# Desenvolvimento de um Sistema de Puericultura para o Posto de Saúde Village

**Julia Vadillo Andrade[1], Daniela Marques[1]**

[1]Instituto Federal de Educação, Ciência e Tecnologia de São Paulo - Câmpus Hortolândia
Avenida Thereza Ana Cecon Breda, s/n, Vila São Pedro - Hortolândia – SP – Brasil

juliavadillo@hotmail.com, marquesdaniela@ifsp.edu.br

***Abstract.*** *Automating a work process can bring several benefits, including increase the productivity and easier access to information. This paper describes the development of a system to attend the need to automate a childcare process at Posto de Saúde Village located in the Campinas sub-district, detailing how it was created using technologies for web development and the incremental development model.*

***Resumo.*** *Automatizar um processo de trabalho pode trazer diversos benefícios, entre eles estão aumento de produtividade e facilidade de acesso à informação. Este artigo descreve o desenvolvimento de um sistema para atender a necessidade de automatização de um processo da área de puericultura do Posto de Saúde Village situado no subdistrito de Campinas detalhando como foi sua criação com uso de tecnologias para desenvolvimento web e o modelo de desenvolvimento incremental.*

## 1. Introdução

A puericultura é a área da saúde que se dedica aos cuidados da criança do 0 aos 2 anos, ela consiste em realizar consultas regulares para acompanhar o crescimento e desenvolvimento da criança, detectar distúrbios e doenças precocemente, monitorar a vacinação e obter dados de saúde sanitária da região no qual a criança se encontra. De acordo com Quiles (2015), do Departamento de Pediatria Ambulatorial e Cuidados Primários da Sociedade de Pediatria de São Paulo (SPSP):

> *"Puericultura é a arte de promover e proteger a saúde das crianças, através de uma atenção integral, compreendendo a criança como um ser em desenvolvimento com suas particularidades. É uma especialidade médica contida na Pediatria que leva em conta a criança, sua família e o entorno, analisando o conjunto bio-psico-sócio-cultural".*

O Posto de Saúde Village fica situado na comunidade Village Campinas do distrito de Barão Geraldo, em Campinas. estado de SP, sua instalação é em uma antiga galeria e as salas e espaço foram adaptados para atender a população da região. Atualmente o posto atende em média 90 crianças por ano para realizar as consultas de puericultura através do Sistema Único de Saúde (SUS).

Após uma visita ao posto para conhecer como é o processo de puericultura, foi levantado que atualmente todas as informações das consultas são anotadas em um caderno e controladas por uma única enfermeira. Destacou-se também que anualmente a prefeitura solicita informações a respeito das consultas realizadas como o total de crianças

atendidas, crianças vacinadas, não comparecimento de consultas, incidências de doenças na região, entre outras informações que são solicitadas esporadicamente. Por estar tudo anotado manualmente em um caderno e por ter uma única pessoa no controle desses dados, fica difícil o levantamento das informações quando solicitadas pela prefeitura assim como um controle preciso durante as consultas. Os pais das crianças possuem uma caderneta de controle (Figura 1), onde são registrados os dados de cadastro e nascimento da criança, bem como o controle de vacinação e uma curva de crescimento para acompanhar o ritmo de desenvolvimento (Figura 2).


**Figura 1. Caderneta utilizada para acompanhamento das crianças**

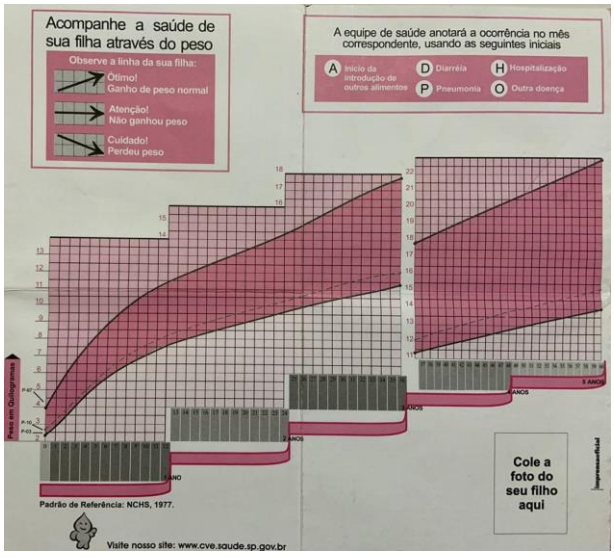


**Figura 2. Curva de crescimento da criança**

A partir do desejo dos funcionários do posto Village de possuir um sistema de consulta automatizado que pudesse ser utilizado pelos médicos e enfermeiros que realizam o atendimento, do reconhecimento da importância da puericultura e de como os dados bem extraídos podem gerar informações de grande importância para a comunidade,

foi reconhecido a necessidade de desenvolver um sistema para fazer o controle de crianças, consultas e geração de relatórios, o sistema também deveria ter uma arquitetura simples para que a infraestrutura do posto, com computadores de segunda mão e internet a rádio, conseguisse aguentar e operá-lo sem problemas de desempenho.

Este artigo descreve em suas seções como foi realizado o desenvolvimento do sistema para o Posto de Saúde Village, na Seção 2 estão descritos os Materiais e Métodos utilizados para que o sistema fosse desenvolvido, na Seção 3 se encontra o Desenvolvimento, onde todo o processo é detalhado e por fim na Seção 4 está a Conclusão e sugestões para atualizações futuras do sistema.

## 2. Materiais e Métodos

### 2.1 Engenharia Web

Com o surgimento da Web e o exponencial crescimento da internet, *sites* e aplicativos surgem cada vez mais neste meio, suas atualizações e evoluções pediram uma maior estrutura e um novo modelo de metodologia para atender todas as novas características da web como concorrência, carga imprevisível, disponibilidade, sensibilidade ao conteúdo, evolução dinâmica, imediatismo, segurança e estética (PRESSMAN, 2010), com essa demanda, foi criada a nova subárea da Engenharia de Software, a Engenharia Web. Esta nova subárea é baseada na engenharia padrão, foca nas metodologias, técnicas e ferramentas para manter aplicações e sistemas baseados na web de alta qualidade.

Além de metodologias e técnicas de desenvolvimento, ao final é necessário avaliar o produto gerado. Para uma boa avaliação da qualidade final de uma Aplicação Web, devem ser notadas características tais como funcionalidade, usabilidade, confiabilidade, eficiência, manutenibilidade e portabilidade (ISO, 1991).

## 2.2 Modelo Incremental

O modelo Incremental é uma combinação entre o modelo linear e o modelo de prototipação, no qual o desenvolvimento é feito em partes independentes denominadas incrementos, esses incrementos são entregáveis e prontos para uso do cliente.

O primeiro incremento é frequentemente chamado de “núcleo do produto” (PRESSMAN, 2006) e contém a implementação dos requisitos básicos para que o sistema possa funcionar e atender minimamente às necessidades do cliente. Cada aprimoramento é lançado como uma versão. Novas versões são criadas até que o sistema fique completo e adequado, para então, ser lançada a versão final, como pode ser visto na Figura 3.

## 2.3 Engenharia de Requisitos

De acordo com Sommerville (2007, p.49), a Engenharia de Requisitos tem como objetivo definir o que o sistema deve fazer, quais as necessidades reais e identificar quais restrições existem para que o software seja desenvolvido. Uma das fases da engenharia de requisitos é a elicitação e análise de requisitos que é definida como a etapa que o engenheiro de software trabalha com os clientes e usuários finais do sistema para aprender sobre o domínio da aplicação, serviços que serão fornecidos, desempenho esperado e alguma restrição para o sistema se houver.

A coleta de requisitos é realizada nessa fase de elicitação e análise de requisitos e possui várias técnicas para que seja realizada. Neste projeto a técnica escolhida foi a entrevista, uma das técnicas tradicionais realizada em projetos que software, que produz

bons resultados na fase inicial de levantamento de requisitos. Por meio desta técnica a equipe de desenvolvimento formula questões para os *stakeholders* sobre o sistema que eles usam e o sistema a ser desenvolvido, e a partir de suas respostas é possível identificar as dificuldades do sistema atual, as necessidades do cliente e como deverá ser a interação do cliente com o sistema.

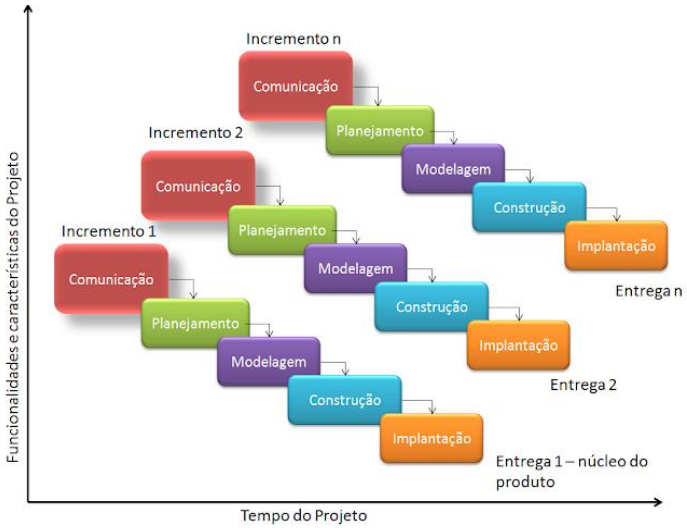


**Figura 3. Ciclo de vida Modelo Incremental**

Fonte: (PRESSMAN, 2010)

## 2.4 Diagrama de Classe

Diagrama de classes é uma representação da estrutura e relações das classes que servem de modelo para objetos, de acordo com a *Unified Modeling Language* (UML). Neste diagrama, uma classe é representada por um retângulo com três divisões, são elas: nome da classe, seus atributos e seus métodos. A Figura 4 representa uma classe de acordo com a notação da UML.

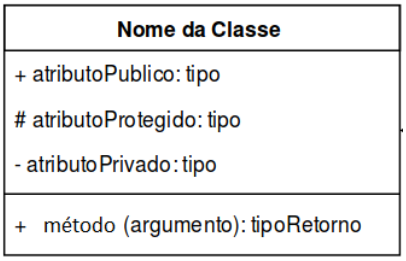


**Figura 4. Representação de classe no modelo UML**

## 2.5 Tecnologias

Para realizar o desenvolvimento do sistema a linguagem de programação escolhida foi o Java e as tecnologias escolhidas são todas gratuitas, ou seja, não possuem custo para obter a licença. Nas subseções abaixo estão descritas as tecnologias utilizadas.

### 2.5.1 JSF

*Java Server Faces* (JSF) é uma tecnologia que permite criar aplicações Java para Web utilizando componentes visuais pré-prontos em um formulário e ligando-os a objetos Java, fazendo a separação das camadas de apresentação e de aplicação (Oracle, 2020).

### 2.5.2 Maven

O Apache Maven é uma ferramenta de automação e gerenciamento de projetos Java, embora também possa ser utilizada com outras linguagens. Ela fornece às equipes de desenvolvimento uma forma padronizada de automação, construção e publicação de suas aplicações, agregando agilidade e qualidade ao produto final. É também utilizada em processos como *build*, gerenciamento de dependências e documentação de projetos (Apache Maven, 2020).

### 2.5.3 Hibernate

O Hibernate é um *framework* para o mapeamento objeto-relacional. Este *framework* facilita o mapeamento dos atributos entre uma base de dados relacionais e o modelo objeto de uma aplicação, mediante o uso de arquivos (XML) ou anotações Java (DevMedia, 2009).

### 2.5.4 Materialize

Desenvolvido pela Google e inspirado no *Material Design* (*design* utilizado no sistema operacional para *smartphones* Android desde a versão 5.0), o Materialize é um *framework Front-End* que apresenta opções de componentes prontos em sua biblioteca com *design* mais leve, fazendo com que o projeto fique visualmente mais agradável (Materialize, 2020).

### 2.5.5 Jasper Report

O JasperReport é um *framework* desenvolvido em Java que fornece funcionalidades que permitem criar relatórios complexos de forma estruturada, a partir da elaboração de um *template*, e exportá-los para diversos formatos, como: HTML, PDF e DOC. O *template* é um arquivo XML com a extensão .jrxml e ele pode ser criado através da ferramenta JasperSoft Studio (DevMedia, 2012).

### 2.5.6 PostgreSQL

O PostgreSQL foi o gerenciador de banco de dados escolhido para armazenar os dados do sistema, ele é um dos gerenciadores Open Source mais avançados do mercado que fornece diversos recursos (PostgreSQL WebSite).

### 2.5.7 GitHub

GitHub é um repositório *online* de código utilizado para armazenar, gerenciar e versionar os códigos nele depositados através do Git, que é o sistema de controle de versão utilizado no projeto. (Hostinger, 2019)

## 3. Desenvolvimento

Nesta seção é descrito todo o processo de desenvolvimento do sistema de puericultura. Para iniciar o desenvolvimento do sistema foi criado um repositório no GitHub (https://github.com/juliavadillo/pueri) para que os incrementos desenvolvidos fossem armazenados.

## 3.1 Levantamento de Requisitos

De acordo com o modelo incremental, a primeira etapa do desenvolvimento é a Comunicação, nesta etapa que acontece o Levantamento de Requisitos. Neste projeto foi aplicada a técnica da entrevista com aos profissionais do Posto de Saúde Village para compreender a dinâmica de uma consulta de puericultura, as necessidades dos profissionais, quais as informações eram necessárias no sistema, o material que era utilizado em cada consulta, quais eram as prioridades para as funcionalidades do sistema e qual a infraestrutura disponível no postinho para hospedar o sistema.

Nesta etapa foram apresentadas as cadernetas já exibidas anteriormente. Elas são a base de informações do sistema e a partir delas foram explicados todos os detalhes do seu preenchimento.

Ao contrário do que traz o modelo incremental que a comunicação com o usuário seja realizada no começo de cada incremento, nesse projeto a comunicação foi realizada apenas uma vez, a entrevista nos possibilitou ter os requisitos bem definidos e por isso não foi necessário realizar comunicação nos incrementos seguintes.

## 3.2 Análise de Requisitos e Definição dos incrementos

De acordo com o modelo Incremental, é necessário fazer seu planejamento e sua modelagem. Após a entrevista foi possível realizar a análise de requisitos, identificando requisitos funcionais e não-funcionais. O requisito não-funcional levantado foi que todas as tecnologias utilizadas no sistema deveriam ser gratuitas por se tratar de um cliente da saúde pública e há uma peculiaridade no posto de saúde, a internet é por rádio. Os requisitos funcionais foram agrupados em funcionalidades e separados por incrementos relacionados na Tabela 1.

**Tabela 1. Relação funcionalidades e incrementos**

| Incrementos | |
|---|---|
| Incremento 1 | Cadastro de Paciente |
| Incremento 2 | Consulta de Paciente |
| Incremento 3 | Registro de Consulta |
| Incremento 4 | Lista de consulta por Paciente |
| Incremento 5 | Geração de Relatórios |
| Incremento 6 | Login no Sistema |

### 3.2.1 Desenvolvimento dos Incrementos

Outra etapa do modelo Incremental é a Construção. O desenvolvimento dos incrementos baseou-se principalmente na criação ou adaptação das entidades Java envolvidas, criação da página xhtml para fazer a interface com o usuário e na construção da classe e ou métodos *bean* que fariam a interação com a aplicação para aplicar as regras do sistema, as dependências do Maven também foram adicionadas de acordo com o andamento do projeto e a necessidade.

#### 3.2.1.1 Diagrama de Arquitetura MVC

As tecnologias citadas anteriormente foram escolhidas por serem próprias para desenvolvimento web, abaixo na Figura 5 está o Diagrama de Arquitetura estruturado no modelo MVC para o projeto. Na camada *Controller* está o *Faces Servlet* que é responsável por receber as requisições do cliente e distribuir para as classes de regra no

*Model* e posteriormente renderizar as páginas com a resposta da requisição. Na camada *Model* estão os *ManagedBeans* e as entidades JPA que realizam o acesso ao banco de dados PostgreSQL e que são responsáveis por executar as regras da aplicação e devolver para o *Controller* a saída da regra. Por último está a camada *View* que possui as páginas JSF em xhtml com componentes visuais do Materialize que quando acionadas pelo *controller* devolvem a resposta da requisição para o cliente.

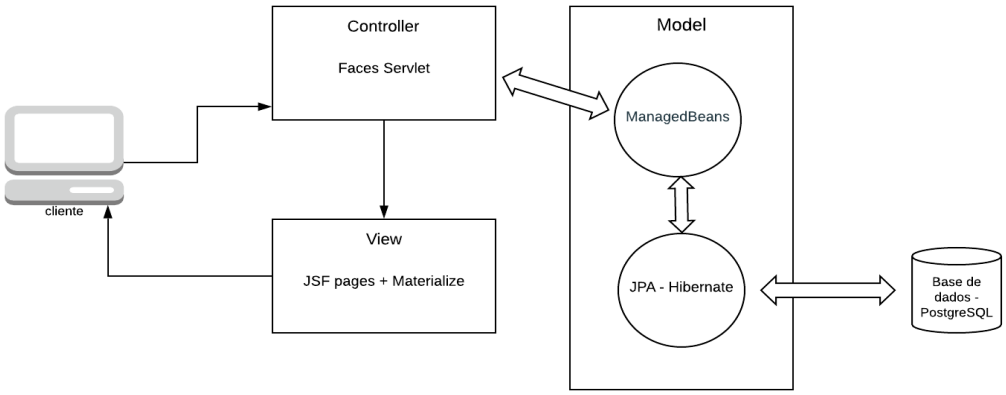


**Figura 5. Diagrama de Arquitetura MVC**

### 3.2.1.2 Diagrama de Classes

As entidades utilizadas para persistência no banco de dados estão descritas no diagrama de classes na Figura 6. As entidades foram mapeadas de acordo com o desenvolvimento dos incrementos para que fosse possível utilizar o Hibernate para fazer as interações com o banco de dados e o diagrama foi criado após o término do desenvolvimento por engenharia reversa.

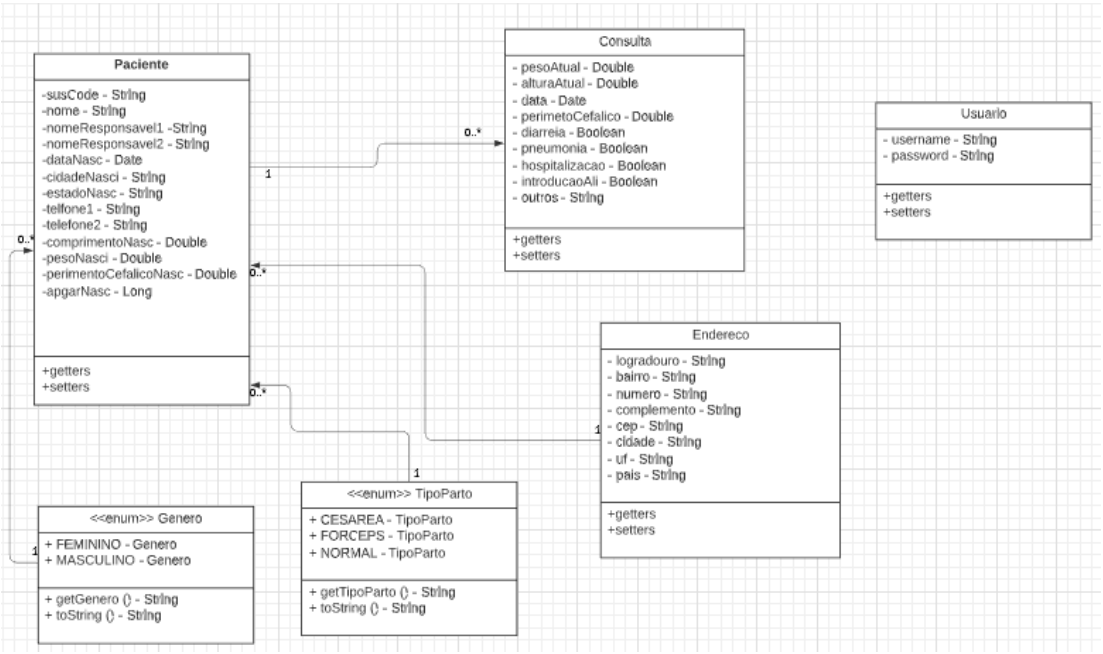


**Figura 6. Diagrama de Classe das Entidades**

### 3.2.2 Incremento 1 - Cadastro de Paciente

Nesse incremento foi desenvolvida a funcionalidade Cadastro de Paciente (Figura 7), considerado núcleo do projeto pois a partir deste cadastro que se darão as outras funcionalidades. No cadastro são inseridos dados básicos do paciente e dados do nascimento. Os dados utilizados são baseados na caderneta do paciente utilizada atualmente e apresentada durante o levantamento de requisitos.

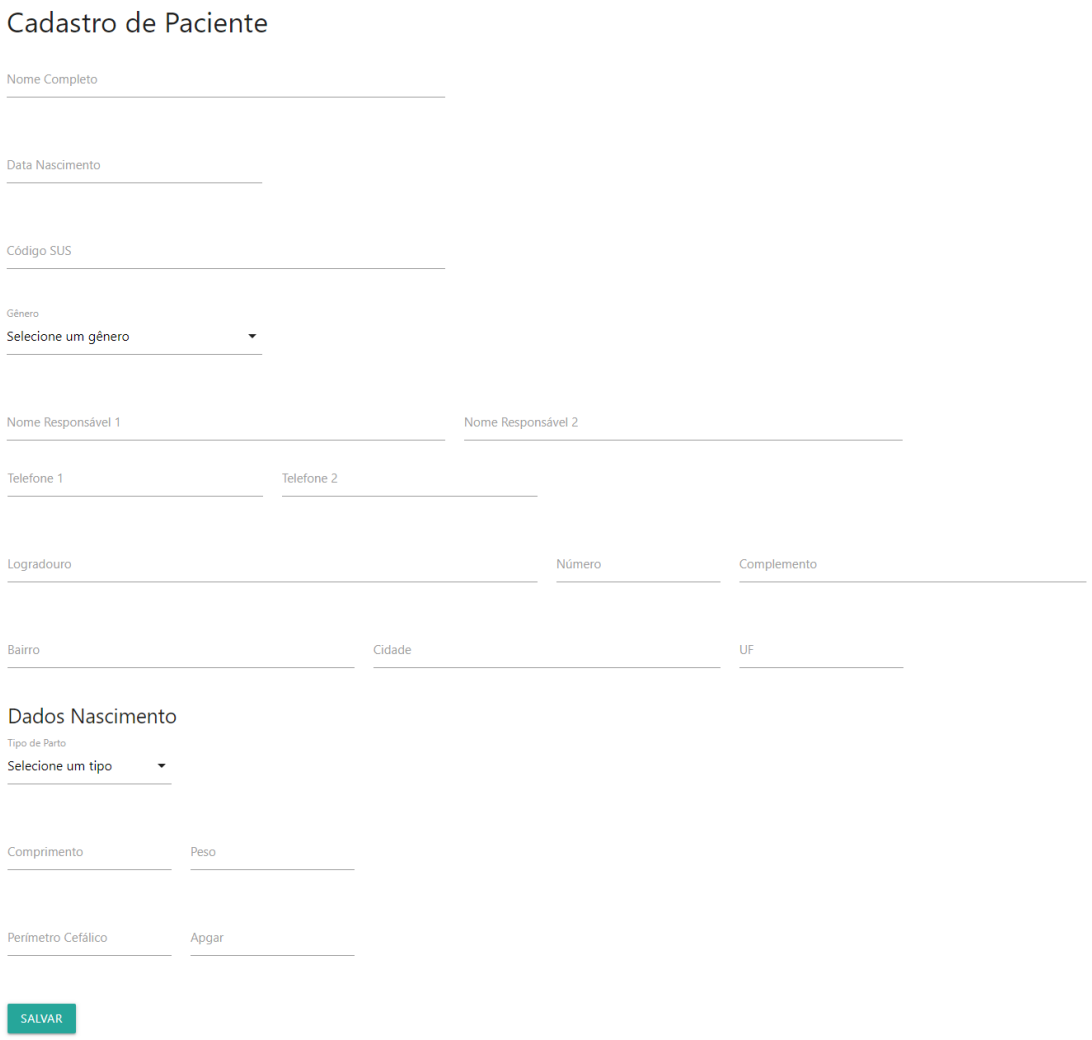


**Figura 7. Tela de Cadastro de Paciente**

Essa tela e as demais telas do sistemas foram criadas utilizando o JSF e o Materialize, inicialmente foi criado o formulário no arquivo xhtml, nesse incremento o arquivo foi nomeado de insertPaciente.xhtml onde os campos são criados e dispostos na tela e os componentes do Materialize são utilizados para dar formatos e um visual mais agradável aos campos. A requisição da tela através do botão “Salvar” (Figura 8) chama um *bean* gerenciado para que as regras, que nesse caso é para o cadastro do paciente, sejam executadas. O *ManagedBean* utilizado por essa tela foi nomeado de PacientManagedBean e nele tem o método createPacient() (Figura 9) que realiza o

cadastro do paciente, nesse método a persistência dos dados é realizada utilizando uma classe que implementa métodos utilizado o EntityManager (Figura 10) que persiste os objetos pelo Hibernate.

insertPaciente.xhtml

```
157                 <h:outputLabel for="tipoParto">Tipo de Parto</h:outputLabel>
158                 <p:message for="tipoParto" />
159             </div>
160         </div>
161         <div class="row">
162             <div class="input-field col s2">
163                 <h:inputText value="#{paciente.paciente.comprimentoNasc}"
164                     id="comprimentoNasc" maxlength="3" type="text" required="true"
165                     requiredMessage="Comprimento obrigatório" />
166                 <h:outputLabel for="comprimentoNasc">Comprimento</h:outputLabel>
167             </div>
168             <div class="input-field col s2">
169                 <h:inputText value="#{paciente.paciente.pesoNasc}" id="pesoNasc"
170                     type="text" required="true" requiredMessage="Peso obrigatório" />
171                 <h:outputLabel for="pesoNasc">Peso</h:outputLabel>
172             </div>
173         </div>
174         <div class="row">
175             <div class="input-field col s2">
176                 <h:inputText value="#{paciente.paciente.perimetroCefalicoNasc}"
177                     id="perimetroCefalicoNasc" type="text" required="true"
178                     requiredMessage="Perímetro Cefálico Obrigatório" />
179                 <h:outputLabel for="perimetroCefalicoNasc">Perímetro
180                     Cefálico</h:outputLabel>
181             </div>
182             <div class="input-field col s2">
183                 <h:inputText value="#{paciente.paciente.apgarNasc}" id="apgarNasc"
184                     maxlength="1" type="text" required="true"
185                     requiredMessage="Apgar obrigatório" />
186                 <h:outputLabel for="apgarNasc">Apgar</h:outputLabel>
187             </div>
188         </div>
189
190         <div class="row">
191             <div class="input-field col s2">
192                 <h:commandLink value="Salvar"
193                     actionListener="#{paciente.createPacient}"
194                     styleClass="btn waves-effect waves-light" update="pacienteForm" />
195             </div>
196         </div>
197     </h:form>
198 </h:body>
199 </html>
```

**Figura 8. Final do arquivo insertPaciente.xhtml que contém a chamada do bean na linha 193**

insertPaciente.xhtml | PacienteManagedBean.java

```
125
126     @Transactional
127     public void createPacient(ActionEvent evt) {
128
129         SimpleDateFormat formato = new SimpleDateFormat("dd-MM-yyyy");
130         Date dataFormatada = null;
131         try {
132             dataFormatada = formato.parse(dataNasc);
133         } catch (ParseException e1) {
134             log.error(e1);
135         }
136         this.paciente.setDataNasc(dataFormatada);
137
138         FacesContext context = FacesContext.getCurrentInstance();
139         try {
140
141             EntityManagerImpl entityManager = new EntityManagerImpl();
142             this.paciente.setEndereco(endereco);
143             entityManager.insert(this.paciente);
144
145             limparPaciente();
146             context.addMessage(null, new FacesMessage("Cadastro efetuado", "Cadastro efetuado"));
147             paciente = new Paciente();
148         } catch (Exception e) {
149             context.addMessage(null, new FacesMessage("Falha", "Não foi possível efetuar o cadastro"));
150             log.error(e);
151         }
152
153     }
```

**Figura 9. Método createPacient() do *bean* PacienteManagedBean**

insertPaciente.xhtml | PacienteManagedBean.java | EntityManagerImpl.java

```java
package control;

import java.util.List;

public class EntityManagerImpl {

    private static final String PERSISTENCE = "persistence";

    public void insert(Object object) {
        EntityManagerFactory factory = Persistence.createEntityManagerFactory(PERSISTENCE);
        EntityManager entityManager = factory.createEntityManager();
        entityManager.getTransaction().begin();
        entityManager.persist(object);
        entityManager.getTransaction().commit();
        entityManager.close();
        factory.close();

    }
```

**Figura 10. Método de *insert* da classe EntityManagerImpl que realiza a persistência do objeto no banco de dados pelo Hibernate**

### 3.2.3 Incremento 2 - Consulta de Pacientes

Esse incremento tem como objetivo listar todos os pacientes cadastrados com a possibilidade de realizar a busca pelo nome (Figura 11). A partir do paciente listado o usuário poderá realizar as outras operações do sistema, como remover e atualizar o cadastro do paciente que também foi desenvolvido nesse incremento.

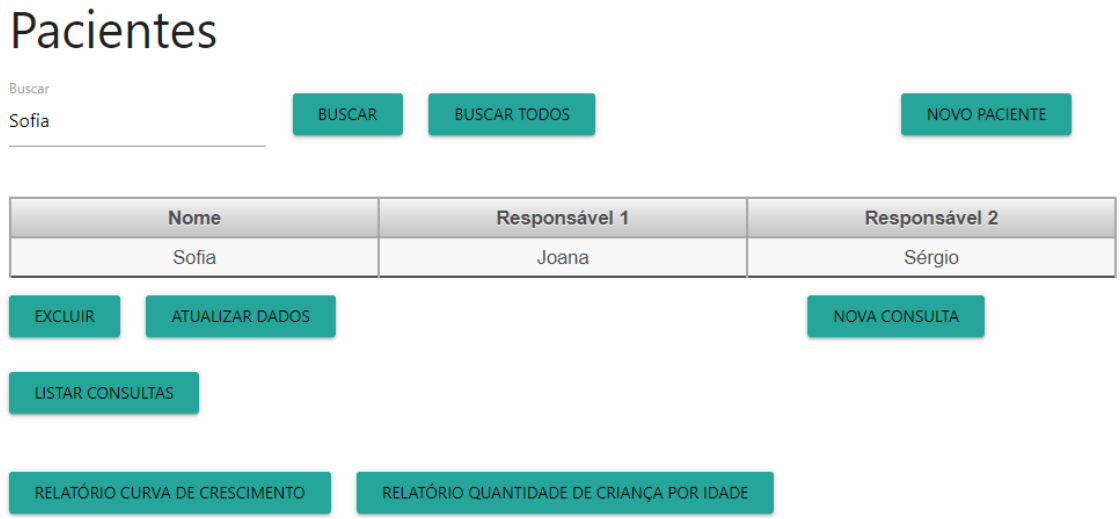


**Figura 11. Lista de Pacientes**

### 3.2.4 Incremento 3 - Registro da Consulta

Nesse incremento é realizado o registro da consulta de puericultura (Figura 12), nessa funcionalidade o usuário a partir do paciente listado poderá criar uma nova consulta e registrar os dados que são recolhidos durante o atendimento ao paciente. Os dados para cadastro da consulta foram levantados a partir da entrevista com o usuário e da caderneta do paciente mostrada na Figura 3.

Alguns campos são obrigatórios tais como data, peso atual, altura atual e perímetro cefálico atual, as opções disponíveis nos campos de sintomas são livres e deverão ser preenchidos caso o paciente tenha tido algum deles desde a última consulta. Esses dados são usados para controle de saúde sanitária da região e de desenvolvimento da criança.

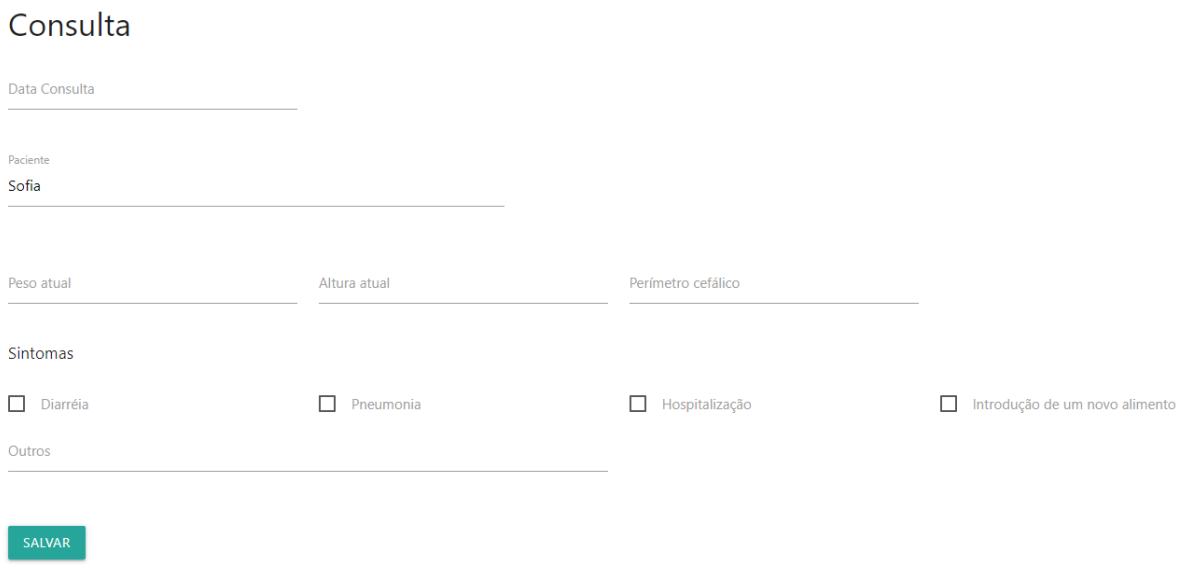


**Figura 12. Tela de registro de Consulta**

### 3.2.5 Incremento 4 - Lista de Consulta por Paciente

Esse incremento traz a funcionalidade de listar as consultas cadastradas de cada paciente (Figura 13), a partir de um paciente listado é possível trazer a lista das consultas realizadas (Figura 14).

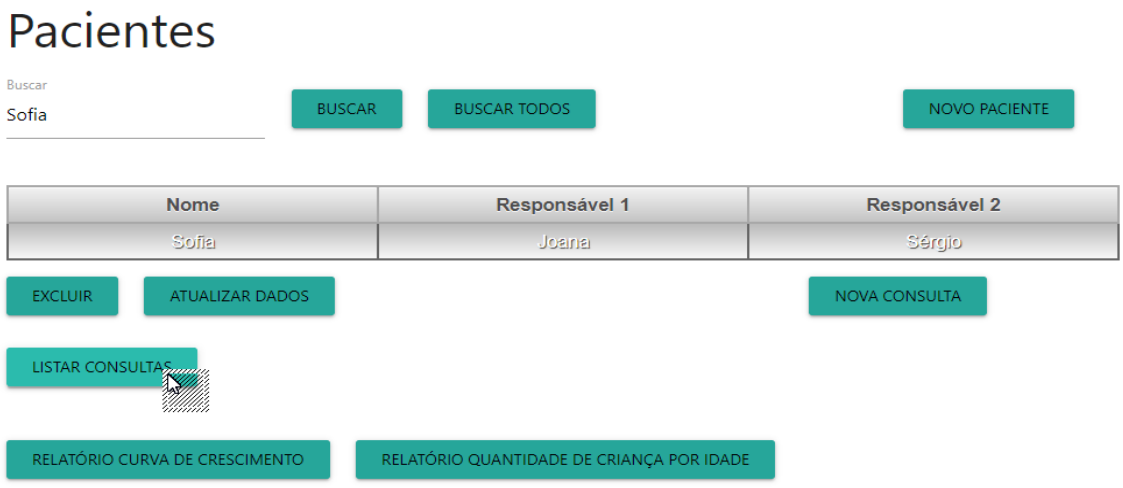


**Figura 13. Operação de listar consultas a partir do paciente listado**


**Figura 14. Lista de Consultas**

### 3.2.6 Incremento 5 - Geração de Relatórios

Esse incremento possui as funcionalidades de gerar relatório com a relação de Quantidade de Crianças por Idade (em meses) (Figura 15) e a curva de crescimento do paciente de acordo com os dados registrados na consulta (Figura 16).

O modelo dos relatórios foi criado na ferramenta Jaspersoft Studio utilizando templates em branco e o relatório foi gerado na aplicação utilizando as bibliotecas do JasperReport que já fornecem o método para gerar o relatório em formato PDF.

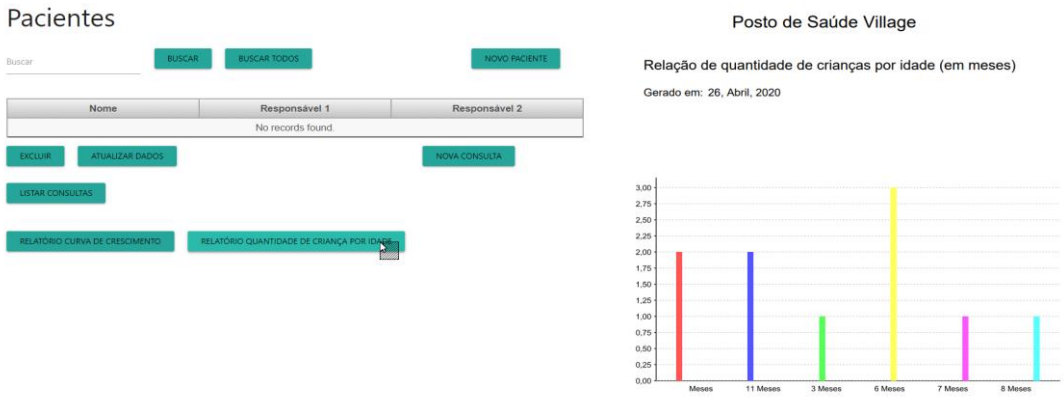


**Figura 15. Geração do Relatório de Relação de Quantidade de Crianças por Idade em Meses**

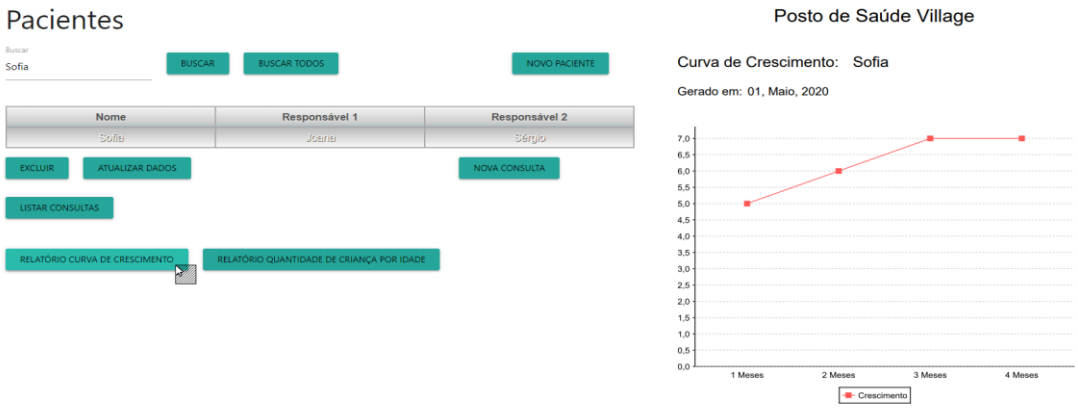


**Figura 16. Geração do Relatório Curva de Crescimento por Criança**

### 3.2.7 Incremento 6 - Login no Sistema

O último incremento desenvolvido foi o Login no Sistema (Figura 17), que traz a funcionalidade de um login com usuário administrador previamente cadastrado na base de dados e após realizar o login é redirecionado para a tela principal do sistema que é a Lista de Pacientes.


**Figura 17. Tela de login do sistema**

### 3.2.8 Testes do Incrementos

A cada finalização de cada incremento foi realizado teste funcional que segundo Pressman o Teste Funcional procura, entre outras coisas, mostrar que os requisitos funcionais do software são satisfeitos. (PRESSMAN, 2002)

## 4. Conclusão

O desenvolvimento do sistema proposto foi realizado com sucesso, todos os incrementos foram finalizados e testados. O desenvolvimento de um projeto para a área de atendimento da saúde cumpre seu objetivo de ser feito de maneira automática, podendo aumentar sua produtividade quando as pessoas estiverem treinadas e com maior afinidade com o sistema, e principalmente traz a disponibilidade e a facilidade de acesso à informação.

O conhecimento em tecnologias para desenvolvimento web foi imprescindível para a criação do software e as comunidades *online* fizeram de grande ajuda em momentos de dúvida ou de utilização de novos componentes, as matérias Engenharia de Software, Desenvolvimento Web, Banco de Dados, Tópicos Especiais e Linguagem de Programação foram fundamentais para construir o conhecimento necessário para esse projeto.

Por fim, a possibilidade de contribuir com a comunidade local e ajudar a melhorar o dia a dia dos profissionais com o conhecimento adquirido em sala de aula é gratificante e motivador para enxergar o propósito de se tornar um Analista de Sistemas, mostrando novas possibilidades de contribuir com essa e outras comunidades ao nosso redor. Recomenda-se como trabalhos futuros a criação de novos relatórios para que o posto de saúde seja capaz de levantar mais informações a partir dos dados coletados em consultas, o desenvolvimento de funcionalidades para o acompanhamento de vacinas dos pacientes que atualmente está presente somente na caderneta do paciente, o aprimoramento da

usabilidade do sistema que não foi explorada no desenvolvimento desse projeto e também a melhoria da segurança do sistema, como controle de sessão e criptografia.

Devido a pandemia do COVID19 não foi possível fazer a Implantação seguindo o modelo Incremental, ou seja, a entrega no posto de saúde e o *feedback* do usuário final deverá ser feito no futuro.

## Referências Bibliográficas


Apache Maven "Welcome to Apache Maven" Disponível em <http://maven.apache.org/> Acesso em: 29 Abril. 2020

DevMedia "Desenvolvendo com Hibernate" Disponível em <https://www.devmedia.com.br/desenvolvendo-com-hibernate/14756> Acesso em: 29 Abril. 2020

DevMedia. Gerando Relatórios com Jasper Report. Disponível em

<https://www.devmedia.com.br/gerando-relatorios-com-jasperreports/24798> Acesso em: 29 Abril. 2020

Hostinger "O que é GitHub e para que é usado?" Disponível em https://www.hostinger.com.br/tutoriais/o-que-github/ Acesso em: 29 Abril. 2020

ISO/IEC 9126 Standard for Information Technology, Software Product Evaluation – Quality Characteristics and Guidelines for their use, Geneve,1991.

Materialize (2020). Disponível em: <https://materializecss.com/about.html> Acesso em: 05 de Maio de 2020

Oracle (2020) "JavaServer Faces Technology Overview". Disponível em <https://www.oracle.com/technetwork/java/javaee/overview-140548.html> Acesso em: 05 de Maio de 2020

PostegreSQL Website "New to PostegreSQL?" Disponível em https://www.postgresql.org/ Acesso em: 29 Abril. 2020

PRESSMAN, Roger S. "Engenharia de Software", 6º Edição. Editora MCGrawHill: Porto Alegre, 2010.

Pressman, R. S. "Engenharia de Software", 5º Edição. Editora Mc Graw Hill: Rio de Janeiro, 2002.

Quiles, R. "O que é Puericultura?" Disponível em http://www.pediatraorienta.org.br/o-que-e-puericultura/ Acesso em: 28 Abril. 2020

# Documento Digitalizado Público

## Anexo I - Artigo Final

**Assunto:** Anexo I - Artigo Final
**Assinado por:** Daniela Marques
**Tipo do Documento:** Projeto
**Situação:** Finalizado
**Nível de Acesso:** Público
**Tipo do Conferência:** Documento Original

Documento assinado eletronicamente por:

- **Daniela Marques**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 22/05/2020 09:27:43.

Este documento foi armazenado no SUAP em 22/05/2020. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifsp.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

**Código Verificador:** 415488
**Código de Autenticação:** eaf763309e