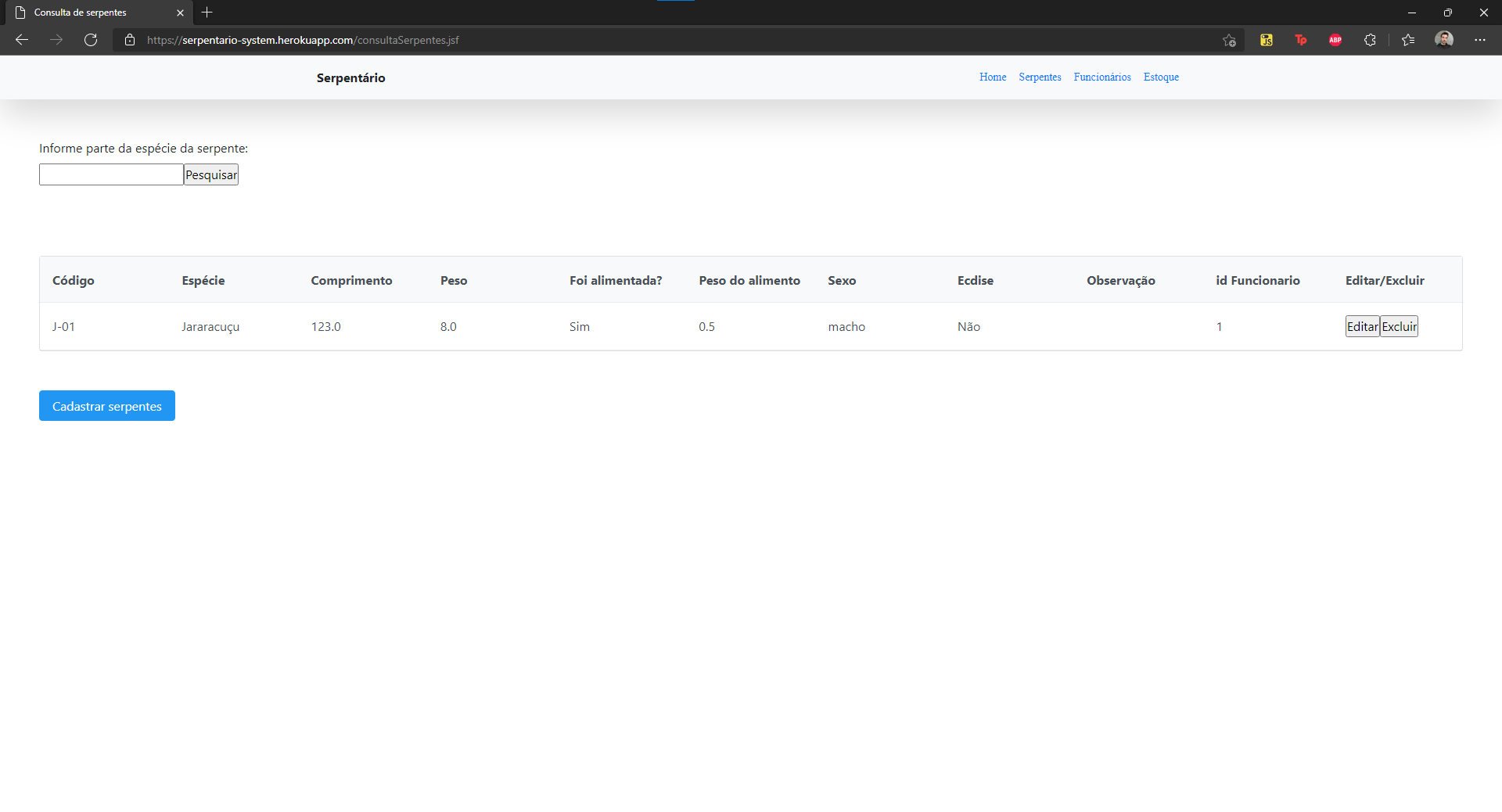


Imagem 06 – Consulta de serpentes

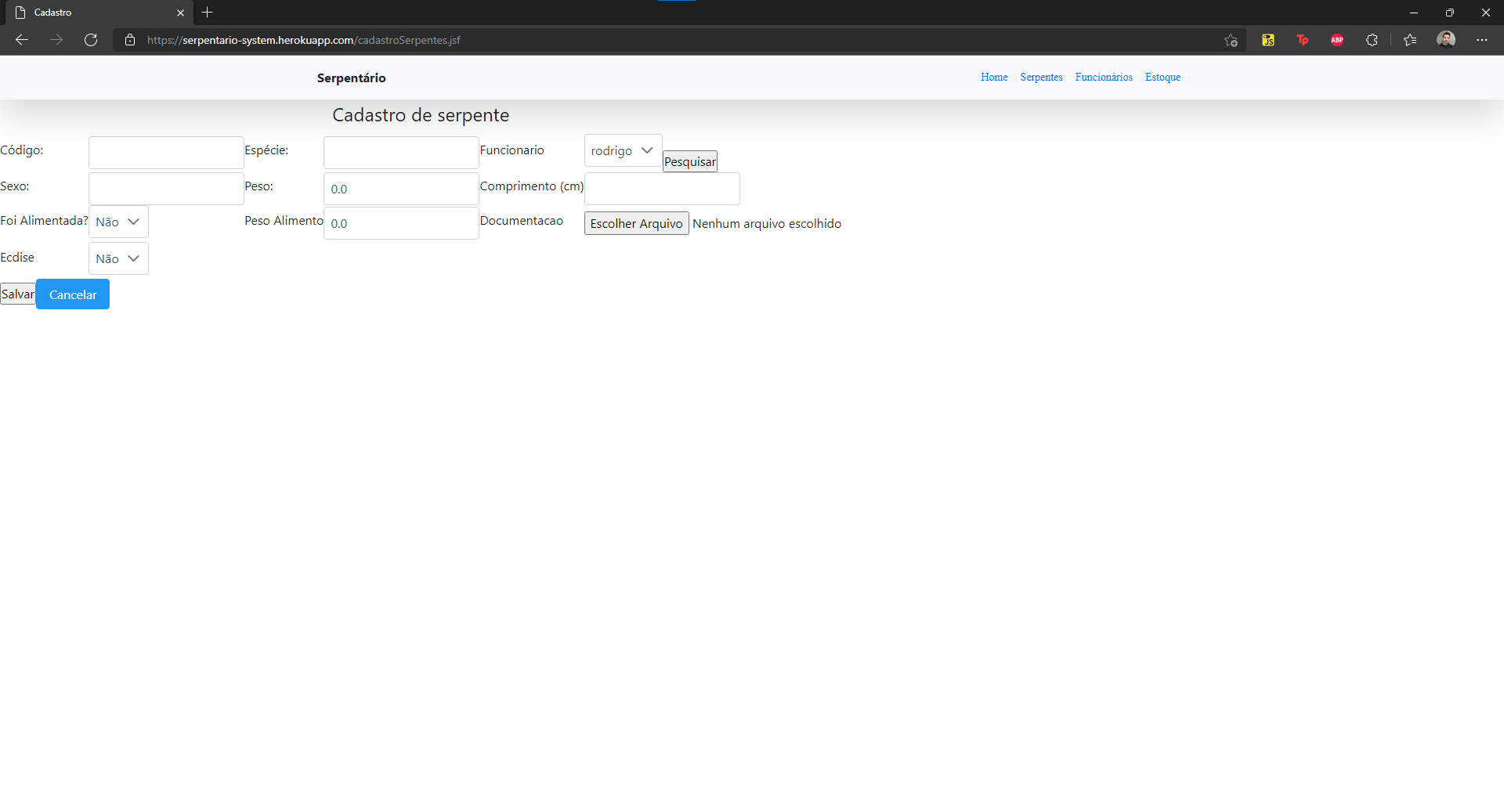


Imagem 07 – Cadastro de serpentes

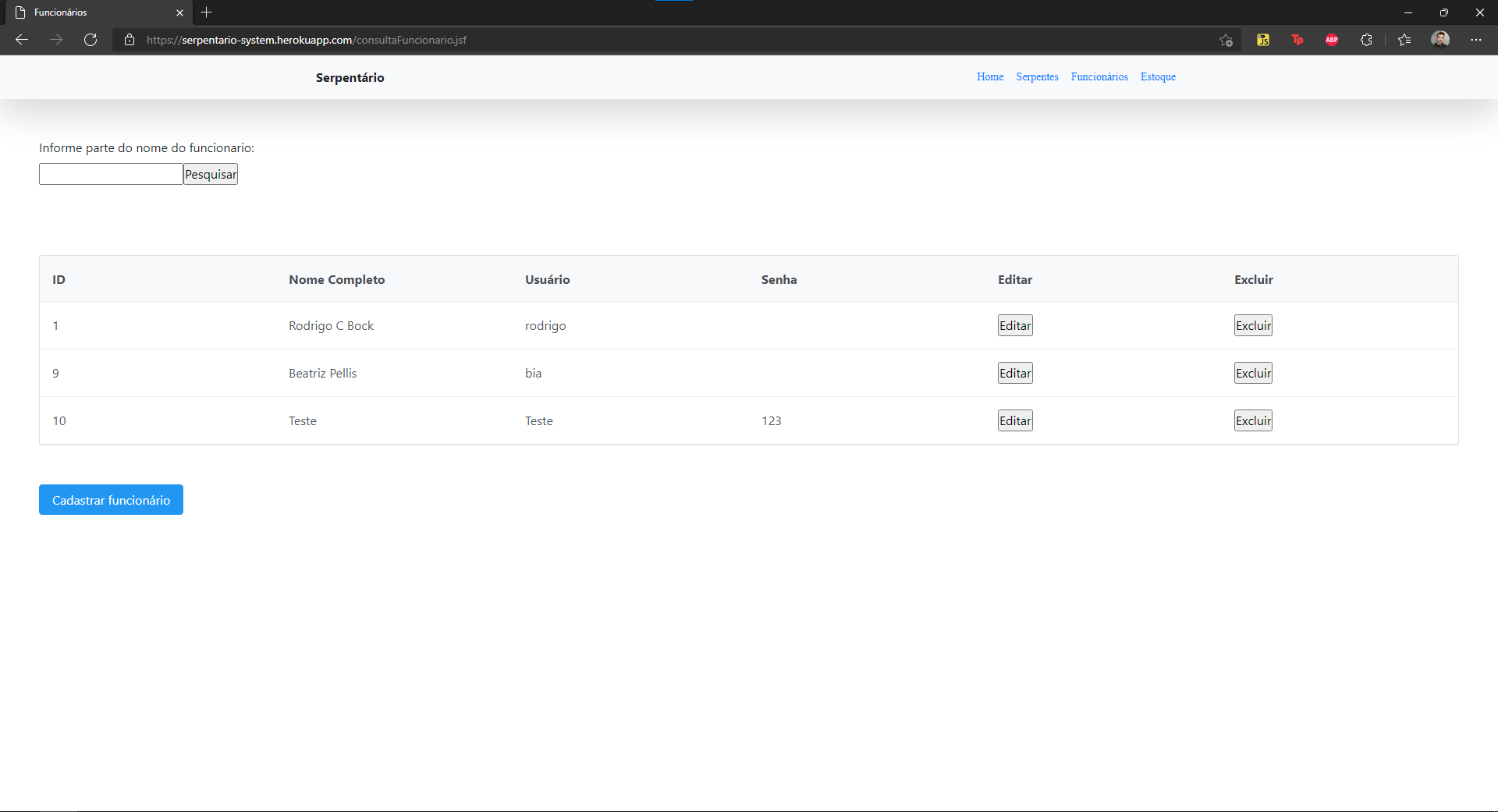


Imagem 08 – Consulta de Funcionários

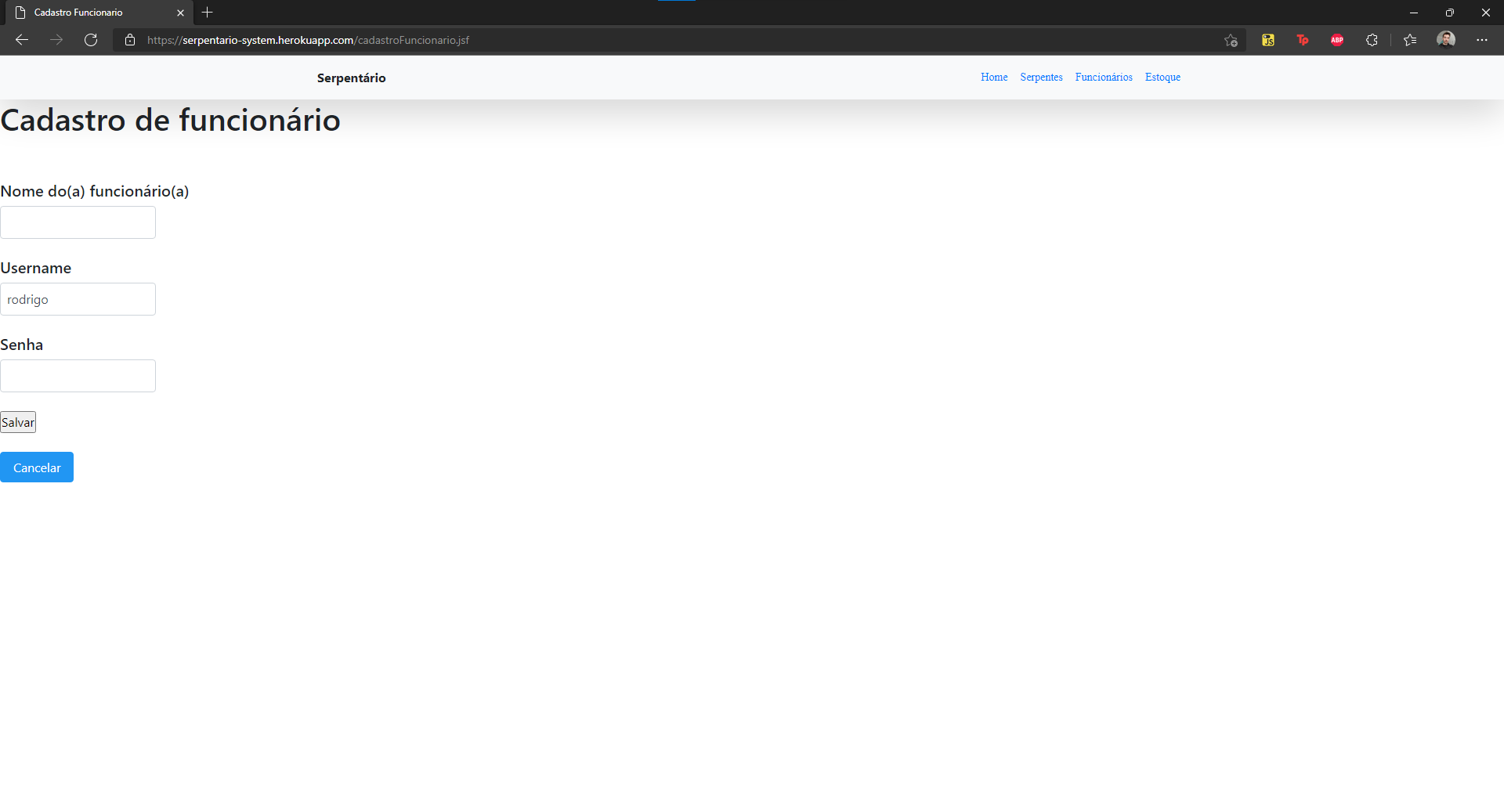


Imagem 09 – Cadastro de funcionário

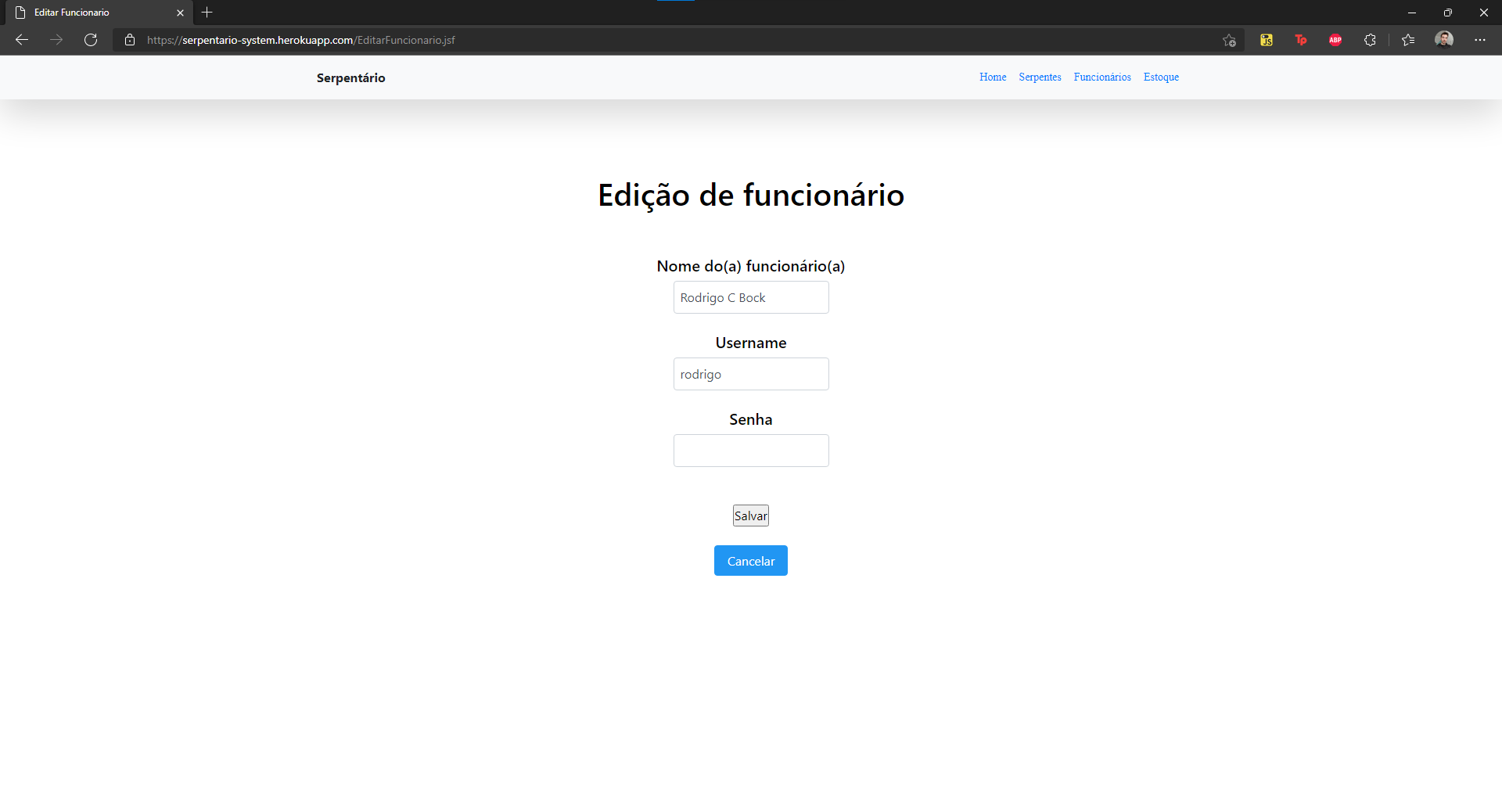


Imagem 10 – Edição de funcionário



Imagem 11 – Edição de serpentes

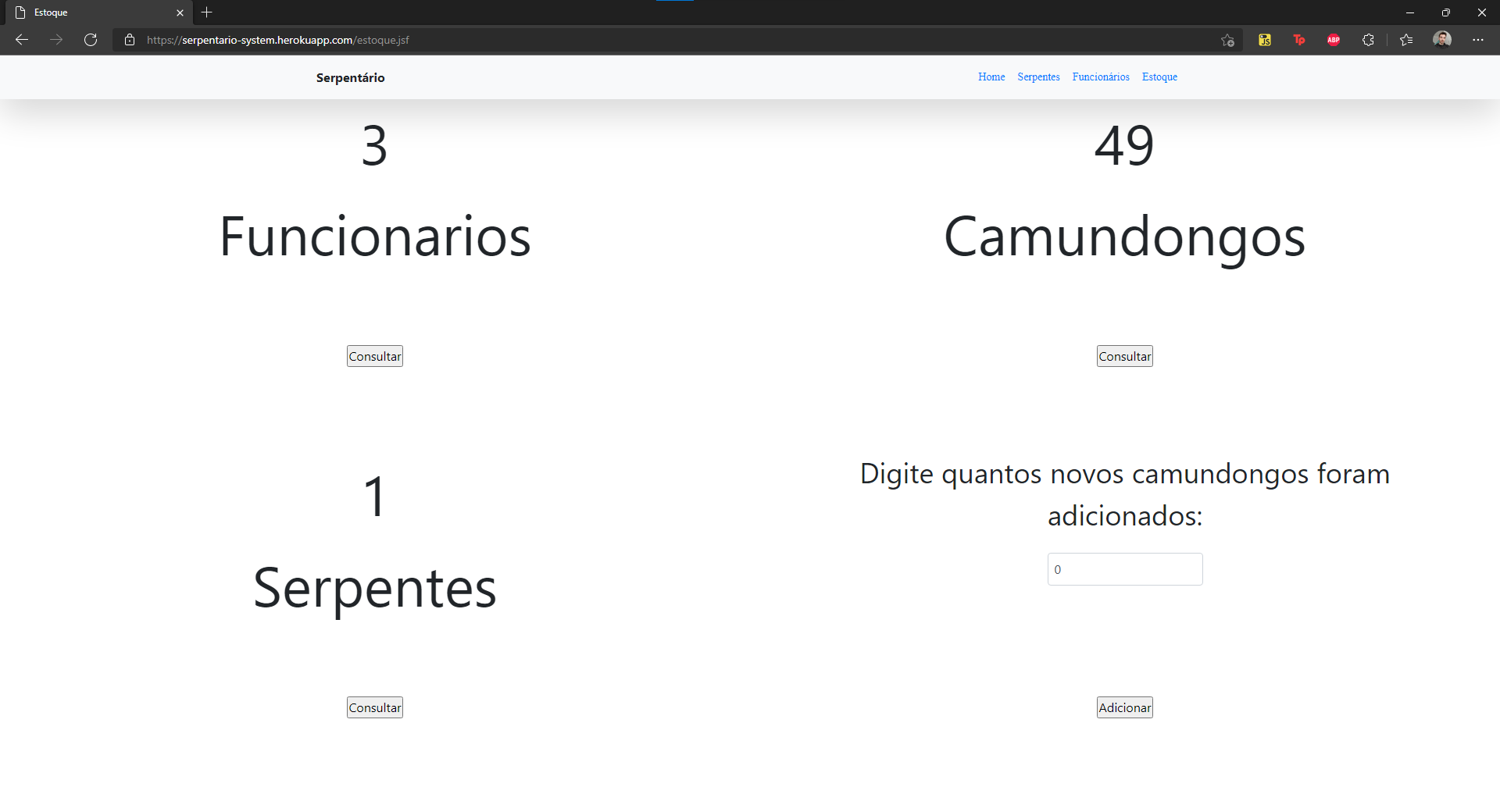


Imagem 12 – Tela de estoque

# **7. Implementação**

A implementação do projeto será feita utilizando o *site* Heroku. O banco de dados já foi migrado para o Heroku e atualmente o projeto já foi migrado também, rodando totalmente online.

# **8. Resultados Obtidos**

Até o seguinte momento todo o desenvolvimento do projeto está seguindo de forma planejada. O banco de dados foi criado e adicionado ao projeto, o *CRUD* (Criar, Editar, Excluir) está funcionando completamente tanto para o funcionário como para as serpentes.

Algumas telas precisaram ser refeitas ou reorganizadas para que fiquem mais bonitas visualmente. Outras telas ainda precisam ser ajustadas para que fiquem posicionadas corretamente na tela. Veja trabalhos futuros.

Problemas ocorreram desde a criação do primeiro protótipo até o atual projeto, no primeiro projeto a ideia era utilizar JPA para conectar ao banco de dados, porém houve falhas e erros ao conectar o banco e salvar os registros, dessa forma foi decidido que os comandos SQL seriam escritos dentro dos métodos do *back-end*.

Até o dado momento, quase todas as telas já estão concluídas, faltando apenas a criação da tela de gráficos.

Outro problema que pode ser citado é que inicialmente o projeto utilizava o servidor *GlassFish* para rodar, quando decidido que a aplicação iria rodar em um servidor remoto, todas as opções apontavam para o servidor *Mavem*. Então foi necessário criar um novo projeto com *mavem* e copiar todas as telas, classes e bibliotecas.

# **9. Considerações Finais e Trabalhos Futuros**

Para os trabalhos futuros, segue abaixo uma lista do que é necessário fazer para melhorar o sistema:

Criação das telas com gráficos Altura x Peso, Altura x Idade, Peso x Idade.

Aprimoramento do front-end: algumas telas já estarão prontas, porém pode ser necessário correções.

# **10. Referências Bibliográficas**

JAVA.The Java EE 6 Tutorial. Disponível em: https://docs.oracle.com/javaee/6/tutorial/doc/bnbpz.html. Acesso em: 28 jun. 2021.

MORO, Vilson. Tadsprogv2021-1. Disponível em: https://github.com/instrutorvilson/tadsprogv2021-1.git. Acesso em: 28 jun. 2021.

PELLIS, Beatriz. Comunicação pessoal. Blumenau, Universidade Regional de Blumenau, 2021.

Porto Editora – *serpentário* no Dicionário infopédia da Língua Portuguesa [em linha]. Porto: Porto Editora. [consult. 2021-10-10 17:22:05]. Disponível em https://www.infopedia.pt/dicionarios/lingua-portuguesa/serpentário

DevMedia – Guia de PostgreSQL. [consult. 2021-10-10 18:10:05].

Disponivel em: https://www.devmedia.com.br/guia/guia-de-postgresql/34328

NETBEANS. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2021. Disponível em: <<https://pt.wikipedia.org/w/index.php?title=NetBeans&oldid=61784947>>. Acesso em: 10 out. 2021.

Bootstrap. Pagina Inicial do site Bootstrap.

Disponível em: < https://getbootstrap.com.br/ >

Aceso em: 27/01/2022.

Heroku, What is Heroku.

Disponivel em: < <https://www.heroku.com/what>>

Acesso em: 27/01/2022