



Departamento de Polícia Rodoviária Federal

Projeto: Sistema de Controle de Frequência

Nota Técnica

PRF	Controle de Frequência	
------------	-------------------------------	--

Revisão	Descrição	Autor	Data
1.0	Construção do documento	Israel Branco	30/09/2020

1 Sumário

2 Considerações iniciais.....	4
3 Apresentação do cenário atual.....	5
3.1 Documentação existente.....	5
3.2 Diagrama de componentes.....	5
4 Análise técnica.....	6
4.1 SonarQube.....	6
4.2 OWASP ZAP.....	7
4.3 Banco de dados.....	9
4.4 Estrutura do projeto.....	10
4.5 Manutenibilidade de código.....	11
4.6 Confiabilidade.....	12
4.7 Performance e estabilidade.....	12
4.8 Padrões de codificação.....	12
4.9 Padrões de interface.....	13
5 Conclusão.....	14

2 Considerações iniciais

O presente documento tem por objetivo reportar a análise efetuada no sistema de Controle de Frequência denominado daqui por diante puramente de Controle de Frequência. 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.

Como insumo para esta análise a tag <https://git.prf/sistemas-nacionais/anprf/frequencia/-/tags/tag-ctis-nota-tecnica> foi gerada a partir da master branch em 30/09/2020.

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 5 com banco de dados MySQL.

Sua arquitetura não está baseada em nenhum arquétipo de framework de mercado, a mesma também não está baseada no template visual da PRF. Sua estrutura basicamente utiliza tecnologia PHP, HTML, CSS.

3.1 Documentação existente

Não há documentos disponíveis tais como, documento de arquitetura, documento comercial, manual do usuário, manual do desenvolvedor e manual de implantação.

3.2 Diagrama de componentes

A ferramenta não utiliza composição de componente, todas as páginas são independentes e não há utilização de padrões de codificação.

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):

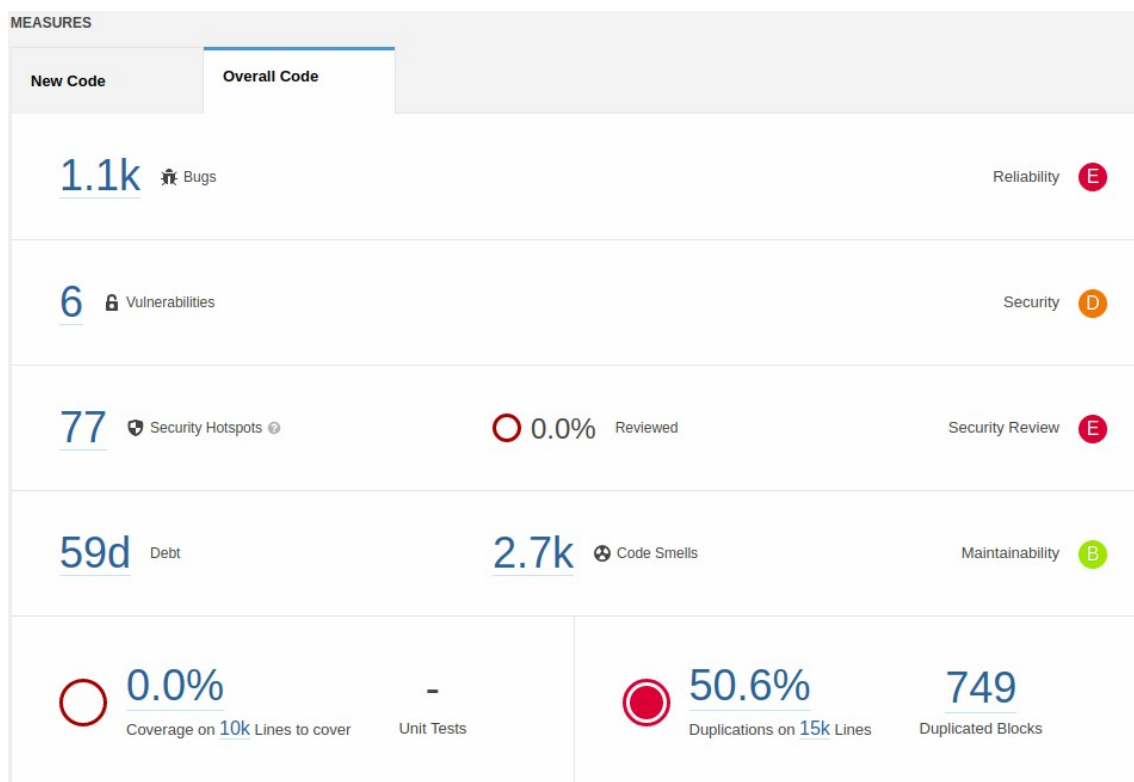


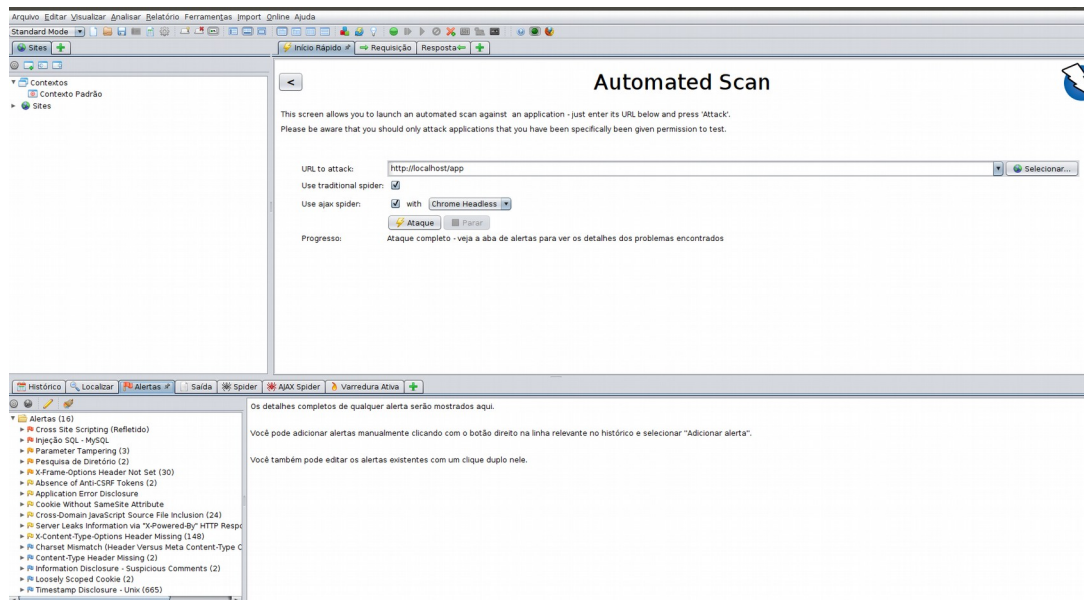
Figura 1: Análise estática de código

Nesta análise obtivemos os seguintes resultados:

- 1.100 mil bugs;
- 6 vulnerabilidades de código;
- 77 violações de segurança;
- ~3 mil violações de código ruim (complexidade cognitiva , complexidade ciclomática e débito técnico);;
- 50.6% de duplicidade de código;

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.



PRF	Controle de Frequência	
------------	-------------------------------	--

-
- 0 vulnerabilidade de severidade alta;
- 7 vulnerabilidade de severidade média;
- 17 vulnerabilidades de baixa média;
- 12 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 MySQL, sua estrutura relacional possui apenas um único schema contendo 35 tabelas.

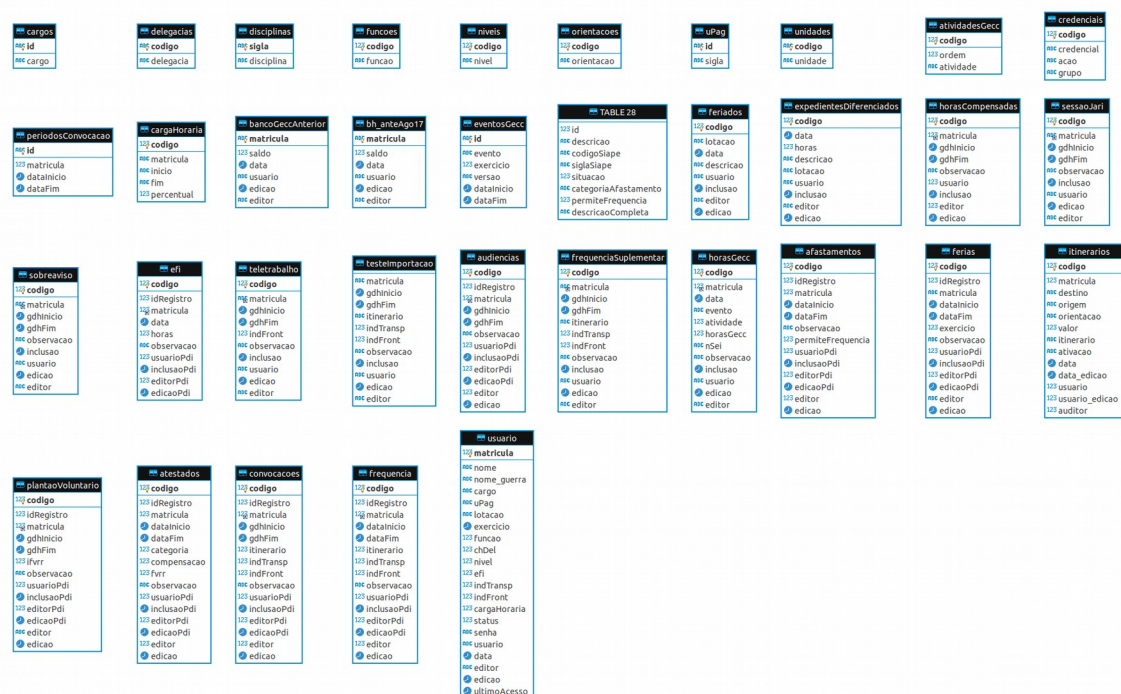


Figura 2: MER - schema frequência

4.4 Estrutura do projeto

Não estrutura organizada por pacotes, todos os arquivos de código fonte estão dispostos na raiz do projeto. Há 3 subdiretórios que possuem respectivamente arquivos de stilo CSS, imagens e por fim arquivos php contendo as constantes e conexão com banco de dados.

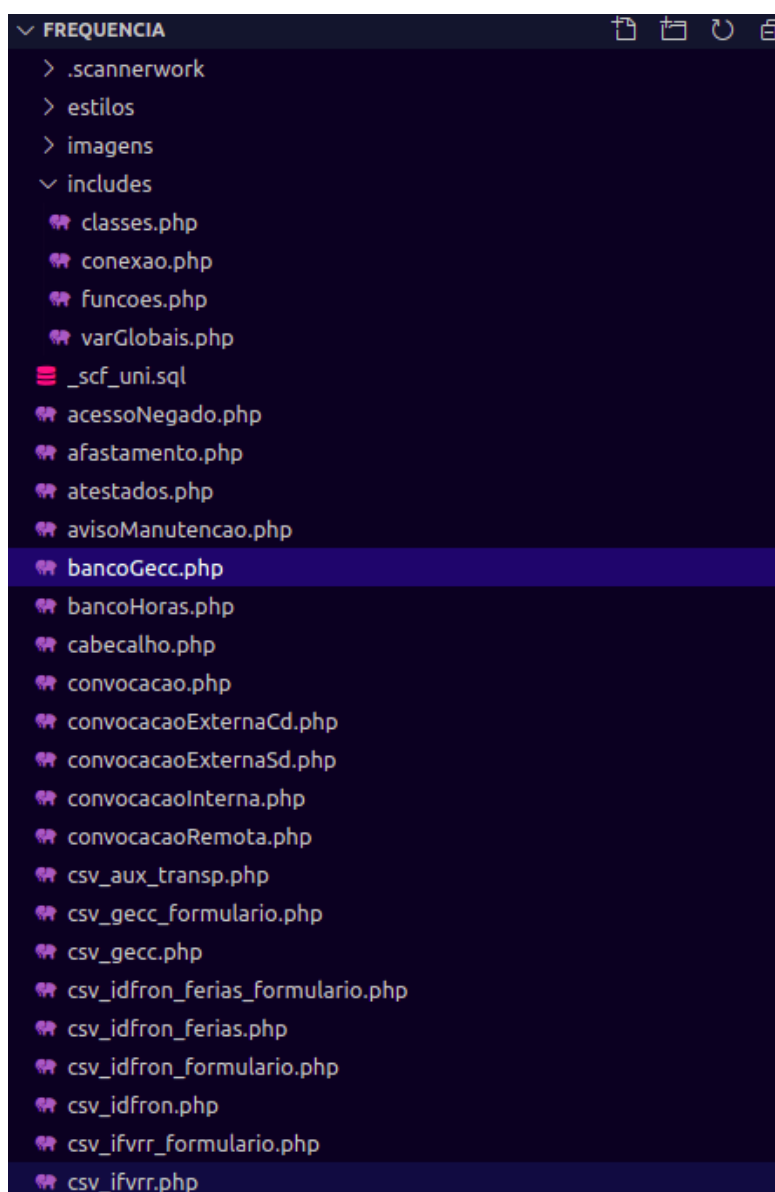


Figura 3: Estrutura do projeto

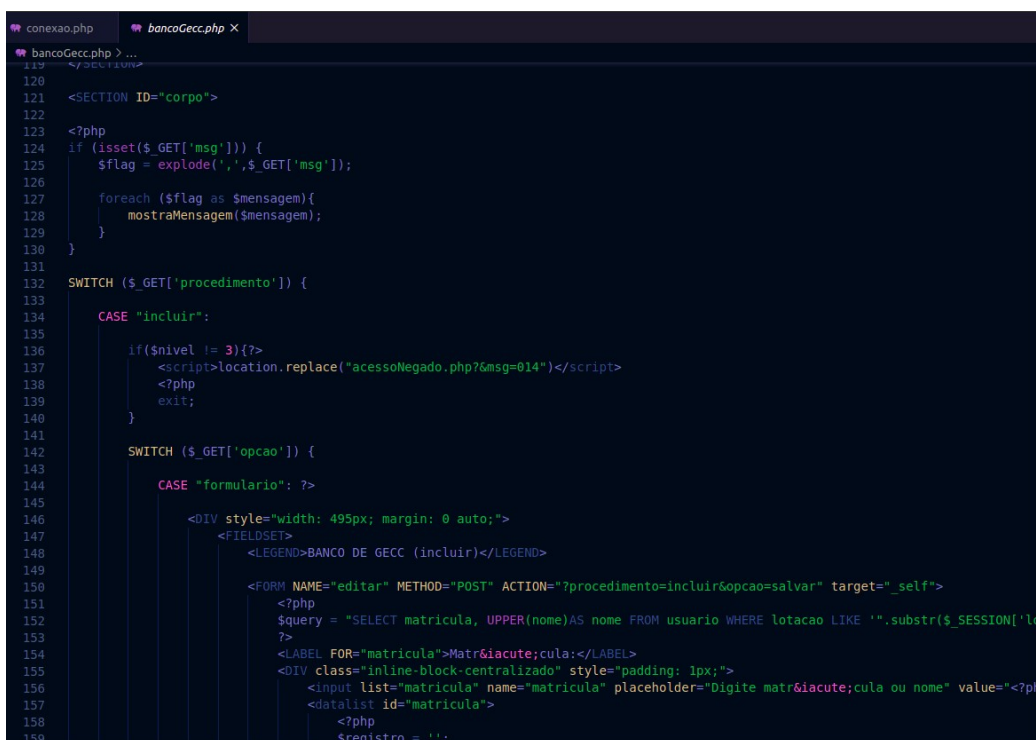
4.5 Manutenibilidade de código

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.

A aplicação não apresenta segregação entre o conteúdo HTML, JavaScript e PHP, o que torna os arquivos longos, sem potencial de reúso e com programação macarrônica.

Não há segregação de camadas, não há utilização de Designer Patterns, não há padrão para nomenclatura e não há utilização de orientação a objetos.

A figura a seguir demonstra a característica de codificação encontrada em toda a aplicação.



```

120
121 <SECTION ID="corpo">
122
123 <?php
124 if (isset($_GET['msg'])) {
125     $flag = explode(',', $_GET['msg']);
126
127     foreach ($flag as $mensagem) {
128         mostraMensagem($mensagem);
129     }
130 }
131
132 SWITCH ($_GET['procedimento']) {
133
134     CASE "incluir":
135
136         if ($nivel != 3) {
137             <script>location.replace("acessoNegado.php?msg=014")</script>
138             <?php
139                 exit;
140             >
141
142         SWITCH ($_GET['opcao']) {
143
144             CASE "formulario": ?>
145
146                 <DIV style="width: 495px; margin: 0 auto;">
147                     <FIELDSET>
148                         <LEGEND>BANCO DE GECC (incluir)</LEGEND>
149
150                         <FORM NAME="editar" METHOD="POST" ACTION="?procedimento=incluir&opcao=salvar" target="_self">
151                             <?php
152                                 $query = "SELECT matricula, UPPER(nome)AS nome FROM usuario WHERE lotacao LIKE '".substr($_SESSION['lo
153                                     ?>
154                                 <LABEL FOR="matricula">Matrícula;</LABEL>
155                                 <DIV class="inline-block-centralizado" style="padding: 1px;">
156                                     <input list="matricula" name="matricula" placeholder="Digite matrícula ou nome" value="<?ph
157                                     <datalist id="matricula">
158                                     <?php
159                                     $registro = '';

```

Figura 4: Padrão de construção da aplicação - trecho de código

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. Também não há uma modelagem de dados que seja consistente tendo em vista há inexistência de relacionamentos entre as tabelas.

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.

Toda estrutura de geração de código SQL sugere vulnerabilidade por ataque de injeção de dependência.

O algoritmo MD5 utilizado para encriptação das senhas é um algoritmo fraco e vulnerável.

4.7 Performance e estabilidade

Não foi analisado o funcionamento da aplicação para avaliar demais requisitos não funcionais, contudo foi percebido a falta de tratativa para paginação das consultas. Fato este que degrada a performance da aplicação.

4.8 Padrões de codificação

Não há padrão de codificação consistente, não há utilização de uma arquitetura de referência, orientação a objetos,

PRF	Controle de Frequência	
------------	-------------------------------	--

segregação de tecnologias, framework ou qualquer outra boa prática na construção de aplicações Web utilizando tecnologia PHP.

4.9 Padrões de interface

A aplicação não utiliza padrões de interface visual da PRF.

SCF
Sistema de Controle de Frequência
Usuário não autenticado

Credenciais de acesso

Login:

Senha:

Figura 5: Tela de login

SCF
Sistema de Controle de Frequência
Usuário: 0123456 123 [sair]

DEA (Relatório)

Matricula:

Início (data):

Término (data):

Evento:

Notice

: Undefined index: indTransp in /opt/lampp/htdocs/app/horaGecc.php on line 82

Notice

: Undefined index: indFront in /opt/lampp/htdocs/app/horaGecc.php on line 83

Notice

: Undefined index: efi in /opt/lampp/htdocs/app/horaGecc.php on line 84

Menu
Incluir ▾
Consultar ▾
Relatório ▾
Senha ▾

Figura 6: Erros de renderização

5 Conclusão

A ferramenta apresenta uma série de vícios em sua concepção, sendo eles:

- Falta de arquitetura de referência;
- Falta de segregação de tecnologias;
- Falta de utilização do padrão MVC;
- Falta de modelagem consistente de dados;
- Utilização de criptografia fraca para o armazenamento de senhas;
- Falta de controle transacional na manipulação de dados;
- Falta de padrão visual da PRF;
- A exportação dos relatórios não funciona;
- Falta de utilização de encode;
- Falta de paginação nas consultas;
- Falta de gerenciamento de perfis de acesso;

Diante ao exposto não recomenda-se a utilização/continuidade desta no ambiente da PRF, uma vez que o esforço empreendido para sanar os diversos problemas apresentados são maiores do que a construção de uma nova ferramenta que opere dentro dos padrões da PRF.