

Relatório Técnico de Desenvolvimento de Software

Lamberjack’s ORM

*Diego Francklin Martins dos Santos*

Sorocaba

Junho – 2018



Relatório Técnico de Desenvolvimento de Software

Lamberjack’s ORM

Diego Francklin Martins dos Santos

Prof. Cristiane Palomar Mercado - Orientador

Sorocaba

Junho – 2018

**Dedicatória**

Dedico este trabalho aos meus pais, amigos e esposa que sempre me ajudaram e incentivaram a seguir adiante e vencer todas as barreiras e desafios.

**Agradecimento**

Agradeço a todos que de uma forma ou de outra colaboraram para com este trabalho.

**Sumário**

[Resumo 1](#_Toc508550222)

[1. Introdução 2](#_Toc508550223)

[2. Análise da Situação Atual 3](#_Toc508550224)

[2.1. Descrição da Situação Atual 3](#_Toc508550225)

[2.2. Objetivo 3](#_Toc508550226)

[3. Análise de Requisitos 4](#_Toc508550227)

[3.1. Situação Proposta 4](#_Toc508550228)

[3.2. Requisitos Funcionais 4](#_Toc508550229)

[3.2.1. Diagrama de Caso de Uso de Alto Nível 4](#_Toc508550230)

[3.2.2. Diagrama de Caso de Uso de Baixo Nível 4](#_Toc508550231)

[3.3. Requisitos Não-Funcionais 4](#_Toc508550232)

[3.3.1. Requisitos de Software 4](#_Toc508550233)

[4. Projeto Detalhado do Software 5](#_Toc508550234)

[4.1. Arquitetura da Aplicação 5](#_Toc508550235)

[4.1.1. Alguma coisa aqui 5](#_Toc508550236)

[4.2. Tecnologias a serem usadas 5](#_Toc508550237)

[4.2.1. Alguma coisa aqui 5](#_Toc508550238)

[4.3. Diagrama de Classe 5](#_Toc508550239)

[4.3.1. Requisitos de Software 5](#_Toc508550240)

[4.4. Diagrama de Sequência 5](#_Toc508550241)

[4.4.1. Alguma coisa aqui 5](#_Toc508550242)

[5. Codificação 6](#_Toc508550243)

[6. Análise dos Resultados 7](#_Toc508550244)

[Referências 8](#_Toc508550245)

[Glossário 9](#_Toc508550246)

**Lista de Figuras**

[Figura 1: Estrutura do Framework 4](#_Toc508547902)

**Lista de Tabelas**

**Lamberjack’s ORM**

# Resumo

Durante a criação e manutenção de projetos é investido um grande esforço e uma grande quantidade de tempo para analisar, criar e integrar bancos de dados à aplicação. E geralmente, torna-se muito trabalhoso migrar para outros bancos de dados e com grande potencial de se tornar um verdadeiro pesadelo.

O ORM (*Object-Relational Mapping*) é uma técnica que visa o mapeamento entre mundo orientado à objetos (os nossos modelos) e o mundo relacional (o banco de dados).

Essa técnica está em crescente uso no mercado pois permite abstrair a utilização de comandos SQL, não cria dependência com uma marca de banco de dados específica e pode-se observar um ganho de produtividade nas tarefas diárias.

Diversos *frameworks* que implementam essa técnica surgiram no mercado nos últimos anos escritos em diversas linguagens de programação, para facilitar a utilização dessa técnica. Porém alguns possuem certas limitações que este trabalho pretende remover.

# Introdução

Para desenvolver um software de qualidade, é preciso pensar em manutenção e evolução. Desenvolvedores lidam com diversas variáveis e resolver vários problemas ao longo de um dia de trabalho. Muitos desses problemas que surgem, são problemas repetidos, e ao invés de desenvolver novas soluções, eles reutilizam soluções que funcionaram no passado, e os utilizam repetidamente em seus projetos.

Por isso a importância dos padrões de projetos e *design patterns*, eles nos proporcionam reaproveitamento de soluções para projetos, e não apenas a reutilização de código.

O *framework* Lumberjack’s ORM será desenvolvido para facilitar a utilização de bancos de dados em aplicações desenvolvidas em PHP, permitindo que estas eliminem por completo a dependência com a linguagem SQL e com um banco de dados de marca específica, tornando simples a configuração e integração com os bancos de dados mais comuns do mercado.

O framework conta com as seguintes funcionalidades em sua arquitetura:

* Mapeamento dos modelos de dados utilizando o conceito de *Annotations*;
* Armazenamento e utilização de múltiplas conexões com bancos de dados;
* Possibilitar a criação de tabelas no banco de dados caso não existam baseando-se no mapeamento dos modelos;
* Criação consultas possibilitando a utilização de quaisquer tipos de relacionamentos, filtros, ordenações, agrupamentos e funções de agregação.

Para fins de demonstração das funcionalidades do framework, será criada uma aplicação simples.

Ao final do projeto, o ORM desenvolvido será disponibilizado gratuitamente à comunidade de desenvolvedores através de um repositório público no GitHub, uma vez que possua uma versão estável.

# Análise da Situação Atual

## Descrição da Situação Atual

Atualmente no mercado existem alguns *frameworks* ORM em PHP, porém, vários tem problemas com documentação, robustez ou é insuficiente no que diz respeito a relacionamentos entre tabelas, principalmente em um relacionamento N para N. E a criação de novos projetos, exige uma preocupação extra no que se relaciona a banco de dados, pois, sem a utilização de um *framework* ORM as aplicações ficam com muita dependência de um determinado banco em si. O que torna a manutenção e/ou evolução da aplicação muito mais complicada de ser realizada, por exemplo, para migrar para um outro banco de dados mais robusto no futuro.

## Objetivo

O *framework* tem como objetivo facilitar a comunicação com diversos bancos de dados relacionais (como mostrado na figura 1) em projetos PHP, que seja possível substituir o banco de dados utilizado de maneira fácil e transparente.

Figura 1: Estrutura do Framework



Fonte: (Autor,2016)

De forma que o *framework* forneça um suporte à diversas funcionalidades que visam simplificar tarefas, que antes seriam feitas manualmente, e ocupariam muito tempo. Tempo este que poderia ter um aproveitamento mais significativo se focado em tarefas mais importantes do projeto.

As funcionalidades englobam:

* Mapeamento de entidades (tabelas ou *views*) utilizando *Annotations*, possibilitando também que seja mapeado todos os tipos de relacionamentos entre as entidades;
* Persistência de dados de novos registros ou registros existentes.
* Deleção de dados existentes no banco.
* Consultas complexas contando com filtros, agrupamentos, ordenações e agregações.

# Análise de Requisitos

Texto.

## Situação Proposta

Texto.

## Requisitos Funcionais

Texto.

### Diagrama de Caso de Uso de Alto Nível

Texto.

### Diagrama de Caso de Uso de Baixo Nível

Texto.

## Requisitos Não-Funcionais

Texto.

### Requisitos de Software

Texto.

# Projeto Detalhado do Software

Texto.

## Arquitetura da Aplicação

Texto.

### Alguma coisa aqui

Texto.

## Tecnologias a serem usadas

Texto.

### Alguma coisa aqui

Texto.

## Diagrama de Classe

Texto.

### Requisitos de Software

Texto.

## Diagrama de Sequência

Texto.

### Alguma coisa aqui

Texto.

# Codificação

Toda a parte de codificação será anexa na mídia física entregue junto a esta documentação.

O código fonte relacionado ao *framework* Lamberjack’s ORM pode ser encontrado no seguinte repositório do GitHub https://github.com/dfrancklin/orm/

# Análise dos Resultados

Texto.

# Referências

# Glossário

**Anexo – Manual do Desenvolvedor**

Texto.