SYSTEM NAME

Static and Dynamic Application Diagnostics

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Autor** |
| 31/03/2017 | 1.0 | Document elaboration | Pablo Santiago Sánchez |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Summary**

[1. Introduction 4](#_Toc755420559)

[2. General view of the analysis process 5](#_Toc1274966179)

[2.1. Tools used 5](#_Toc1890717860)

[3. Non-functional requirement analysis 6](#_Toc971481239)

[3.1. Security 6](#_Toc1196894217)

[3.2. Usability, Responsivity, accessibility and on-line manual / user documentation 6](#_Toc1067945511)

[4. Static Analysis 7](#_Toc246832995)

[4.1. Total Technical Debt 7](#_Toc362786690)

[4.2. Confiability 7](#_Toc1847510876)

[4.3. Security 7](#_Toc662521186)

[4.4. Maintainability 7](#_Toc1411015423)

[4.5. Duplicity 7](#_Toc159550637)

[4.6. Size and Complexity 7](#_Toc612961892)

[4.7. Code documentation 7](#_Toc1586745189)

[4.8. Good practices violations 7](#_Toc767398398)

[5. Software Architecture, Logic and Physic Code Organization 8](#_Toc46094090)

[5.1. Framework in use 8](#_Toc2058666765)

[5.2. Development mechanisms presents in the framework 8](#_Toc1279700825)

[5.2.1. Database reverse-engineer 8](#_Toc141942416)

[5.2.2. Scaffolding generation 8](#_Toc1496111603)

[5.3. Application Health 8](#_Toc687556052)

[5.3.1. Layered code 8](#_Toc1487208641)

[5.3.2. Modular functions 8](#_Toc2028575843)

[5.3.3. Vendor separation 8](#_Toc946836210)

[5.3.4. Cache Mechanisms 8](#_Toc1422364110)

[5.3.5. Session use (size and information stored) 8](#_Toc7030941)

[6. Analysis Resume Panel 9](#_Toc1267832415)

[7. Conclusion 10](#_Toc699092885)

[7.1. Good news: 10](#_Toc847152272)

[7.2. Bad news: 10](#_Toc81440143)

[7.3. Recommended improvements: 10](#_Toc1573509625)

[7.3.1. Framework: 10](#_Toc1602572831)

[7.3.2. Code organization: 10](#_Toc1356406322)

[7.3.3. Code implementation: 10](#_Toc1316743837)

[7.3.4. Usability, responsivity, accessibility and on-line manual / user documentation: 10](#_Toc426570422)

[7.3.5. Deployment process: 10](#_Toc405816891)

[7.3.6. Database migrations 10](#_Toc237205700)

[7.3.7. Unit tests 10](#_Toc673403417)

[8. References 11](#_Toc768603582)

# 1. Introduction

# 2. General view of the analysis process

## 2.1. Tools used

# 3. Non-functional requirement analysis

## 3.1. Security

3.1.1. HTTPS access only

3.1.2. Access control to features and system resources

3.1.3. Auditing mechanisms

3.1.4. Runtime Errors Handling

3.1.5. Environment Setup Overwrite (ini\_set)

3.1.6. Code Injection (eval, exec, system calls)

3.1.7. SQL Injection (non-prepared statements)

## 3.2. Usability, Responsivity, accessibility and on-line manual / user documentation

3.2.1. Usability / User Experience

3.2.2. Responsivity

3.2.3. Accessibility

3.2.4. On-line Manual / User Documentation

# 4. Static Analysis

## 4.1. Total Technical Debt

## 4.2. Confiability

## 4.3. Security

## 4.4. Maintainability

## 4.5. Duplicity

## 4.6. Size and Complexity

## 4.7. Code documentation

## 4.8. Good practices violations

# 5. Software Architecture, Logic and Physic Code Organization

## 5.1. Framework in use

## 5.2. Development mechanisms presents in the framework

### 5.2.1. Database reverse-engineer

### 5.2.2. Scaffolding generation

## 5.3. Application Health

### 5.3.1. Layered code

### 5.3.2. Modular functions

### 5.3.3. Vendor separation

### 5.3.4. Cache Mechanisms

### 5.3.5. Session use (size and information stored)

# 6. Analysis Resume Panel

|  |  |  |  |
| --- | --- | --- | --- |
| Group | Item | Result | Recommendation |
| Security | HTTPS access only | High priority |  |
| Access control to features and system resources | Normal priority |  |
| Auditing mechanisms | Low priority |  |
| Runtime Errors Handling | OK |  |
| Environment Setup Overwrite (ini\_set) |  |  |
| Code Injection (eval, exec, system calls) |  |  |
| SQL Injection (non-prepared statements) |  |  |
| Usability, responsivity, accessibility and on-line manual / user documentation | Usability / User Experience |  |  |
| Responsivity |  |  |
| Accessibility |  |  |
| On-line Manual / User Documentation |  |  |
| Static Analysis | Total Technical Debt |  |  |
| Confiability |  |  |
| Security |  |  |
| Maintainability |  |  |
| Duplicity |  |  |
| Size and Complexity |  |  |
| Code documentation |  |  |
| Good practices violations |  |  |
| Software Architecture, Logic and Physic Code Organization | Framework in use |  |  |
| Development mechanisms - Database reverse-engineer |  |  |
| Development mechanisms - Scaffolding generation |  |  |
| Application Health - Layered code |  |  |
| Application Health - Modular functions |  |  |
| Application Health - Vendor separation |  |  |
| Application Health - Cache Mechanisms |  |  |
| Application Health - Session use |  |  |

# 7. Conclusion

## 7.1. Good news:

## 7.2. Bad news:

## 7.3. Recommended improvements:

### 7.3.1. Framework:

### 7.3.2. Code organization:

### 7.3.3. Code implementation:

### 7.3.4. Usability, responsivity, accessibility and on-line manual / user documentation:

### 7.3.5. Deployment process:

### 7.3.6. Database migrations

### 7.3.7. Unit tests

# 8. References