Departamento de Polícia Rodoviária Federal

Projeto: Radio Digital

Absorção de Sistema

Revisão	Descrição	Autor	Data
1.0	Construção do documento	Israel Branco	06/04/2020

1 Sumário

2 Considerações iniciais

O presente documento tem por objetivo a verificação do processo de absorção de tecnologia do projeto Radio Digital disponibilizado no repositório https://git.prf/sistemas-nacionais/radio-digital. Este processo consiste em analisar as necessidades para preparação de ambiente local de desenvolvimento, impedimentos tecnológicos para continuidade da solução, análise estática de código e análise de vulnerabilidade de dependências.

E para atender ao objetivo expresso acima, compreende-se que o termo transferência de tecnologia é definido como um processo entre duas entidades sociais, em que o conhecimento tecnológico é adquirido, desenvolvido, utilizado e melhorado por meio da transferência de um ou mais componentes de tecnologia. Existe ainda a necessidade de inovar e evoluir com autonomia em busca de novas funcionalidades, de forma continuada, preservando os interesses originais encontrados no desenvolvimento do projeto.

3 Apresentação do cenário atual

A aplicação feita sob a premissa de operar sob protocolo HTTP servidor páginas web arquiteturalmente construída para trabalhar como aplicação monolítica em tecnologia PHP.

Possui frameworks, estrutura e organização que pregam boas práticas no desenvolvimento de aplicações WEB com tecnologia PHP:

- PHP 5.6 - linguagem de programação interpretada, base do projeto;
- Compose – gerenciador de dependências para linguagem de programação PHP;
- Láravel 5.4 – framework MVC open source para linguagem de programação PHP;
- Yarn – gerenciador de dependências para o NodeJS;
- Webpack – empacotador de arquivos javascript e css;
- Bootstrap 4 – framework para desenvolvimento de componentes de interface web;
- ~~PostgreSQL~~ – Banco de dados relacional;

A aplicação está baseada em um arquétipo já consolidado pela PRF, uma vantagem no ponto de vista arquitetural é que esta estrutura já foi testada em projetos que antecederam o mesmo. O arquétipo está disponível no repositório git https://git.prf/php/PRFWSClientPHP e sua documentação na WIKI https://git.prf/php/ArquetipoPHP/wikis/home.

3.1 Documentação existente

Código fonte: Disponibilidade completa no repositório GIT https://git.prf/sistemas-nacionais/radio-digital;

Documentação de requisitos: Não é disponível.

Documentação de implantação: Não é disponível;

Documentação para criação de ambiente: Não é disponível;

4 Análise técnica

Este tópico descreve a ferramenta do ponto de vista técnico, tanto nos aspectos de codificação, análise estática de código, análise de vulnerabilidade de dependências e particularidades de implementação.

4.1 SonarQube

Ferramenta utilizada para verificação de estática de código. Para esta análise não foram utilizadas as métricas de qualidade implantadas no SonarQube da DPRF, contudo foram utilizadas as regras padrões de análise da ferramenta. Os resultados foram os seguintes para as aplicações (tag sonda-nota-tecnica):

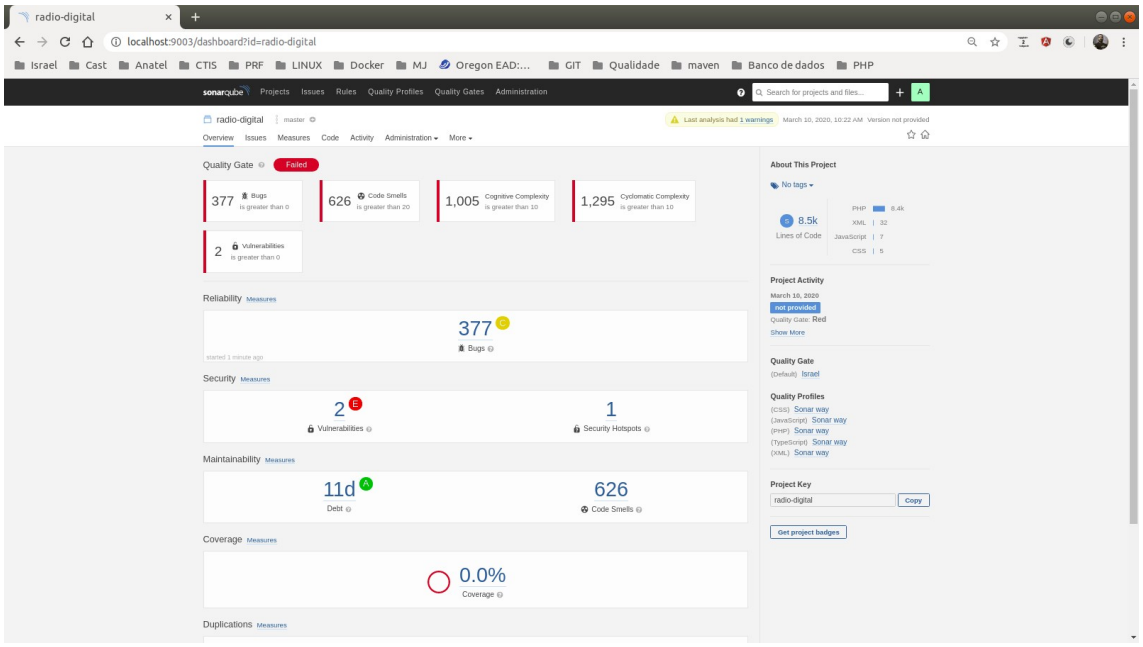

Figura 1: Análise estática de código

Nesta análise obtivemos os seguintes resultados:

- 377 bugs;
- 626 violações de más práticas;
- 1005 violações de complexidade cognitiva (dificuldade de entendimento de código);
- 1295 violações de complexidade ciclomática (complexidade de código);
- 2 vulnerabilidades;

4.2 OWASP ZAP

Ferramenta funciona como scanner de segurança, utilizada para realização de testes de vulnerabilidade de aplicações WEB. Atualmente trata-se de um dos projetos mais ativos na comunidade de software livre.

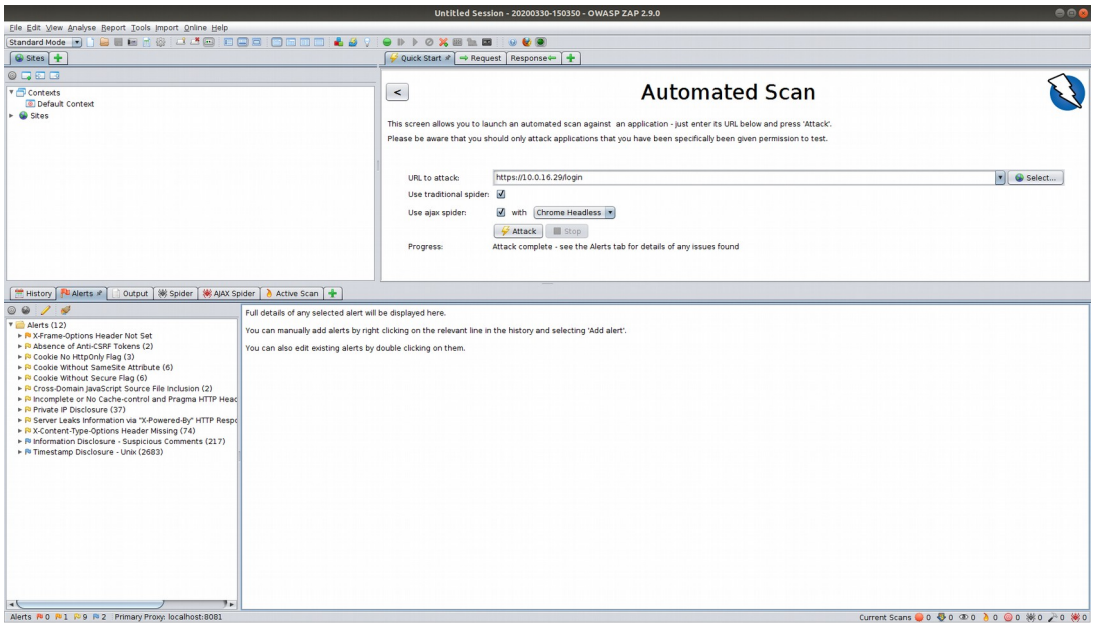

Figura 2: OWASP ZAP - Teste de intrusão

-
- 0 vulnerabilidade de severidade alta;
- 1 vulnerabilidade de severidade média;
- 9 vulnerabilidades de baixa média;
- 2 vulnerabilidades a nível informativo;

O relatório completo dos testes aplicados estão disponíveis no anexo I deste documento.

4.3 Banco de dados

Sistema gerenciador de banco de dados utilizado neste projeto foi o PostgreSQL, sua estrutura relacional possui apenas um único schema e 21 tabelas.

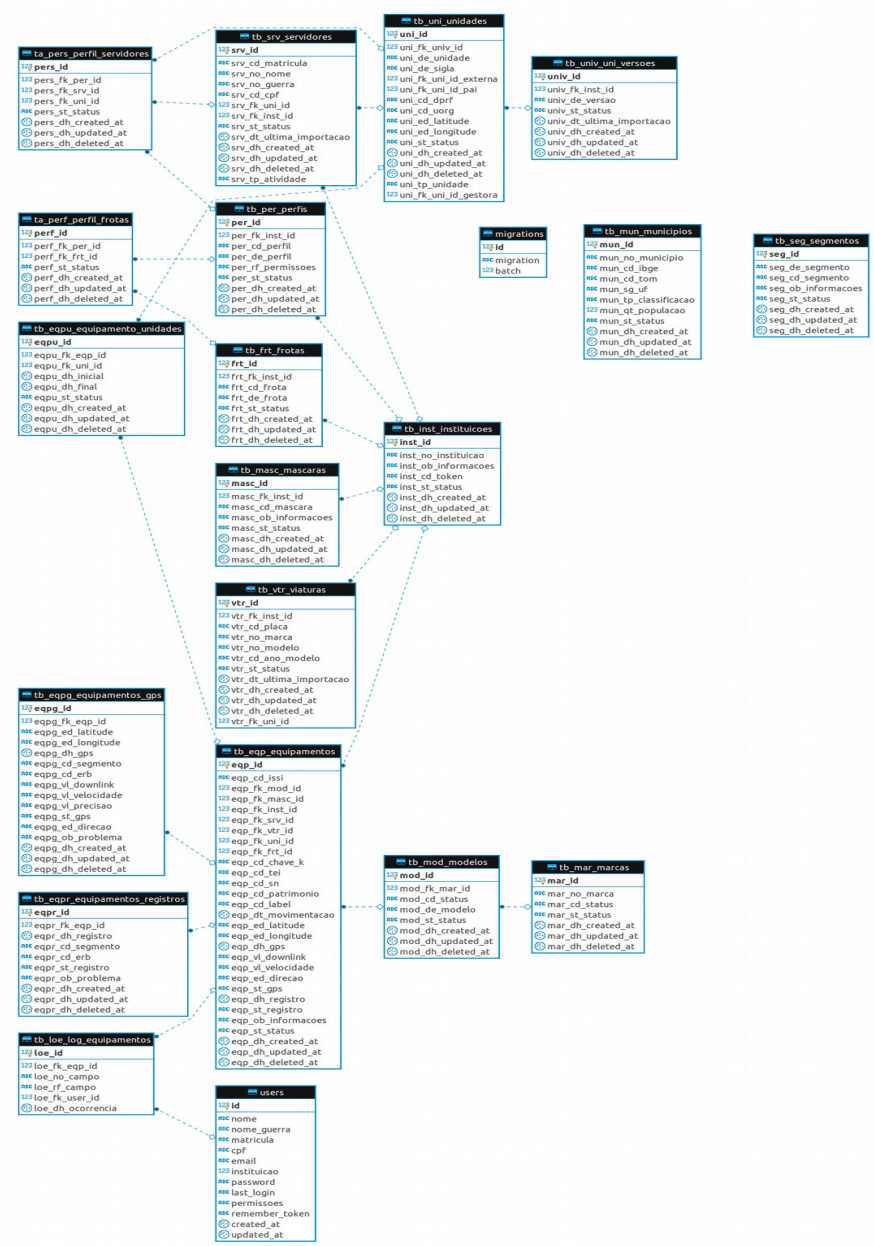

Figura 1: Modelo entidade relacionamento

4.4 Estrutura do projeto

A nomenclatura utilizada na organização nos pacotes está projetada e disposta de forma intuitiva e coesa. A componentização do monólito facilita o entendimento trazendo consigo diminuição de tempo na curva de aprendizado para novos ingressos a equipe de desenvolvimento do projeto.

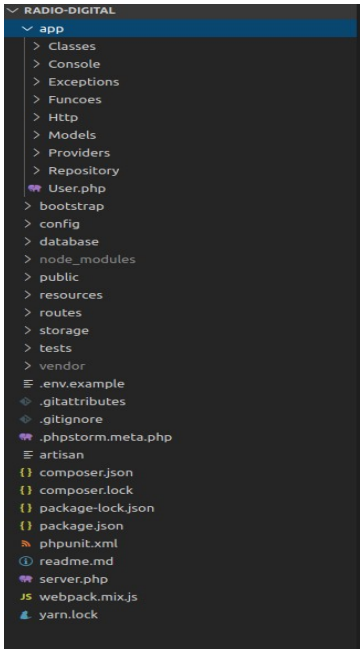

Figura 3: Estrutura do projeto

4.5 Manutenibilidade de código

Os relatórios apresentados pela ferramenta SonarQube demonstram uma série de vícios adotados durante o processo de construção do software e alinhado a estes vícios, a inexistência de cobertura de testes de unidade que trazem a dificuldade no processo de refactoring da aplicação, uma vez que não há condições de mensurar impactos durante o processo de manutenção corretiva/adaptativa.

A alta complexidade ciclomática e a falta de artefatos de testes de unidade dificultam o processo de refactoring, a ilustração que seguem demonstram o cenário apontado (OBS: a característica apresentada é utilizada de forma recorrente em diversos momentos do código).

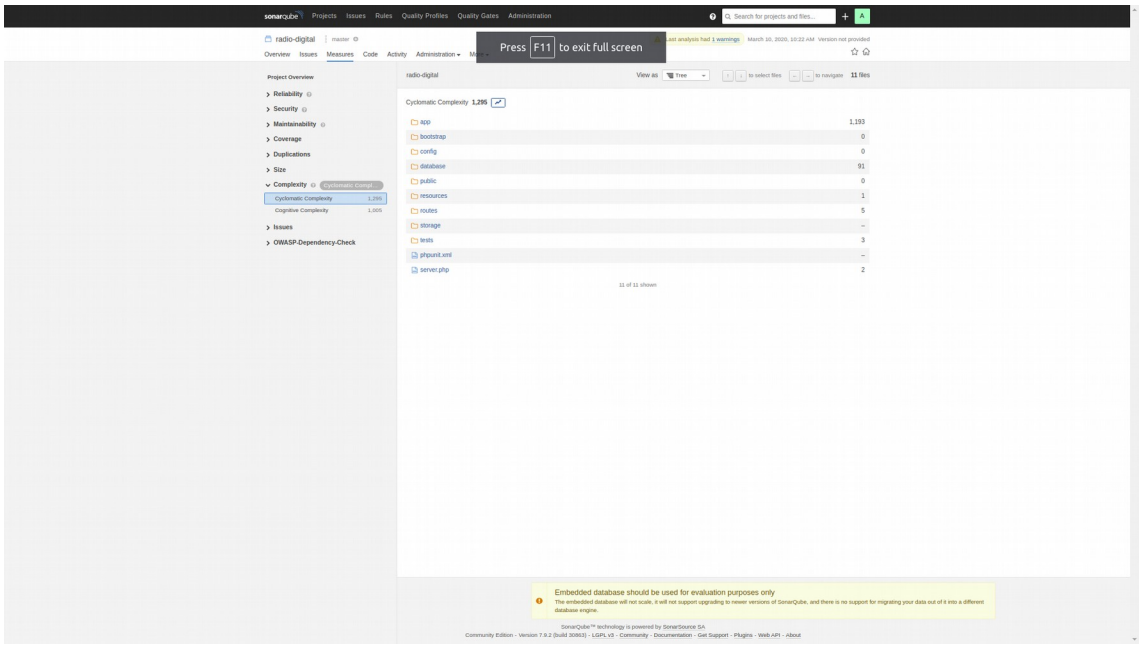

Figura 4: Complexidade ciclomática por pacotes

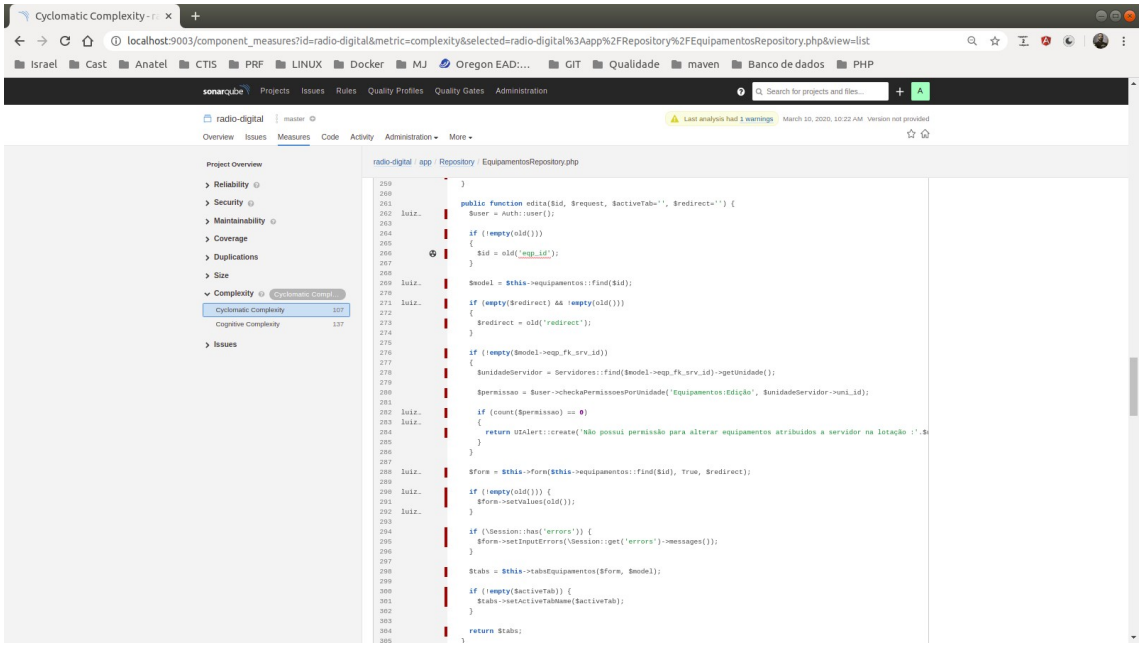

Figura 5: Complexidade ciclomática EquipamentoRepository.php

4.6 Confiabilidade

Não há evidências que demonstre a existência de tratativas de controle transacional na aplicação, este tratamento segue as boas práticas de desenvolvimento de aplicações. A falta deste controle transacional impede a garantia das propriedades ACID do SGBD.

A manutenção da consistência de dados é algo fortemente desejado, contudo esta não garante toda a confiabilidade da solução. A quantidade elevada de bugs, vulnerabilidades no código encontradas nos relatórios apresentados trazem riscos a confiabilidade da ferramenta.

4.7 Performance e estabilidade

Não foi analisado o funcionamento da aplicação para avaliar demais requisitos não funcionais, recomenda-se a utilização de ferramentas de APM para mensurar performance e recursos de

máquina utilizados.

4.8 Padrões de codificação

Embora haja boa segregação nos pacotes da aplicação, é perceptível a falta de coesão nos mesmos. Não há camada intermediária entre os controladores e a camada de persistência e uitas regras negociais são tratadas dentro das classes de modelo e das classes de persistência, sendo que ambas possuem acesso a banco de dados.

Outra característica encontrada no código é que toda tratativa de gestão de autorização está sendo tratada na camada de persistência, não uma um interceptador central para realização desta atividade.

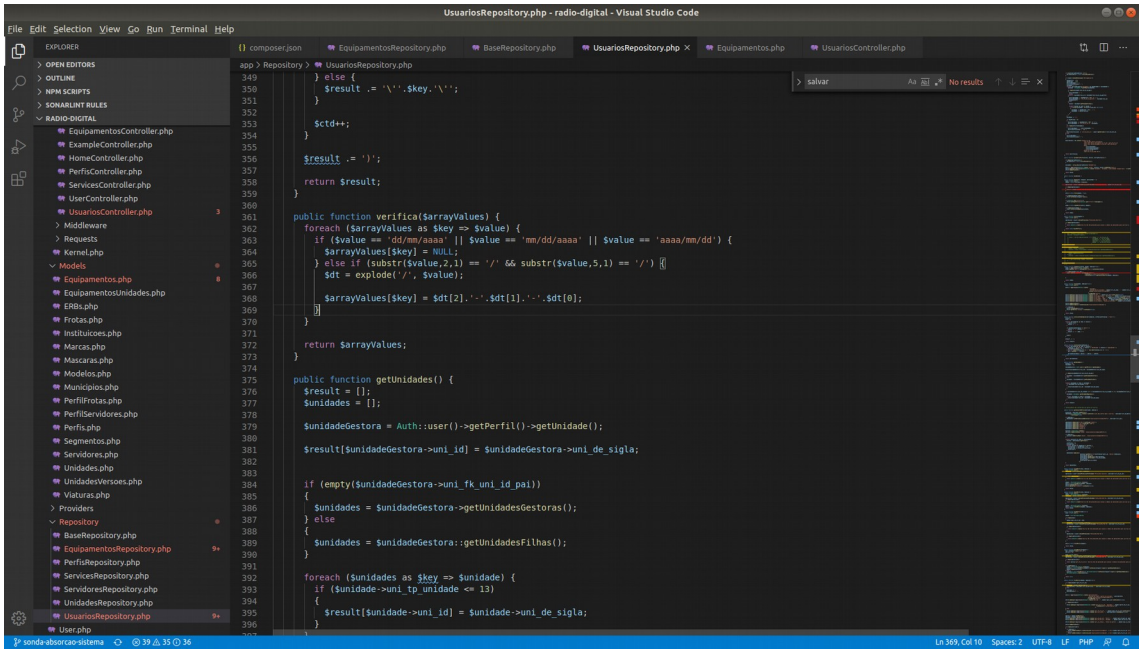

Figura 6: Aplicação de regras na camada de acesso a dados

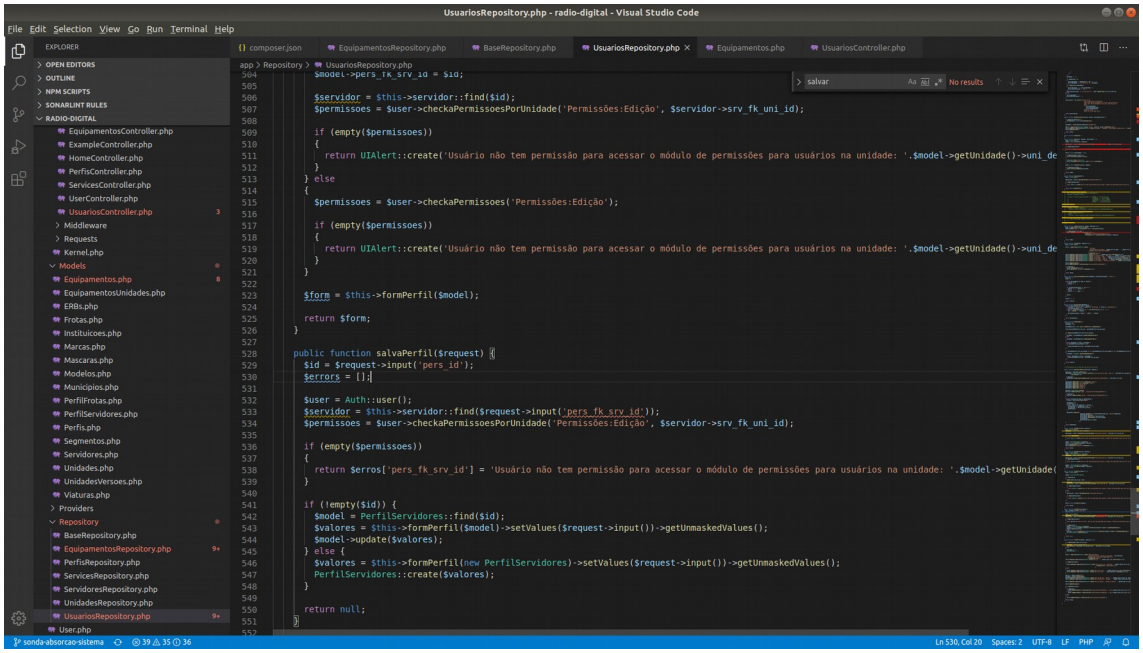

Figura 7: Verificação de autorização na camada de persistência

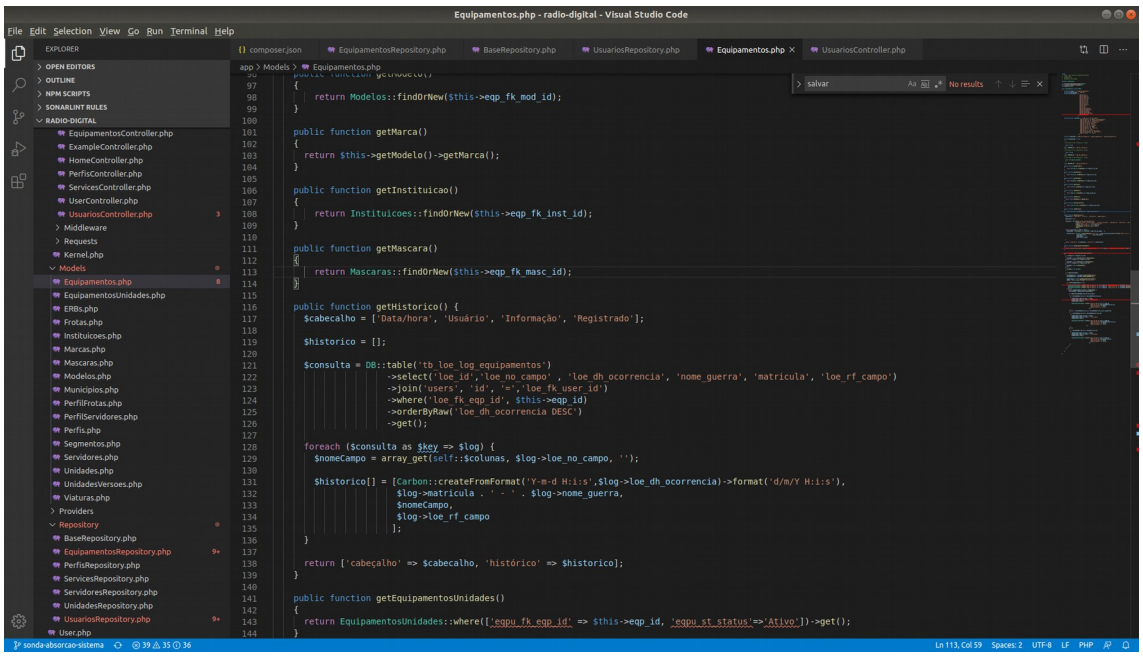

Figura 8: Classes modelo efetuando chamdas ao banco de dados

4.9 Padrões de interface

A página inicial da aplicação contempla itens oriundos do template utilizado e que não estão sendo utilizados, sendo elas o menu lateral esquerdo (com componente show/hide) e os ícones no menu superior direito.

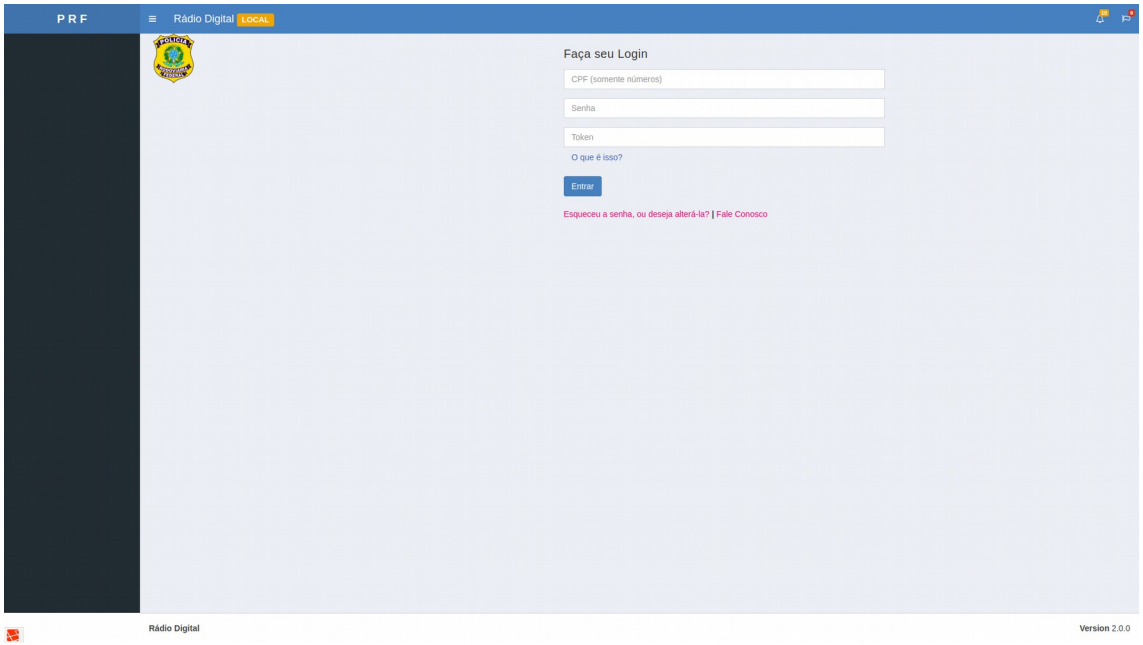

Figura 9: Página inicial - login

5 Recomendações

É altamente recomendado que seja efetuado refactoring de código dos bugs e vulnerabilidades de código apontadas pelo SonarQube , estas atividades certamente trarão maior confiabilidade a ferramenta e estabilidade em seu uso. Para os demais itens apontados pela ferramenta SonarQube durante o processo de análise de código são altamente desejáveis, contudo este processo de ajuste de código é moroso e trás consigo risco em potencial e está diretamente aliado a falta de cobertura de testes de unidade.

O relatório de análise dos testes de intrusão não reportam grande vulnerabilidade do sistema, contudo recomenda-se que os ajustes sejam providenciados. Recomenda-se também que a aplicação seja disponibilizada sob o protocolo HTTPS que utilize certificado válido.

Recomenda-se a implantação de ferramentas de APM para que sejam criadas métricas e alarmes que auxiliem na continuidade do serviço em ambiente produtivo(monitoramento de processamento e memória por exemplo), tendo em vista que este tipo de ferramenta fornece mecanismos para determinarmos o comportamento da solução (auxiliam no refactoring de código) e também subsidia para o correto dimensionamento da infraestrutura.

Recomenda-se a alteração do arquivo Readme do projeto para que contenha os dados específicos do projeto , atualmente este arquivo é uma cópia do arquivo Readme do arquétipo.