

CURSO BACHARELADO EM SISTEMAS DE INFORMAÇÃO

PALOMA GOMES MONTEIRO

**GUIARmaps: SISTEMA DE ADMINISTRAÇÃO DE MAPAS FUNCIONAIS PARA
SUPORTE À LOCOMOÇÃO DE PESSOAS COM DEFICIÊNCIA VISUAL**

Campos dos Goytacazes/RJ

2015

PALOMA GOMES MONTEIRO

GUIARmaps: SISTEMA DE ADMINISTRAÇÃO DE MAPAS FUNCIONAIS PARA
SUPORTE À LOCOMOÇÃO DE PESSOAS COM DEFICIÊNCIA VISUAL

Monografia apresentada ao Instituto Federal
de Educação, Ciência e Tecnologia
Fluminense como requisito parcial para
conclusão do Curso Bacharelado em Sistemas
de Informação.

Orientador: Prof. Msc. David Vasconcelos Corrêa da Silva

Campos dos Goytacazes/RJ

2015

PALOMA GOMES MONTEIRO

GUIARmaps: SISTEMA DE ADMINISTRAÇÃO DE MAPAS FUNCIONAIS PARA
SUPORTE À LOCOMOÇÃO DE PESSOAS COM DEFICIÊNCIA VISUAL

Monografia apresentada ao Instituto Federal
de Educação, Ciência e Tecnologia
Fluminense como requisito parcial para
conclusão do Curso Bacharelado em Sistemas
de Informação.

Aprovada em

Banca Avaliadora:

.....
Prof. Msc. David Vasconcelos Corrêa Da Silva (Orientador)

.....
Prof. Msc. Eduardo Francisco Da Silva Freire

.....
Prof. Msc. Osvaldo Gomes Terra Junior

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus pela oportunidade de encarar essa caminhada e pela força me dada para concluí-la. Agradeço também a minha família e amigos pelo apoio, principalmente a minha mãe que sempre acreditou em mim.

Não posso deixar de agradecer aos bons professores que fizeram parte da minha história no Instituto Federal Fluminense. Deles recebi orientação para construir minha base de conhecimento.

Sempre que você vir uma pessoa de sucesso, você verá as glórias, nunca os sacrifícios
que os levaram a conquistá-lo.
(Autoria não confirmada)

RESUMO

Esse Projeto de Conclusão de Curso visa desenvolver o protótipo do sistema GUIARmaps, que é uma ferramenta *web* capaz de criar e disponibilizar mapas usados pelo sistema GUIAR para suporte na locomoção de pessoas com deficiência visual. Levando-se em conta que o desenvolvimento de *software* sem qualquer tipo de auxílio é uma tarefa árdua, utilizou-se o MySQL, como sistema gerenciador de banco de dados, e a linguagem PHP em conjunto com a solução *Zend Framework*. Obteve-se um sistema que se adapta à tela em que é apresentado e cujos mapas atendem as necessidades do sistema GUIAR e ficam disponíveis para download pelos usuários. Ao final foram executados os testes do Modelo de Acessibilidade em Governo Eletrônico para verificar a conformidade do protótipo com as melhores práticas em acessibilidade digital.

Palavras-chave: Desenvolvimento *Web*; PHP; *Zend Framework*

ABSTRACT

This project is intended to develop the GUIARmaps, that is a web tool able to create and available of maps used by GUIAR system to support the locomotion of visually impaired persons. Taking into account that the the development of software without any help is a difficult task, were used the MySQL, as database management system, and the PHP language in conjunction with Zend Framework solution. There was obtained a system that adapts to the screen that appears and whose maps attend to needs of GUIAR system and are available for download by users. At the end were performed tests of The Accessibility Model in E-government to verify that the system complies with best practice in digital accessibility.

Keywords: Web development; PHP; Zend Framework

LISTA DE ILUSTRAÇÕES

Figura 1 - Atividades do Ciclo de Vida Clássico	24
Figura 2 – Sequência de eventos da Prototipação	25
Figura 3 – O modelo espiral	26
Figura 4 – Diagrama de sequência de aplicações web que utiliza o MVC	28
Figura 5 – Uma página dinâmica sendo executada no servidor	29
Figura 6 - Diagrama de Caso de Uso	36
Figura 7 - Diagrama Entidade Relacionamento	40
Figura 8 – Etapa de seleção de <i>framework</i> na criação de projeto no <i>Netbeans</i>	42
Figura 9 – Estrutura de pastas do protótipo no <i>Netbeans</i> IDE	43
Figura 10 – Parte do código do arquivo <i>Application.ini</i>	44
Figura 11 – Parte do código do controlador Mapa	45
Figura 12 – Janela Executar Zend Comando do NetBeans	46
Figura 13 – Parte do código do formulário Mapa	47
Figura 14 – Tela Inicial	49
Figura 15 – Tela de <i>logon</i>	50
Figura 16 – Tela cadastro de usuários	50

Figura 17 – Tela perfil dos usuários	51
Figura 18 – Tela alteração de senha de usuário	51
Figura 19 – Tela alteração de e-mail de usuário	51
Figura 20 – Tela cadastro de mapa	52
Figura 21 – Tela inclusão de Pontos de Interesse	52
Figura 22 – Tela inclusão de Tipo de Ponto de Interesse	53
Figura 23 – Tela de Mapas Não Publicados	53
Figura 24 – Níveis de pontuação	55

LISTA DE QUADROS

Quadro 1 - Descrição do Caso de Uso Realizar Cadastro	37
Quadro 2 - Descrição do Caso de Uso Desenvolver Mapas	37
Quadro 3 - Descrição do Caso de Uso Publicar Mapas	38
Quadro 4 - Descrição do Caso de Uso Obter Mapa	38
Quadro 5 – <i>Checklist</i> de utilização de <i>links</i>	58
Quadro 6- Descrição da tabela Usuario	65
Quadro 7 - Descrição da tabela Predio	66
Quadro 8 - Descrição da tabela Mapa	66
Quadro 9 - Descrição da tabela Idioma	66
Quadro 10 - Descrição da tabela PontoInteresse	67
Quadro 11 - Descrição da tabela TipoPonto	67
Quadro 12 - Descrição da tabela RFID	67
Quadro 13 - Descrição da tabela ListaPontoEncadeado	43

LISTA DE TABELAS

Tabela 1 - Tabela de total de pontuações e médias das avaliações	54
--	----

LISTA DE GRÁFICOS

Gráfico 1 – Média final dos grupos de itens avaliados	56
---	----

LISTA DE ABREVIATURAS E SIGLAS

API	- Application Programming Interface
CSS	- Cascading Style Sheets
eMAG	- Modelo de Acessibilidade em Governo Eletrônico
HTML	- Hypertext Markup Language
IDE	- Integrated Development Environment
IFF	- Instituto Federal Fluminense
MP	- Ministério do Planejamento
MVC	- Model, View and Controller
PDF	- Portable Document Format
PHP	- Hypertext Preprocessor
RFID	- Radio-Frequency IDentification
SETEC	- Secretaria de Educação Profissional e Tecnológica
W3C	- World Wide Web Consortium

ZF - Zend Framework

SUMÁRIO

1. INTRODUÇÃO	17
1.1. OBJETIVO GERAL.....	19
1.2. OBJETIVOS ESPECÍFICOS	19
1.3. ESTRUTURA DO TRABALHO	19
2. REFERENCIAL TEÓRICO	21
2.1. ACESSIBILIDADE NA WEB E O MODELO DE ACESSIBILIDADE BRASILEIRO	21
2.2. PROCESSO DE DESENVOLVIMENTO DE SOFTWARE	22
2.3. MVC.....	27
2.4. PHP.....	28
2.5. ZEND FRAMEWORK	30
2.6. NETBEANS IDE	31
2.7. WAMPSEVER	31
2.8. TWITTER BOOTSTRAP	32
3. ANÁLISE E PROJETO	34
3.1. REQUISITOS.....	34
3.1.1. REQUISITOS FUNCIONAIS DO SISTEMA.....	34
3.1.2. REQUISITOS NÃO FUNCIONAIS	35
3.2. DIAGRAMAS.....	35
3.2.1. DIAGRAMA DE CASO DE USO	36
3.2.2. DESCRIÇÃO DOS CASOS DE USO	37
3.2.3. DIAGRAMA ENTIDADE RELACIONAMENTO.....	39
4. IMPLEMENTAÇÃO	41
4.1. FERRAMENTAS DE DESENVOLVIMENTO.....	41
4.2. FORMULÁRIOS	45
4.3. PROJETO DA INTERFACE	48
4.4. TESTES.....	53
5. CONCLUSÕES.....	59
5.1. TRABALHOS FUTUROS.....	59

REFERÊNCIAS BIBLIOGRÁFICAS	61
APÊNDICE A: ESTRUTURA DO BANCO DE DADOS	65
APÊNDICE B: IMPLANTAÇÃO DO PROTÓTIPO NO SERVIDOR.....	68
APÊNDICE C: QUESTIONÁRIO DE AVALIAÇÃO DE QUALIDADE.....	69

1. INTRODUÇÃO

Diariamente pessoas portadoras de deficiência visual enfrentam desafios para se locomover por lugares desconhecidos. Segundo Muñoz (2010), o dispositivo mais utilizado no suporte à locomoção do deficiente visual atualmente é a bengala branca. Assim como o cão-guia, que não é tão utilizado devido ao custo de aquisição e treinamento, a bengala branca auxilia o deficiente visual a evitar obstáculos e não o orienta a como chegar ao local desejado.

A dificuldade ou incapacidade de enxergar é uma realidade para muitas pessoas do Brasil e do mundo. Localizar um banheiro ou a saída de um prédio, por exemplo, torna-se uma tarefa difícil para essas pessoas devido à falta de informação visual.

"Para conhecer o ambiente e transpor as limitações, habitualmente são empregados bengalas e cães-guia. Mas além disso, é possível estender o apoio dado a estas pessoas durante seus deslocamentos e tarefas diárias através de tecnologias inclusivas" (MEIRELES, 2009).

Após observar as dificuldades de locomoção que os novos alunos com deficiência visual do Instituto Federal Fluminense enfrentam, foi elaborado por Silva et al. (2013) um projeto de pesquisa com o intuito desenvolver um protótipo de sistema automatizado para locomoção autônoma de deficientes visuais. Recentemente o sistema recebeu o nome de "GUIAR".

Para melhor compreensão do projeto "GUIAR" as atividades desenvolvidas podem ser divididas em:

- Instalar nos ambientes pisos táteis munidos de etiquetas RFID¹;
- Construir uma bengala equipada com hardware próprio, capaz de processar e enviar códigos RFID via *bluetooth*² para um *smartphone*³ do usuário;

¹ Método de identificação automática através de sinais de rádio, recuperando e armazenando dados remotamente através de dispositivos denominados etiquetas RFID. Disponível em: <http://pt.wikipedia.org/wiki/Identificação_por_radiofrequência>. Acesso em: 22 abr. 2015.

- Construir uma aplicação para *smartphones* que, através dos dados passados pelo leitor RFID, seja capaz de identificar a localização do usuário e orientá-lo em sua locomoção;
- Criação de um sistema que crie, gerencie e disponibilize mapas funcionais, permitindo associar as etiquetas RFID presentes nos pisos táteis com a sua localização física.

O sistema GUIAR funciona da seguinte maneira: durante a locomoção do usuário, o leitor de RFID presente na bengala realiza a leitura das etiquetas instaladas nos pisos táteis e envia o código de localização via *bluetooth* para uma aplicação executada no *smartphone* do usuário. Em seguida a aplicação consultará o mapa funcional e informará através de mensagens de voz a localização do usuário.

O escopo deste trabalho se encaixa na atividade de criar, gerenciar e disponibilizar mapas funcionais. No sistema desenvolvido neste trabalho será possível informar a qual distância uma determinada etiqueta RFID está de pontos como banheiros ou escadas por exemplo. Essas informações funcionarão como um mapa para orientar o deficiente visual no seu trajeto. Todas as informações cadastradas de um determinado prédio serão armazenadas em um servidor de banco de dados e disponibilizadas para serem transferidas para um banco de dados local do *smartphone* usado pelo usuário.

² Sistema de comunicação sem fio destinado a substituir os cabos que conectam dispositivos, mantendo elevados níveis de segurança. Disponível em: < <http://www.bluetooth.com/Pages/Basics.aspx>>. Acesso em: 22 abr. 2015.

³ Telefone móvel com funcionalidades avançadas que podem ser estendidas por meio de programas executados por seu sistema operacional. Disponível em: < <https://pt.wikipedia.org/wiki/Smartphone>>. Acesso em: 22 abr. 2015.

1.1. OBJETIVO GERAL

Este trabalho tem como objetivo desenvolver uma aplicação *web*, que seja capaz de criar, gerenciar, e disponibilizar mapas funcionais para suporte a locomoção de pessoas com deficiência visual.

1.2. OBJETIVOS ESPECÍFICOS

Os principais objetivos específicos são:

- Identificar e demonstrar características que diferenciam o desenvolvimento de aplicações *web* através do *Zend Framework*;
- Desenvolver um sistema que permita:
 - Cadastrar usuários e possibilitar que estes cadastrem informações para orientação na locomoção autônoma de deficientes visuais em ambientes fechados;
 - Disponibilizar para download arquivos de banco de dados, chamados de mapas, com as informações cadastradas pelos usuários.

1.3. ESTRUTURA DO TRABALHO

O presente trabalho possui seis capítulos e está organizado da seguinte maneira: o primeiro capítulo apresenta a introdução ao tema do estudo, objetivos do trabalho e a estrutura do mesmo; o segundo capítulo apresenta o referencial teórico, focalizando os principais

conceitos envolvidos no trabalho; no terceiro capítulo são apresentadas as funcionalidades do sistema; o quarto capítulo apresenta os materiais e métodos utilizados no desenvolvimento do trabalho; o quinto capítulo apresenta as conclusões tiradas acerca deste trabalho e as propostas de trabalhos futuros; o último capítulo apresenta as referências bibliográficas que fizeram parte do embasamento teórico do trabalho.

2. REFERENCIAL TEÓRICO

Esse capítulo tem como objetivo apresentar o embasamento teórico para compreensão do presente trabalho. No início do capítulo a Sessão 2.1 apresenta uma visão geral sobre acessibilidade digital e o modelo de acessibilidade criado pelo governo brasileiro; em seguida a Sessão 2.2 define o conceito de processo de desenvolvimento de software; na Sessão 2.3 é a vez de conceituar o padrão MVC; nas demais sessões é esclarecido o que é PHP, *Zend Framework*, *NetBeans IDE*, *WampServer* e *Twitter Bootstrap*.

2.1. ACESSIBILIDADE NA WEB E O MODELO DE ACESSIBILIDADE BRASILEIRO

Em 1960 foi criada nos Estados Unidos uma rede experimental de computadores de longa distância, chamada ARPANET. Seu objetivo era compartilhar recursos computacionais, mas também foi utilizada para troca de mensagens via correio eletrônico, pesquisas a computadores remotos e até desenvolvimento conjunto. Com o passar dos anos a ARPANET tornou-se a espinha dorsal de uma confederação de redes locais e regionais, chamada Internet. (MENDES, 2007).

A *web* é um serviço ofertado pela Internet que proporciona o acesso permanente às informações. O *World Wide Web Consortium* (W3C), consórcio internacional voltado à criação de padrões para a *web*, incentiva a padronização no desenvolvimento de páginas *web* de forma que sejam acessíveis para todos (CARTILHA, 2013).

Mendes (2007) acredita que a Internet se tornou popular devido a não necessidade de experiência para a navegação. Com ela o compartilhamento de conhecimento e o entretenimento tornaram-se tarefas fáceis. Esse termo passou então a ser utilizado para descrever uma rede onde tudo se pode e tudo se consegue.

De acordo com Cartilha (2013), no ambiente específico da web a acessibilidade “pode-se dizer que se trata da possibilidade e da condição de alcance, percepção e entendimento para a utilização, em igualdade de oportunidades, com segurança e autonomia, dos sítios e serviços disponíveis na web”. Já Conforto (2002) resume a acessibilidade como um sinônimo de aproximação, uma forma de permitir que cada usuário faça uso de interfaces que respeitam suas necessidades e preferências. Ele afirma que garantir a acessibilidade na Internet é permitir ouvir e dar voz a todas as pessoas.

O Modelo de Acessibilidade de Governo Eletrônico (eMAG) foi idealizado pelo Ministério do Planejamento, Orçamento e Gestão do Brasil com o intuito de dar suporte na disponibilização de conteúdos governamentais na Internet. Nesse modelo são apresentadas recomendações simplificadas baseadas em regras do W3C, mas adaptadas à realidade e necessidades brasileiras (FREIRE, 2008).

A primeira versão do eMAG foi liberada para consulta pública em janeiro de 2005 visto que em dezembro de 2004 as Leis nº 10.048 e nº 10.098 foram regulamentadas estabelecendo o prazo de um ano para que todos os sites e portais públicos fossem acessíveis a todos. (LOBATO, 2012)

Atualmente na versão 3.1, o eMAG sofreu várias revisões ao longo dos anos. A nova versão apresenta texto mais compreensível, diversos exemplos das recomendações e foi desenvolvido em parceria com o Instituto Federal do Rio Grande do Sul (IFRS) (UFSM, 2015).

E-MAG (2014) explica, seguir as recomendações do eMAG também facilita o acesso ao conteúdo através de ferramentas como Smartphones e navegador por voz e quando houver limitações técnicas como conexão lenta e falta de recursos de mídia.

2.2. PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Segundo Pressman (2011), para desenvolver um software de qualidade devemos perceber quatro fatos reais:

1. Atualmente os softwares estão incorporados em vários aspectos do cotidiano, com isso a quantidade de funções ofertadas por uma aplicação tem crescido bastante. Antes de iniciar o desenvolvimento de uma aplicação ou sistema

embutido é necessário concentrar esforços para compreender o problema, para que o mesmo seja atendido pela aplicação.

2. Os requisitos de tecnologia da informação existentes estão se tornando mais complexos a cada ano. Essa complexidade exige maior atenção com as interações de todas as partes do sistema e tornou a atividade de projeto fundamental no processo de desenvolvimento.
3. Um software deve possuir qualidade elevada, visto que tomadas de decisões estratégicas e táticas estão cada vez mais dependentes de soluções de software e a parada de um software pode acarretar falhas catastróficas.
4. Um software deve ser passível de manutenção, visto que com o aumento de sua longevidade serão necessárias adaptações e aprimoramentos.

Rezende (2005) explica que um software tem um ciclo de vida curto, de no máximo 5 anos, porém ao longo de sua vida exigirá manutenção legal, correções e melhorias ou implementações.

No processo de desenvolvimento de um sistema é importante seguir um roteiro com uma série de passos para ajudar na criação de um sistema de alta qualidade e dentro do prazo determinado (PRESSMAN, 2011).

O ciclo de vida de um software em geral abrange as seguintes fases: nascimento ou concepção, construção, implantação, implementações (ajustes após a implantação), maturidade e utilização plena, declínio, manutenção e morte (REZENDE, 2005).

A engenharia de software abarca um conjunto de etapas comumente chamadas de paradigma de engenharia de software. Um paradigma é escolhido com base na natureza do projeto e da aplicação (PRESSMAN, 2007).

A Figura 1 ilustra o paradigma do ciclo de vida clássico, também conhecido como modelo cascata. Prikladnicki (2008) define o modelo cascata como um ciclo de vida de método sistemático e sequencial, cuja saída de uma fase funciona como entrada de outra.

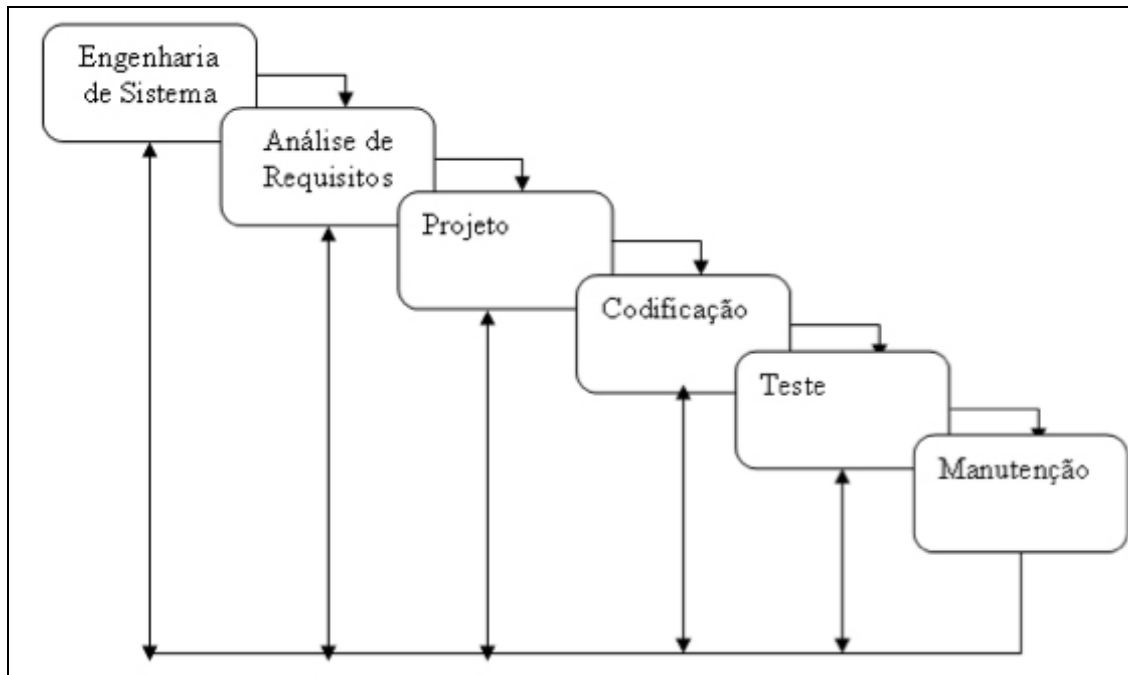


Figura 1 – Atividades do Ciclo de Vida Clássico

Fonte: Adaptado de Pressman (2007)

Prikladnicki (2008) explica que as fases do ciclo de vida clássico são estruturadas para possibilitar que suas atividades sejam executadas por pessoas diferentes, simultaneamente ou não. Segundo Pressman (2007), esse paradigma continua sendo amplamente empregado. Apesar de possuir algumas fragilidades sua utilização é melhor do que uma abordagem casual ao desenvolvimento de software.

Prototipação é outro paradigma de engenharia de software e sua sequência de eventos é apresentada na Figura 2. Pressman (2007) explica que, na prototipação o desenvolvedor e o cliente se reúnem para definir os objetivos do sistema e então o protótipo inicial, o “projeto rápido”, é elaborado a partir do que foi esboçado. Esse projeto rápido leva a construção de um protótipo que é avaliado pelo cliente e utilizado para refinar os requisitos para o sistema que será desenvolvido. Uma interação ocorre quando o protótipo atende as necessidades do cliente, capacitando o desenvolvedor a entender o que precisa ser feito.

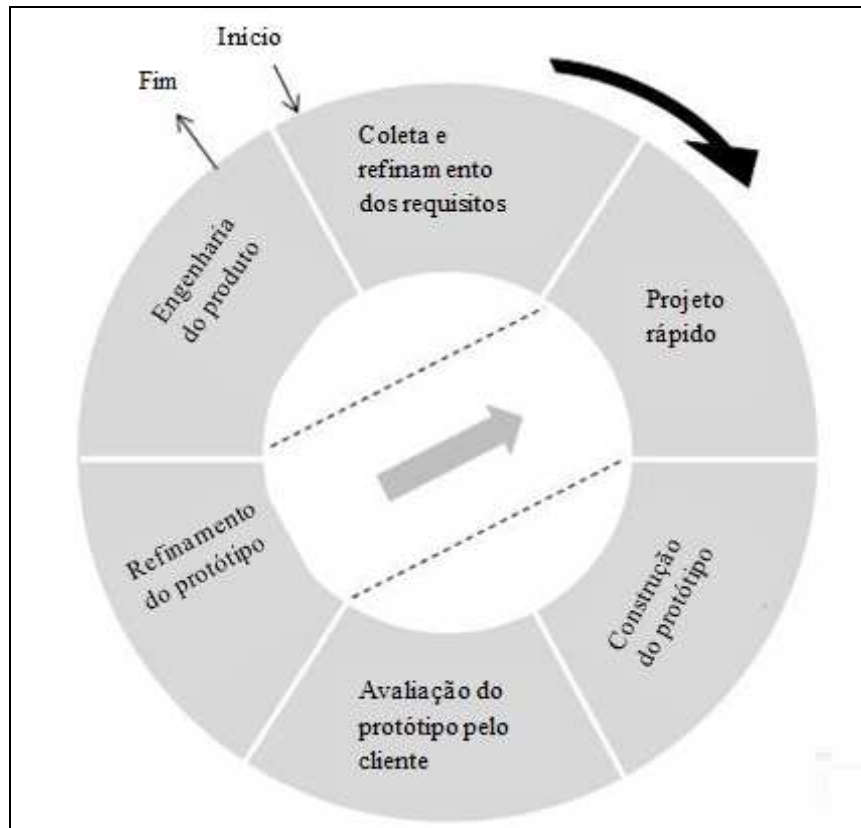


Figura 2 – Sequência de eventos da Prototipação

Fonte: Adaptado de Pressman (2007)

A prototipação é um paradigma eficiente. A chave é definir as regras de forma que o desenvolvedor e o cliente concordem que o protótipo servirá para definir os requisitos do sistema, que será construído levando em conta a qualidade e manutenibilidade (PRESSMAN, 2007).

A Figura 3 descreve as atividades do modelo espiral. De acordo com Pressman (2007), o modelo espiral foi desenvolvido para englobar as melhores características do ciclo de vida clássico e da prototipação, acrescentando a análise dos riscos. Esse paradigma apresenta as seguintes atividades:

1. Planejamento, onde os objetivos, alternativas e restrições são determinados.
2. Análise de riscos, onde as alternativas são analisadas e onde ocorre a identificação/resolução dos riscos.
3. Engenharia, onde é desenvolvido o produto que será apresentado na próxima atividade.
4. Avaliação feita pelo cliente, onde o cliente avalia o que foi produzido na engenharia.

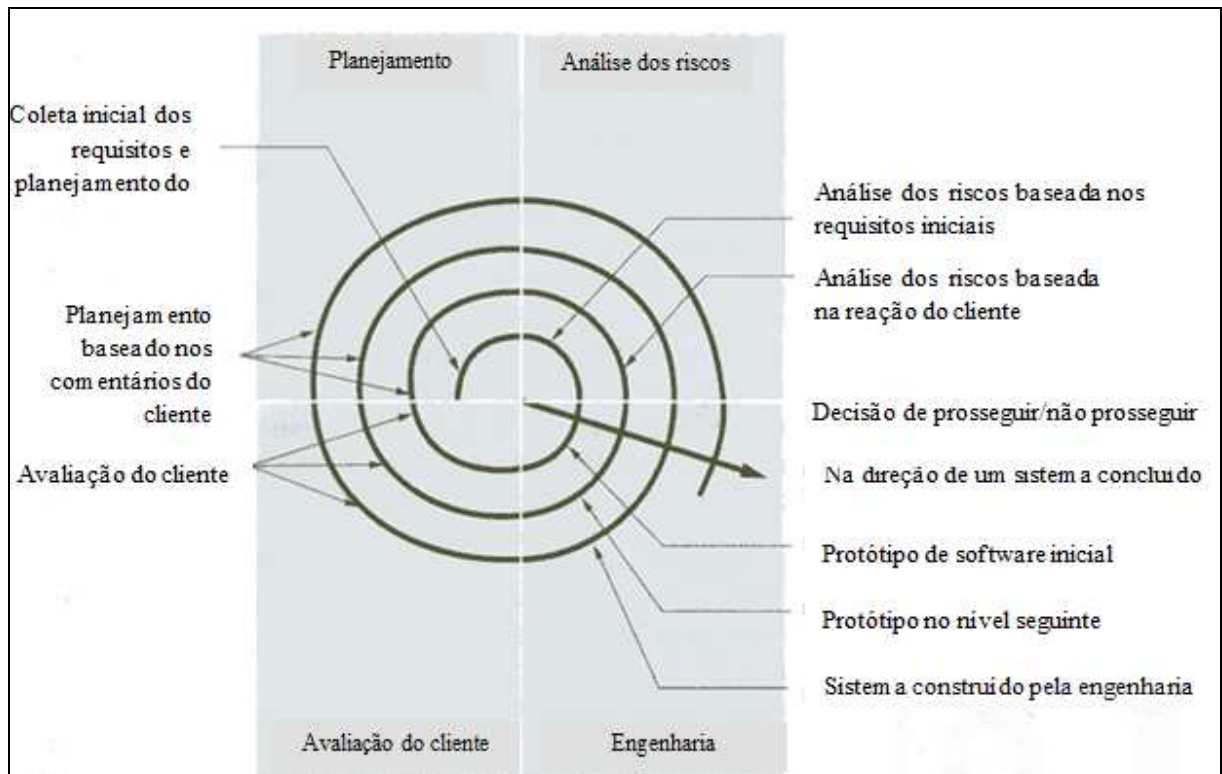


Figura 3 – O modelo espiral

Fonte: Adaptado de Pressman (2007)

O modelo espiral é um ciclo de vida realista capaz de capacitar a equipe do projeto e o cliente a entender e reagir aos riscos de cada etapa evolutiva da espiral (PRIKLADNICKI, 2008).

Técnicas de quarta geração é outro paradigma de desenvolvimento vastamente discutido. Pressman (2007) define esse paradigma como um conjunto de ferramentas de software que possibilita que o desenvolvedor especifique algumas características do software de forma elevada. Com base nessas especificações a ferramenta gera automaticamente o código-fonte.

A pesar dos paradigmas de engenharia de software serem bem definidos Pressman (2007) acredita que é possível combinar as abordagens de forma que o todo seja maior do que a soma das partes. Ele explica que a natureza da aplicação que deve ditar a abordagem a ser seguida.

2.3. MVC

O padrão MVC é explicado por Minetto (2007) como: uma abreviação para Modelo, Visão e Controle (do inglês *Model*, *View* e *Controller*) que tem por objetivo separar o desenvolvimento de uma aplicação em três partes. O modelo realiza o gerenciamento do comportamento dos dados, a parte da aplicação que é visível ao usuário fica na visão e o *controller* é a parte que interpreta as informações inseridas pelo usuário, comandando a Visão e o Modelo.

O padrão MVC surgiu décadas atrás e atualmente mostra-se presente em diversos frameworks e linguagens. Essa arquitetura oferece aos projetistas e desenvolvedores uma separação clara dos elementos com o intuito de separar as responsabilidades da equipe desenvolvedora, permitir a reutilização de códigos fontes e facilitar a manutenção do sistema. Devido à escalabilidade fornecida pelo MVC essa arquitetura mostra-se fundamental em aplicações web e desktop (BAPTISTELLA, 2011). Em outras palavras, Mourato (2009) explica que, “a arquitetura MVC trouxe o desacoplamento dos módulos do sistema, facilitando a manutenção do código, além de deixar mais elegante o desenvolvimento do sistema”.

O diagrama de sequência apresentado na Figura 4 mostra como uma aplicação web que utiliza a arquitetura MVC se comporta ao receber requisições de um navegador. Ranthum (2005) explica que, a requisição do Navegador é recebida pelo Controlador que realiza a ação correspondente no Modelo e escolhe a Visão a ser exibida. A Visão obtém os dados necessários do modelo e a página HTML gerada é exibida no navegador.

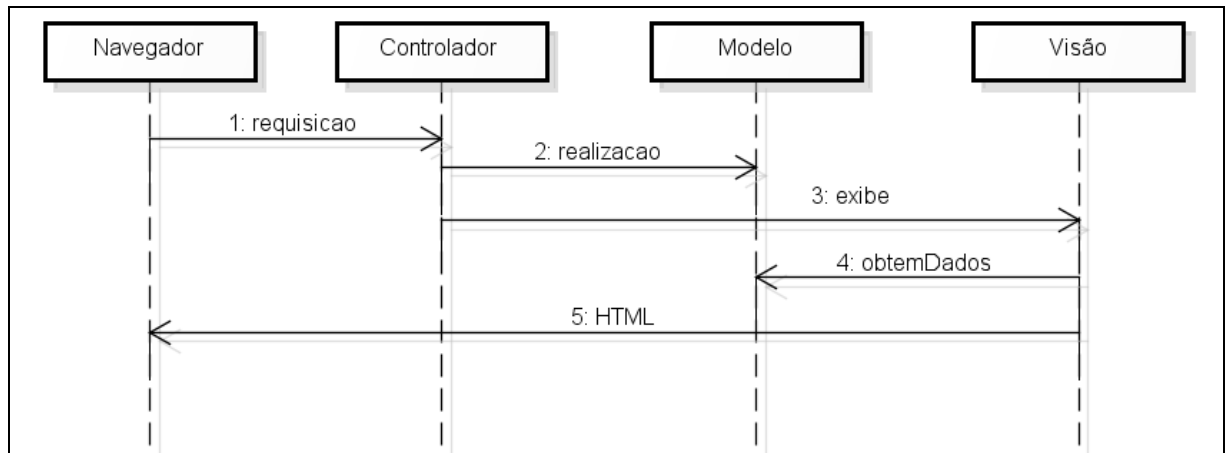


Figura 4 – Diagrama de sequência de aplicações web que utiliza o MVC

Fonte: Adaptado de Ranthum (2005)

A vantagem de utilização do MVC é a separação entre lógica e apresentação do sistema, que beneficia o desenvolvimento em equipe. Um *designer* pode trabalhar na camada *view*, enquanto um administrador de banco de dados trabalha na camada *model* e um programador concentra-se em incluir regras de negócios no *controller* (MINETTO, 2007).

2.4. PHP

O termo PHP é uma abreviatura de *Hypertext Preprocessor*, que em português significa Pré-processador de Hipertexto. PHP é uma linguagem de *script* de código aberto que quando embutida em código HTML gera páginas com conteúdo dinâmico (MELO, 2007).

O PHP é muito popular na web e está presente em milhões de sites no mundo inteiro. A capacidade do PHP de interagir com o mundo *web* é seu diferencial em relação às demais linguagens. O surgimento da linguagem PHP transformou totalmente os *web* sites que possuem páginas estáticas (NIEDERAUER, 2011).

Minetto (2007) explica que a linguagem PHP foi criada como uma linguagem estruturada, mas ao longo dos anos vários recursos foram adicionados à ao PHP com a intenção de transformá-lo em uma linguagem orientada a objetos.

Niederauer (2011) destaca que além de ser gratuito o PHP é um software de código aberto. Seu código-fonte e sua documentação detalhada estão disponíveis no site oficial, <http://www.php.net>.

Segundo Minetto (2007), o diferencial do PHP é sua facilidade de aprendizado. Como auxílio de livros e tutoriais, um programador é capaz de elaborar formulários HTML utilizando *scripts* PHP. Essa característica favoreceu o aumento da quantidade de programadores dessa linguagem e o surgimento de grandes *softwares* elaborados com ela.

Ao contrário de outras linguagens, quando se acessa uma página PHP por meio de um *browser*, todo o código PHP é executado no servidor e o resultado é enviado para o *browser*. Sendo assim, ninguém tem acesso às linhas de programação já que o *browser* exibe apenas o resultado do código processado (NIEDERAUER, 2011).

O esquema da Figura 5 apresenta os passos seguidos quando um produto é pesquisado em uma página da *web*.

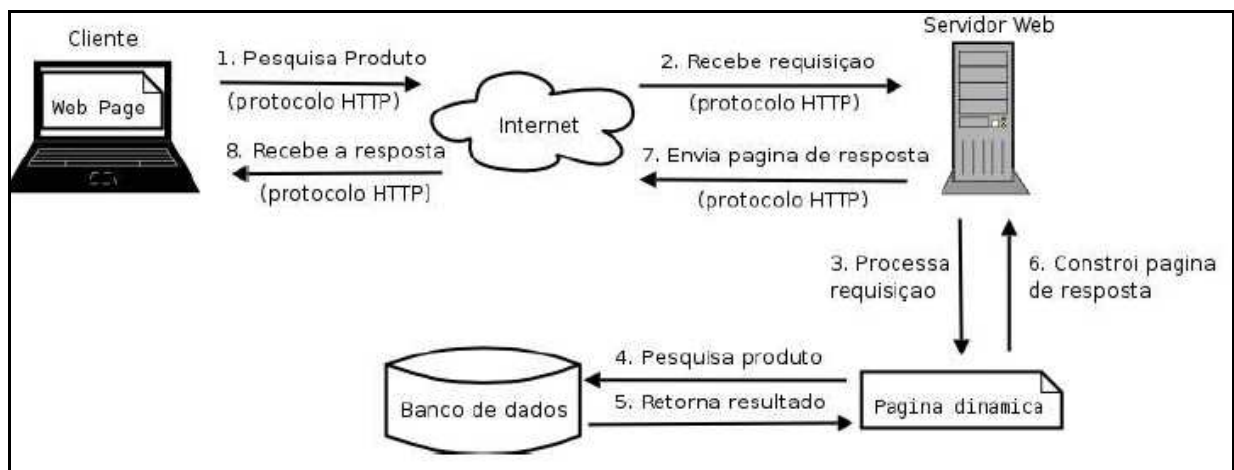


Figura 5 – Uma página dinâmica sendo executada no servidor

Fonte: Adaptado de CONALLEN (2002)

O PHP possui um grande potencial, chegando ao ponto de poder causar vários estragos em um servidor. Para obter segurança nessa linguagem é necessário tomar medidas preventivas para que acidentes não aconteçam. Conhecer os pontos fracos dos programas é uma chave para evitar ataques e surpresas desagradáveis (JOBSTRAIBIZER, 2009).

2.5. ZEND FRAMEWORK

Zend Framework é uma arquitetura de desenvolvimento criada pela *Zend Technologies*, contribuidora e mentora do PHP. Ele é uma alternativa para a criação de sistemas de forma ágil e padronizada (LISBOA, 2008).

Segundo Jobstraibizer (2009), “um *framework* é um conjunto de componentes que contém uma arquitetura e uma estrutura interna básica para o desenvolvimento de uma aplicação. Funciona como uma aplicação semipronta que deve ser estendida e personalizada para que um sistema desenvolvido funcione corretamente”.

O *Zend Framework*, ou simplesmente ZF, é um *framework open source* para desenvolvimento em PHP. O ZF possui código-fonte disponível e compatível com a *BSD License*, o que autoriza que os softwares construídos com esse *framework* possuam código-fonte proprietário (LISBOA, 2008).

De acordo com o levantamento apresentado por Skvorc (2013), o *Zend Framework* esteve na lista dos dez *frameworks* PHP mais promissores para o ano de 2014. Nessa lista também aparecem *frameworks* como o *Symfony*, *CakePHP* e *Laravel*.

O *Zend Framework* tem como principal vantagem ser uma solução com grande flexibilidade de uso, enquanto outros *frameworks* impõem a utilização de todos os seus componentes (SERPRO, 2008).

Lisboa (2008) explica que, o *Zend* foi idealizado para ser um *framework* simples. Ele fornece uma biblioteca com grande parte das funcionalidades que todo o desenvolvedor precisa. A baixa curva de aprendizado do ZF é possível graças à:

- ▲ uma base de códigos bem testada e extensível
- ▲ uma arquitetura flexível
- ▲ dispensa de arquivos de configuração para “acordar” e “rodar” a aplicação

SERPRO, BBC e Cisco WebEx são alguns exemplos de corporações que utilizam o *Zend Framework* no desenvolvimento.

2.6. NETBEANS IDE

Um Ambiente Integrado de Desenvolvimento (IDE) é um *software* que possui características e ferramentas para suporte de desenvolvimento de sistemas com o intuito de agilizar esse processo (OLIVEIRA, 2008).

A utilização de um IDE apropriado durante o processo de desenvolvimento de *software* é de suma importância em um desenvolvimento de sucesso, principalmente quando trata-se de um desenvolvimento *web* (CHAVES, 2008).

O *NetBeans* IDE permite que se desenvolva rapidamente e facilmente uma aplicação Java para *desktop*, *mobile* e aplicações *web*, bem como aplicações HTML5 com HTML, *JavaScript* e CSS. O IDE também fornece um grande conjunto de ferramentas para PHP e desenvolvedores de C/C++. O *Netbeans* é um *software* livre, de código aberto e tem uma grande comunidade de usuários e desenvolvedores em todo o mundo (NETBEANS IDE, 2014).

Böck (2011) acredita que vale a pena usar a plataforma *NetBeans*, mesmo se aplicativo é muito pequeno ou muito grande, devido a qualidade do ambiente de execução e das inúmeras APIs que oferecem soluções práticas para problemas frequentes. Já Oliveira (2008) recomendada o uso do *NetBeans*, pois ele possui ferramentas de suporte para *frameworks* de teste, servidores *web* e monitoramento de aplicações. Além disso, possui uma boa documentação disponível.

Segundo Gartner (2011), a plataforma *Netbeans* disponibiliza todas as funcionalidades necessárias para edição de código fonte, além de fornecer ferramentas como dicas e sugestões de códigos automáticos, compilação, depuração e suporte à internacionalização. Esse ambiente pode ser executado nos sistemas operacionais *Windows* e *Linux* e novas funcionalidades podem ser integradas a ele com a utilização de *plugin*.

2.7. WAMPSERVER

WAMP é um acrônimo para *Windows*, *Apache*, *MySQL* e *PHP*. A instalação desse programa permite que um computador funcione como um servidor. Os servidores *web*

usualmente recebem requisições HTTP, processam a requisição e retornam uma informação para quem o chamou (SCHMITZ, 2013).

WampServer é um ambiente de desenvolvimento *web* criado para o sistema operacional *Windows*. Ele permite a criação de aplicações *web* com *Apache*, PHP e banco de dados MySQL. O *Wamp* também fornece o *phpMyAdmin*, que permite o fácil gerenciamento das informações presentes no banco de dados (WAMPSEVER, 2014).

Hedengren (2011) explica que, após instalação e execução do *WampServer* um ícone de menu será apresentado na parte inferior direita da tela. Esse ícone permite o acesso as configurações e arquivos do *Wamp*. Sendo assim, quando necessário arquivos como *php.ini* podem ser rapidamente alterados.

De acordo com Wampserver (2014), o *Wamp* é um software completo e de fácil utilização. Ele permite a alteração o idioma do seu menu e com um clique com o botão esquerdo do *mouse* sobre o ícone é possível:

- Gerenciar os serviços *Apache* e MySQL;
- Alternar o status do servidor (*online/offline*), dando acesso a todos ou somente a *localhost*;
- Instalar e alterar as versões do *Apache*, MySQL e PHP;
- Gerenciar as configurações dos servidores;
- Acessar *logs*;
- Acessar os arquivos de configurações;
- Criar *alias*.

2.8. TWITTER BOOTSTRAP

Desenvolvido por engenheiros da rede social *Twitter*, o *Twitter Bootstrap*, segundo Otto (2011) “é um kit de ferramentas *front-end* para o rápido desenvolvimento de aplicações *web*. É uma coleção de convenções CSS e HTML”.

Lançado em agosto de 2011 por Mark Otto e Jacob Thornton, o *Twitter Bootstrap* permite personalizar a construção do site para atender a sua necessidade. Com ele você escolhe quais recursos deseja no seu site (SPURLOCK,2013).

O *Bootstrap* trabalha para fornecer uma solução limpa e uniforme para que os desenvolvedores de interface encontrem suas tarefas usuais (OTTO, 2011).

O *framework Bootstrap* foi criado para ser utilizados por todos, desde as últimas versões dos grandes navegadores de desktop, até navegadores de *tablets* e *smartphones* via CSS responsivo (BOOTSTRAP, 2014).

O *Twitter Bootstrap* é ideal para a construção de *sites* responsivos. Sua combinação de HTML, CSS e *JavaScript* torna fácil a criação de sites robustos sem adicionar muito código (SPURLOCK, 2013).

3. ANÁLISE E PROJETO

Nesse capítulo as funcionalidades do sistema são apresentadas através da lista de requisitos funcionais e não funcionais e do diagrama de caso de uso e sua descrição. São apresentados também o diagrama entidade relacionamento.

3.1. REQUISITOS

Um requisito pode ser definido como uma característica do projeto, um comportamento ou propriedade do sistema. Ao definir os requisitos está sendo declarado o que se espera que o sistema realize (BOOCH, 2006).

Segundo Ramos (2006), um requisito funcional é aquele que retrata uma determinada ação, ou função, que deve ser suportada pelo sistema. De outro modo, um requisito não funcional determina um aspecto que o sistema deve satisfazer.

3.1.1. REQUISITOS FUNCIONAIS DO SISTEMA

Abaixo estão listados os requisitos do sistema desenvolvido:

- **Cadastrar Usuário:** O sistema deve permitir que os usuários possam se cadastrar.
- **Criar Mapas:** O sistema deve permitir que os usuários logados possam criar mapas.
- **Criar Pontos de Interesses:** O sistema deve permitir que os usuários logados possam criar pontos de interesses, como salas e rampas de acesso, que serão apresentados nos mapas.
- **Disponibilizar Mapa:** O sistema deve permitir que os mapas salvos sejam disponibilizados para download pelo usuário que o criou.
- **Editar Mapa:** O sistema deve permitir que os mapas possam ser editados pelo usuário que o criou.

- Obter Mapa: O sistema deve permitir que qualquer visitante que o acesse possa realizar download dos mapas disponibilizados.

3.1.2. REQUISITOS NÃO FUNCIONAIS

Os requisitos não funcionais do sistema estão listados a seguir.

[RNF01] O sistema deve disponibilizar os mapas, gerando arquivos de exportação de forma automática após a publicação.

[RNF02] Os usuários deverão utilizar *login* e senha para usufruir das funcionalidades avançadas do sistema.

[RNF03] O usuário poderá escolher se deseja publicar o mapa criado ou armazenar para edição.

[RNF04] O sistema deve ser capaz de ser executado em múltiplas plataformas.

[RNF05] O sistema deve obrigatoriamente ser implementado na linguagem PHP.

[RNF06] O visitante receberá o mapa apenas no idioma que solicitou.

3.2. DIAGRAMAS

De acordo com Booch (2006), um diagrama expõe, de forma gráfica, um conjunto de elementos do sistema. Eles são criados para permitir que um sistema seja visualizado em diferentes perspectivas e teoricamente podem possuir qualquer combinação de gráficos de vértices (itens) e arcos (relacionamentos).

3.2.1. DIAGRAMA DE CASO DE USO

Um diagrama de caso de uso detalha a relação entre os casos de uso e os atores do sistema. Esse diagrama possibilita ter uma visão global e de alto nível do sistema (BOOCH, 2006).

A Figura 6 apresenta o Diagrama de Caso de Uso do sistema.

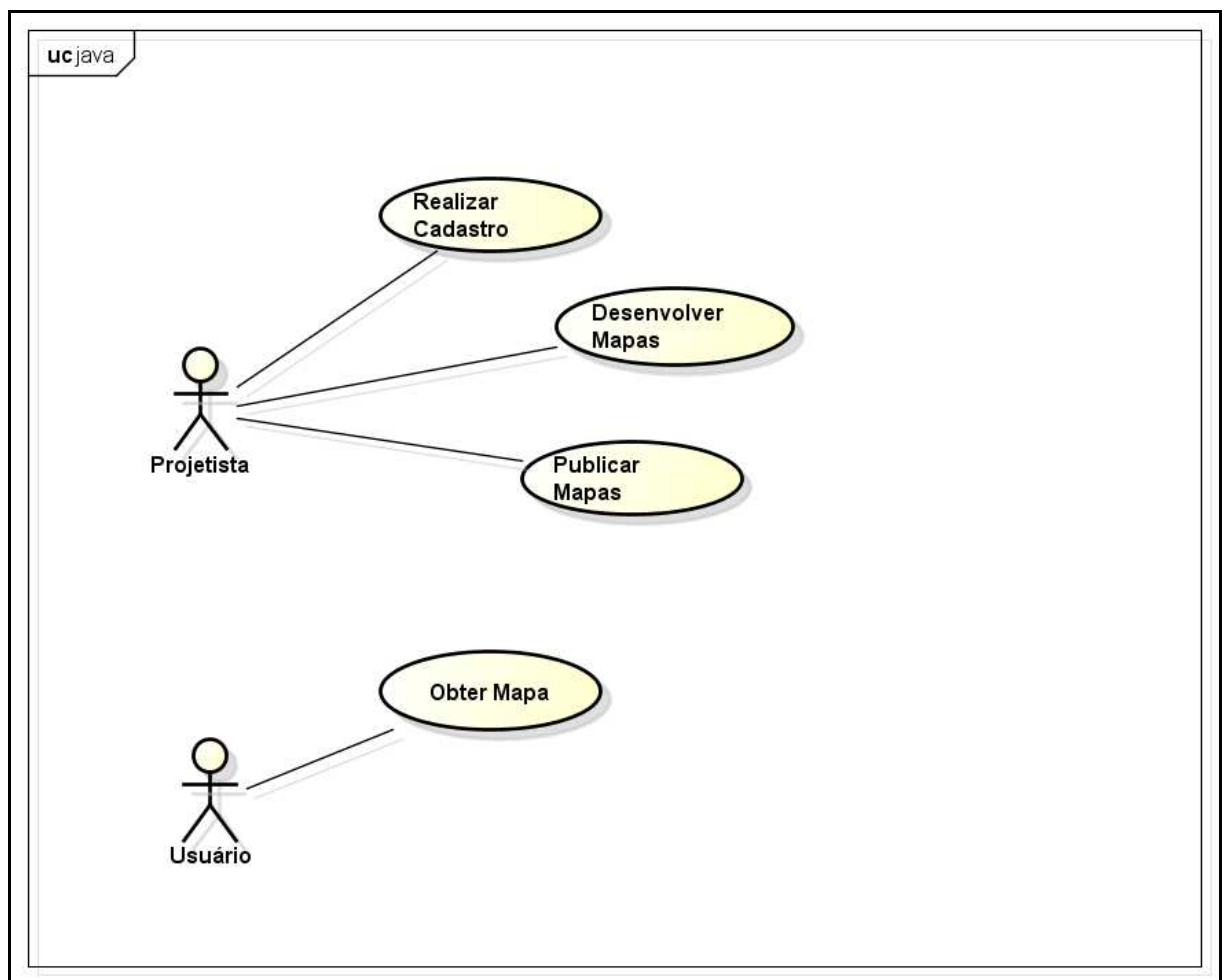


Figura 6 - Diagrama de Caso de Uso

3.2.2. DESCRIÇÃO DOS CASOS DE USO

Os quadros de 1 a 4 apresentam a descrição dos Casos de Uso do sistema.

<i>Caso de Uso</i>	Realizar Cadastro
<i>Ator</i>	Projetista
<i>Pré-Condições</i>	Realizar acesso ao sistema
<i>Pós-Condições</i>	Uma nova conta de acesso será criada no sistema
<i>Fluxo de Eventos Principal</i>	<ol style="list-style-type: none"> 1. Projetista seleciona a opção Cadastre-se no menu principal; 2. Projetista informa os dados para o novo usuário; 3. Projetista confirma a criação clicando no botão Cadastrar Novo Usuário.
<i>Fluxo de Eventos Secundário</i>	<ol style="list-style-type: none"> 1. A operação não é confirmada. 2. O sistema apresenta mensagem de erro. 3. O sistema retorna ao passo 2 do fluxo principal.

Quadro 1 - Descrição do Caso de Uso Realizar Cadastro

<i>Caso de Uso</i>	Desenvolver Mapas
<i>Ator</i>	Projetista
<i>Pré-Condições</i>	Estar logado no sistema
<i>Pós-Condições</i>	Um novo mapa será criado no sistema
<i>Fluxo de Eventos Principal</i>	<ol style="list-style-type: none"> 1. Projetista seleciona a opção Criar Novo Mapa nas opções do seu perfil; 2. Projetista informa os dados para o novo mapa; 3. Projetista confirma a criação e publicação do novo mapa clicando no botão Publicar Mapa;
<i>Fluxo de Eventos Secundário</i>	<ol style="list-style-type: none"> 1. A operação não é confirmada. 2. Projetista clica no botão Salvar Mapa para Edição, para

	<p>publicar o mapa em outro momento.</p> <p>3. O sistema apresenta mensagem de erro.</p> <p>4. O sistema retorna ao passo 2 do fluxo principal.</p>
--	---

Quadro 2 - Descrição do Caso de Uso Desenvolver Mapas

<i>Caso de Uso</i>	Publicar Mapas
<i>Ator</i>	Projetista
<i>Pré-Condições</i>	Realizar acesso ao sistema e estar autenticado no mesmo
<i>Pós-Condições</i>	Um mapa criado no sistema será disponibilizado para download
<i>Fluxo de Eventos Principal</i>	<p>1. Projetista seleciona a opção Mapas Não Publicados nas opções do seu perfil;</p> <p>2. Projetista localiza o mapa desejado e clica no botão Continuar Edição;</p> <p>3. Projetista confirma a publicação do novo mapa clicando no botão Publicar Mapa;</p>
<i>Fluxo de Eventos Secundário</i>	<p>1. A operação não é confirmada.</p> <p>2. O sistema apresenta mensagem de erro.</p> <p>3. O sistema retorna ao passo 2 do fluxo principal.</p>

Quadro 3 - Descrição do Caso de Uso Publicar Mapas

<i>Caso de Uso</i>	Obter Mapa
<i>Ator</i>	Usuário
<i>Pré-Condições</i>	Realizar acesso ao sistema
<i>Pós-Condições</i>	O mapa selecionado será baixado do sistema
<i>Fluxo de Eventos Principal</i>	<p>1. Usuário localiza o mapa desejado na página principal do sistema;</p> <p>2. Usuário clica no botão Download para obter o mapa;</p> <p>3. Usuário salva o mapa no diretório desejado.</p>
<i>Fluxo de Eventos Secundário</i>	<p>1. O sistema apresenta mensagem de erro.</p> <p>3. O sistema retorna ao passo 1 do fluxo principal.</p>

Quadro 4 - Descrição do Caso de Uso Obter Mapa

3.2.3. DIAGRAMA ENTIDADE RELACIONAMENTO

O diagrama entidade relacionamento apresenta os relacionamentos entre as tabelas físicas presentes no banco de dados e se mostra importante para o entendimento dos objetivos do sistema (BRYLA, 2009).

Para o desenvolvimento do sistema foi necessário a criação de oito tabelas, apresentadas no diagrama da Figura 7.

Os dados de todos os usuários que forem cadastrados no sistema ficarão salvos na tabela Usuario. Assim como os dados dos mapas ficarão na tabela Mapa e das etiquetas RFID na tabela RFID.

A tabela Mapa está diretamente relacionada às tabelas Usuario, Predio, Idioma e PontoInteresse, pois um mapa é criado por um usuário, em determinado idioma e possui os pontos de interesses cadastrados para o prédio desejado.

A tabela PontoInteresse armazena informações de pontos importantes do mapa como salas, escadas e rampas de acesso. Além do relacionamento com a tabela Mapa a tabela PontoInteresse está relacionada as tabelas RFID e ListaPontoEncadeado, pois no piso tátil colocado em frente a um banheiro, por exemplo, podem haver várias etiquetas RFID e próximo a esse banheiro podem existir vários pontos importantes. O relacionamento com a tabela ListaPontoEncadeado permite informar ao usuário a distância existente entre o local onde a etiqueta RFID foi lida e os pontos de interesse mais próximos, independente do sentido percorrido pelo usuário.

O APÊNDICE A apresentada a descrição dos campos de cada tabela utilizada.

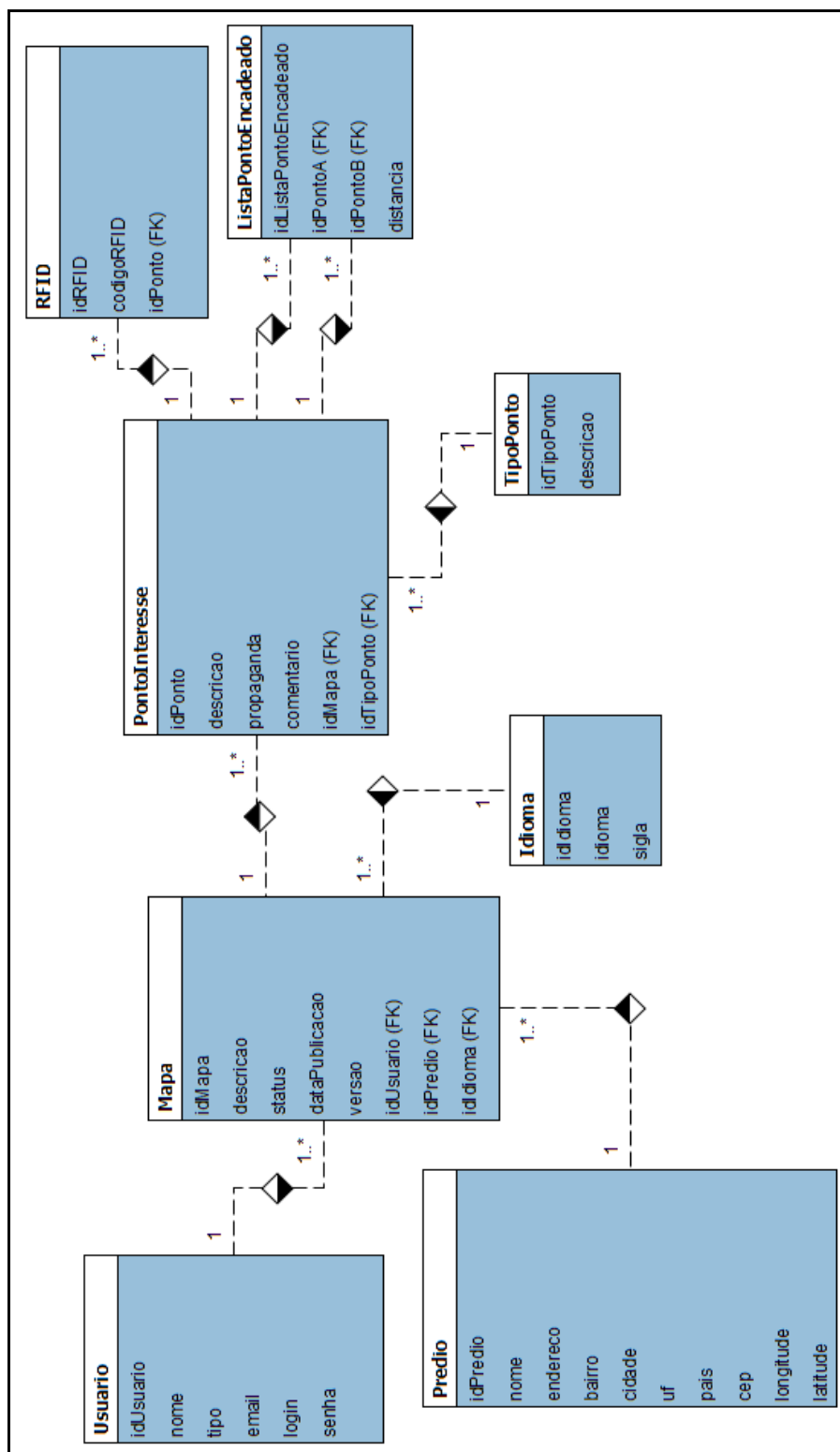


Figura 7 - Diagrama Entidade Relacionamento

4. IMPLEMENTAÇÃO

Esse capítulo apresenta a interface do sistema proposto, os testes realizados no mesmo e trata de aspectos das ferramentas e métodos utilizados no desenvolvimento.

4.1. FERRAMENTAS DE DESENVOLVIMENTO

Ao pesquisar sobre desenvolvimento de aplicações *web* vemos que a criação desse tipo de sistema pode ser realizada através de várias linguagens de programação. Para a construção do sistema foi escolhida a linguagem PHP e para auxiliar no desenvolvimento a ferramenta *Zend Framework*.

Além de ser um *framework* gratuito para o desenvolvimento de aplicações *web*, o *Zend Framework* possui fácil instalação, documentação abundante e suporte de uma empresa reconhecida, a *Zend Technologies*.

O ZF é voltado para o desenvolvimento em PHP e a versão utilizada nesse projeto é a 1.12.6 *FULL*, que está disponível para download no site oficial da aplicação.

Outro *framework* utilizado foi o *Twitter Bootstrap*. Conforme já explicado nesse projeto, o *Twitter Bootstrap* é um *framework fron-end*. Ele provê ferramentas para agilizar e facilitar a criação da interface entre o usuário e o sistema. A versão utilizada é a 3.3.2, que fornece suporte a recursividade e customizações.

Durante o processo de desenvolvimento não foi necessário customizar o código fonte do *Bootstrap*, porém seus componentes foram utilizados de acordo com as funcionalidades do sistema. O *layout* selecionado, por exemplo, foi adaptado para permitir que o visitante do sistema possa usufruir de tudo que o sistema fornece.

No site oficial do *Bootstrap* estão disponíveis gratuitamente alguns modelos de apresentação de páginas, chamados *templates*. Esses *templates* podem ser alterados para

suprir as necessidades de cada projeto, ou *versões* mais elaborados podem ser compradas em *sites* como o <https://wrapbootstrap.com>.

O modelo de apresentação utilizado foi o “*Basic marketing site*” que é fornecido gratuitamente no site do *Bootstrap*. O site do *Twitter Bootstrap* fornece ainda uma documentação de sua base CSS, mostrando os principais elementos HTML como tabelas, formulários e botões. São apresentados exemplos de codificação e imagens com o resultado esperado.

O *NetBeans* IDE e o *WampServer* são dois outros *softwares* usados no processo de criação do sistema. Ao instalar o *WampServer* 2.4 automaticamente foram instaladas as últimas versões do Apache, PHP e MySQL. Já a versão 7.4 do ambiente de desenvolvimento *NetBeans* IDE foi instalada com o intuito agilizar o processo de desenvolvimento, visto que essa ferramenta fornece suporte a frameworks.

Durante a criação do projeto no *NetBeans* foi necessário informar que o *Zend Framework* foi o framework PHP escolhido para o desenvolvimento, conforme apresentado na Figura 8. Através dessa informação o IDE começou a fornecer todo um suporte que não está disponível em outros ambientes de desenvolvimento integrado.

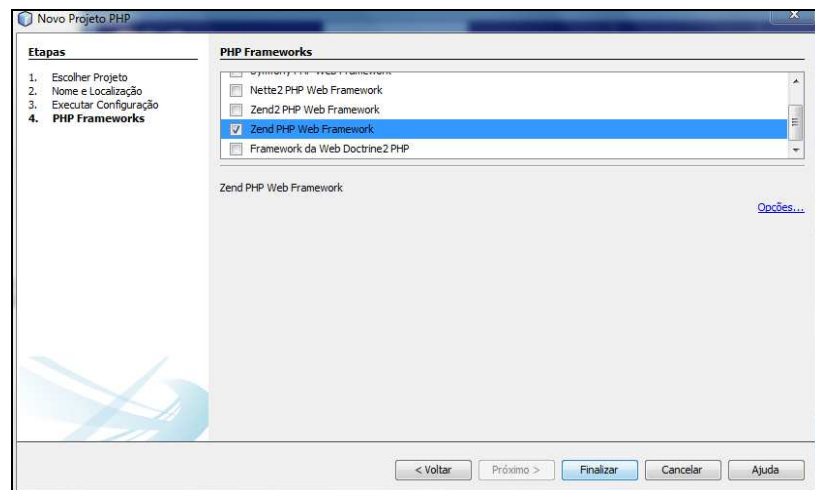


Figura 8 – Etapa de seleção de framework na criação de projeto no *Netbeans* IDE

Automaticamente após esse processo de criação a estrutura padrão de arquivos do ZF é criada dentro do projeto. A Figura 9 apresenta a estrutura do sistema.

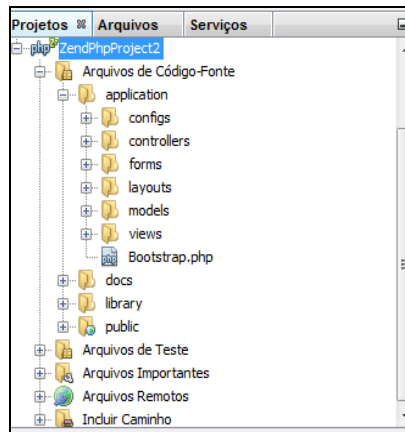


Figura 9 – Estrutura de pastas do sistema no Netbeans IDE

Dentre as pastas visualizadas na figura acima, a pasta Arquivos de Código-Fonte é a mais utilizada durante todo o processo de criação do sistema. Nela está presente a pasta *Application*, que contém toda a programação que não é visualizada pelos usuários da aplicação, a pasta *Docs* que recebe a documentação gerada, a pasta *Library* que armazena as bibliotecas utilizadas e a pasta *Public* que possui os arquivos que serão visualizados externamente como arquivos JavaScript, CSS e imagens.

O diretório *Application* por sua vez possui seis pastas fundamentais para o correto funcionamento do sistema:

- *Configs* – onde estão os arquivos de configuração do sistema.
- *Controllers* – possui os arquivos que definem o comportamento do sistema.
- *Forms* – possui os formulários criados. A pasta *Layouts* contém os *scripts* que definem o *layout* padrão do sistema.
- *Model* – possui os arquivos responsáveis pela manipulação de informações presentes no banco de dados.
- *View* – possui os *scripts* das páginas que poderão ser acessadas pelos usuários do sistema.

O ZF utiliza o MVC de forma avançada e deixa isso evidente na nomenclatura usada em suas pastas.

Conforme explicado na Sessão 2.3 do Capítulo 2, o padrão MVC divide a responsabilidades das aplicações nas camadas Modelo, Visão e Controle. Na estrutura do *Zend Framework* a camada Modelo do MVC é formada pelas classes *Zend_Db* e *Zend_Service*, a Visão é representada pelo *Zend_View* e o *Controller* pelo *Zend_Controller*.

Uma das vantagens do ZF é que a manipulação de banco de dados é realizada por meio de objetos. No desenvolvimento do projeto a forma mais simples da classe *Zend_Db_Table_Abstract* foi utilizada.

Para cada uma das oito tabelas apresentadas na Sessão 3.2.3 do capítulo anterior foi criada uma classe que estende a classe *Zend_Db_Table_Abstract*. Para que as tabelas fossem manipuladas pelos *Controllers* foi necessário informar os dados do banco no arquivo de configuração *Application.ini*, conforme a Figura 10.

```
14  
15     resources.db.adapter = "PDO_MySQL"  
16     resources.db.params.host = "localhost"  
17     resources.db.params.dbname = "bd_tcc"  
18     resources.db.params.username = "root"  
19     resources.db.params.password = ""  
20     resources.db.params.charset = utf8  
21     resources.db.isDefaultTableAdapter = true  
22
```

Figura 10 – Parte do código do arquivo *Application.ini*

No código apresentado na figura acima o adaptador *PDO_MySQL* é definido na linha 15, pois esse foi o sistema de banco de dados utilizado no sistema. A definição do nome do *host*, nome do banco de dados, nome de usuário e senha de acesso ao banco ocorre nas linhas 16, 17, 18 e 19 respectivamente. Já na linha 21 está configurando o adaptador criado como padrão, enquanto na linha 20 é definida a codificação *utf8*.

Além de manipular o banco de dados, os *Controllers* recebem as entradas dos usuários, fazer a atualizações das *views*, dentre outras funções.

As classes que estendem de *Zend_Controller* são constituídas por *actions*, conforme código apresentado na Figura 11. As *actions* representam a lógica de negocio ter um arquivo *view* associado, ou seja, uma *action* pode apresentar ao usuário final as informações que foram processadas.

Na Figura 11 há um fragmento de código da classe *MapaController*, que manipula a lógica de negócio do cadastro de mapas. Na linha 9 a variável *_dbmapa* recebe uma instância da classe *Application_Model_DbTable_Mapas*, que permite acesso a tabela *Mapas* do banco de dados. Na linha 13 encontra-se a função que cria a função *createAction*. A variável

`$predioform` recebe uma nova instância da classe de formulário `Application_Form_Mapas` na linha 15.



```

1  <?php
2
3  class MapasController extends Zend_Controller_Action
4  {
5
6      public function init()
7      {
8          ...
9          $this->_dbmapas = new Application_Model_DbTable_Mapas();
10         ...
11     }
12     ...
13     public function createAction()
14     {
15         $predioform = new Application_Form_Mapas();
16
17         $predios = $this->_dbpredio->getAdapter()->fetchPairs(
18             $this->_dbpredio->select()->from('predio', array('idPredio', 'nome'))->order('nome'));
19         $predioform->predios->setMultiOptions($predios);
20         ...
21
22         $this->view->form = $predioform;
23     }

```

Figura 11 – Parte do código do controlador Mapas

Nas linhas de 17 a 19 a variável `$predios` é criada com uma lista de identificador e nome de todos os prédios cadastrados no banco. Isso é possível porque o método `fetchPairs` pode ser utilizado para buscar informações da `DbTable`.

Por fim, na linha 22 o formulário é enviado para apresentação na *view* da *action create*.

4.2. FORMULÁRIOS

O *framework Zend* surgiu do desejo de reduzir o esforço realizado no desenvolvimento de aplicações *web* na linguagem PHP. Ele oferece vários componentes para tornar o desenvolvimento uma tarefa mais fácil e rápida.

Os desenvolvedores que desejam criar novos documentos PDF e editar documentos existentes fazem uso do componente `Zend_Pdf`. Os que querem enviar e-mail usam o `Zend_Mail` e para aqueles que desejam fácil acesso às APIs de serviços *web* há o `Zend_Service_Amazon`, `Zend_Service_Flickr` e `Zend_Service_Yahoo`.

Algumas facilidades do *Zend Framework* foram estudadas e implementadas no sistema, dentre elas destaca-se o *Zend_Form*. Como o cadastro de informações é uma ação frequente no sistema, usou-se o componente *Zend_Form* para simplificar a manipulação de formulários.

Ao associar *Zend Framework* e *NetBeans* a utilização dos componentes do *Zend* torna-se ainda mais fácil, pois o *NetBeans* fornece a janela Executar Zend Comando, conforme a Figura 12.

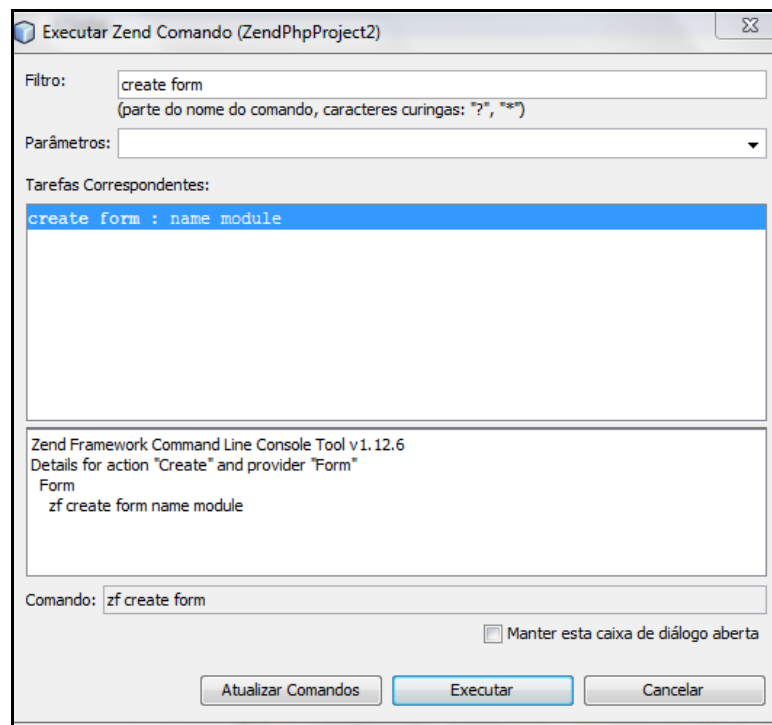


Figura 12 – Janela Executar *Zend* Comando do *NetBeans*

Para visualizar essa janela é necessário clicar com o botão direito do mouse sobre o projeto *Zend*, selecionar a opção *Zend* e então Executar comando. O comando *zf create form* deve ser executado para criar um formulário que estenda *Zend_Form*. Caso o desenvolvedor não saiba qual comando utilizar, basta realizar uma pesquisa através do nome do componente.

Após execução do comando *zf create form* é gerado um arquivo php independente na pasta *forms* presente na pasta *Application*. Esse arquivo não faz menção aos comandos comuns de formulários HTML e funciona como arquivo de configuração, indicando para o *Zend* quais modificações o formulário padrão deve sofrer.

Em um mesmo projeto é possível fazer uso de formulários que utilizam a classe *Zend_Form* e de formulários criados manualmente. No formulário criado pelo comando do *Zend* é possível adicionar outros elementos do ZF como *Zend_Validate*, *Zend_Filter* e *Zend_Loader*.

Zend_Validate fornece um conjunto de validadores comumente necessários. Um dos seus métodos, o *isValid()*, realiza a validação sobre o valor fornecido retornando *true* se o valor passa contra os critérios de validação. *Zend_Filter* é o componente que fornece funções de filtragem de string, enquanto *Zend_Loader* inclui métodos para ajudar a carregar os arquivos de forma dinâmica.

A Figura 13 mostra uma parte da classe do formulário Mapa.php, criada no projeto para gerenciar o cadastro de novos mapas. Na linha 3 a classe está estendendo a classe *Zend_Form*, tornando disponíveis os recursos do componente. Na linha 7 é atribuído valor para a *action* do formulário. Na linha 9 o método *post* é definido como método de envio dos dados. Já na linha 11 o atributo *id* é criado e recebe o valor *form-mapa*.

```

1  <?php
2
3  class Application_Form_Mapa extends Zend_Form {
4
5      public function init() {
6
7          $this->setAction($this->getView()->url(array('controller' => 'mapa', 'action' => 'add-mapa'), null, TRUE));
8
9          $this->setMethod('post');
10         /
11         $this->setAttrib('id', 'form-mapa');
12
13         $descricao = new Zend_Form_Element_Text('descricao');
14         $descricao->setLabel('Descrição');
15         $descricao->addFilter('StringTrim');
16         ...
17
18         $cancelar = new Zend_Form_Element_Submit('Cancelar',
19             array('class' => 'btn btn-primary'));
20         $cancelar->removeDecorator('DtDdWrapper');
21         ...
22         $this->addElements(
23             array(
24                 $descricao,
25                 $predios,
26                 $idiomas,
27                 $incluirpredio,
28                 $cancelar
29             )
30         );
31     }
32 }
33 
```

Figura 13 – Parte do código do formulário Mapa

Nas linhas 13, 14 e 15 é criado o campo descrição, através de um elemento de formulário de texto. O *label* do campo é definido na linha 14 através do comando *setLabel*. Na linha 15 o filtro *StringTrim* tem função de retornar o texto sem espaços em branco à direita e à esquerda. Nas linhas 18 e 19 o botão cancelar é criado. Além do *label* do botão, na no momento da criação também é informada sua classe de CSS. Na linha 20 o decorador padrão *DtDdWrapper* é removido do botão cancelar para que ele possa ser visualizado ao lado do

outro botão também criado nesse formulário. O método *addElement* da linha 22 adiciona todos os campos que devem ser apresentados no formulário.

O resultado do formulário de cadastro de mapa pode ser visualizado na Figura 20 presente na Sessão 4.3 desse capítulo.

O site oficial do *Zend* possui documentação dos componentes disponíveis, para melhor compreensão da utilização. Apesar de estar na língua inglesa, a documentação possui uma rica variedade de exemplos para que o desenvolvedor possa usufruir de todos os recursos necessários.

4.3. PROJETO DA INTERFACE

A interface do sistema, cujas principais telas serão exibidas a seguir, foi desenvolvida seguindo várias das orientações do eMAG, o Modelo de Acessibilidade em Governo Eletrônico apresentado na sessão 2.1 do capítulo 2.

Antes de chegar ao resultado final foram realizados vários protótipos do sistema para validação, seguindo assim o paradigma de prototipação apresentado na Sessão 2.2 do capítulo 2. O orientador desse trabalho, David Silva, realizou o papel de cliente e avaliou o que poderia ser mantido e o que precisava ser alterado nos protótipos.

A Figura 14 mostra o resultado da página que será apresentada quando os usuários realizarem acesso ao sistema.



Figura 14 – Tela Inicial

No canto superior direito da tela há *links* para acessar perfil no sistema e para realizar cadastro. Já no canto superior esquerdo é possível acessar o mapa do site e saber mais o ele.

Na parte central da página é possível realizar *download* dos arquivos publicados, que serão chamados de mapas. Quando um usuário finaliza a inclusão de informações de suporte à locomoção, um arquivo de extensão SQL, ou seja, um banco de dados é salvo em uma pasta do servidor. Informações desse arquivo, como nome e endereço para acesso, são incluídas em um arquivo XML que serve de base para a criação de *links* para *download*. Também através das informações presentes no arquivo XML, é possível pesquisar o *link* do arquivo desejado. Isso é possível porque na pesquisa é utilizada a classe *DOMDocument* do PHP, que permite carregar arquivos XML e localizar informações nele.

A Figura 15, por outro lado, mostra a tela de acessar perfil no sistema quando o *logon* ainda não foi realizado. Para criar mapas e disponibilizá-los o usuário precisa informar seu nome de usuário e senha cadastrados previamente.

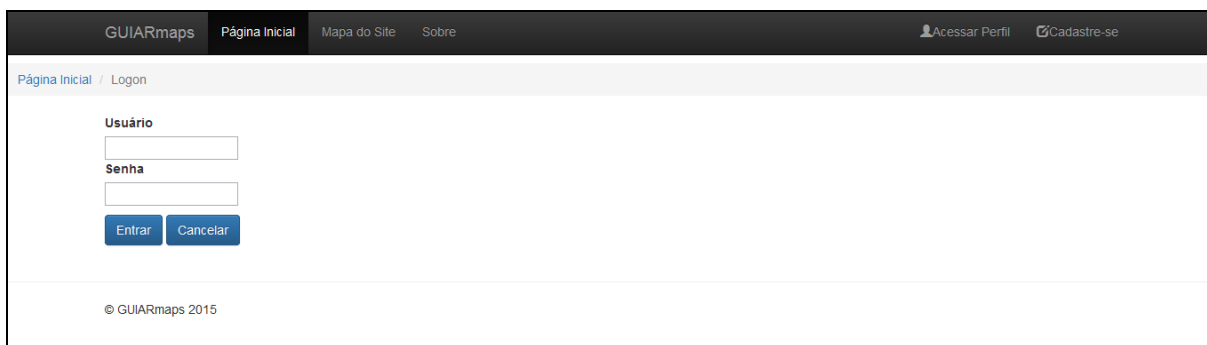


Figura 15 – Tela de *logon*

Aquele que ainda não possui nome de usuário e senha pode realizar seu cadastro no próprio sistema. A Figura 16 apresenta a tela onde os cadastros de novos usuários são realizados.

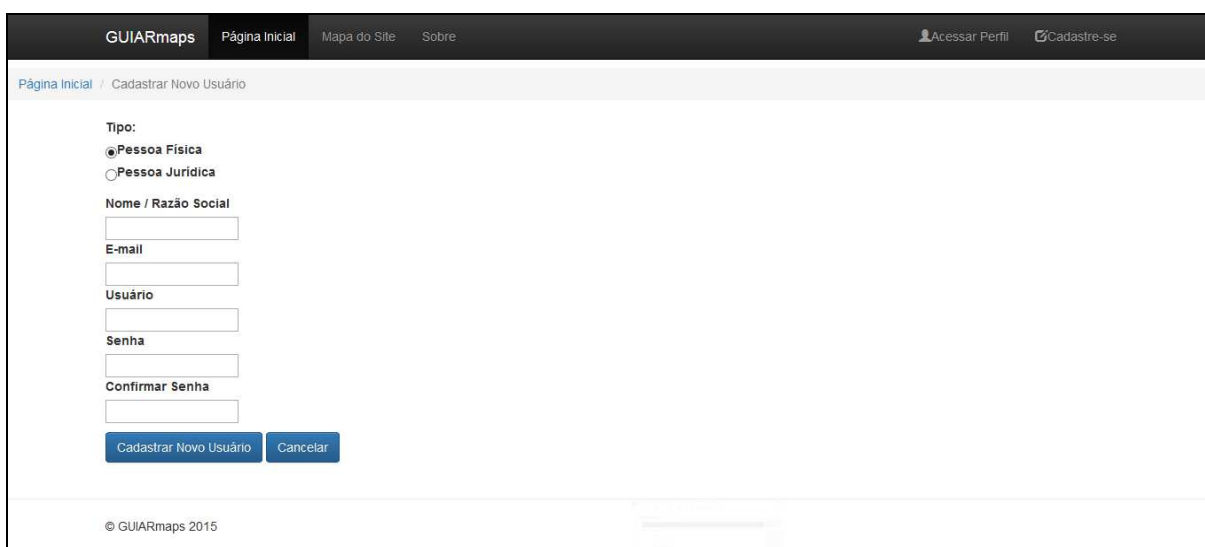


Figura 16 – Tela cadastro de usuários

A Figura 17 mostra a tela visualizada pelo usuário após a autenticação do seu acesso. Nela o usuário pode escolher qual função deseja realizar conforme pré-estabelecido pelo administrador do sistema.

A configuração inicial apresenta as seguintes funções: Criar novo mapa, Visualizar meus mapas e Mapas não publicados.



Figura 17 – Tela perfil dos usuários

A tela do perfil o usuário possui também a opção de alterar senha de acesso e e-mail cadastrado. As Figuras 18 e 19 apresentam a tela de mudança de senha e de e-mail respectivamente.

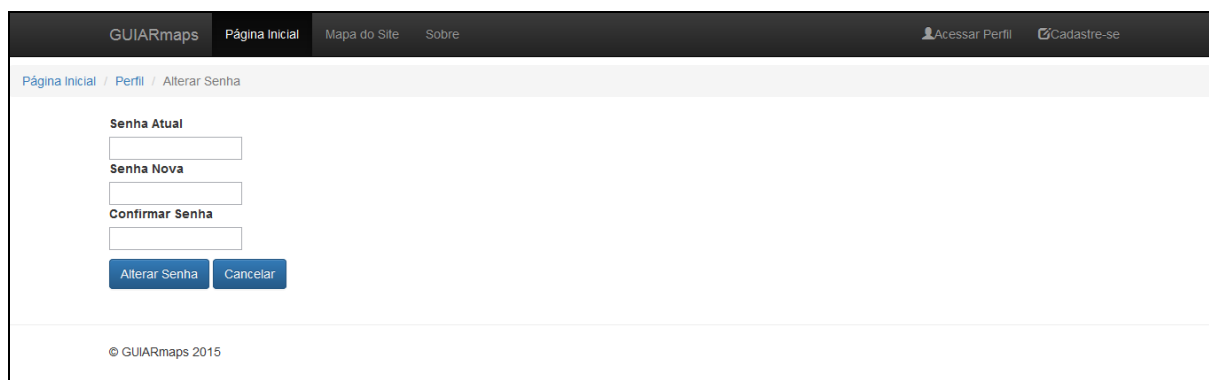


Figura 18 – Tela alteração de senha de usuário

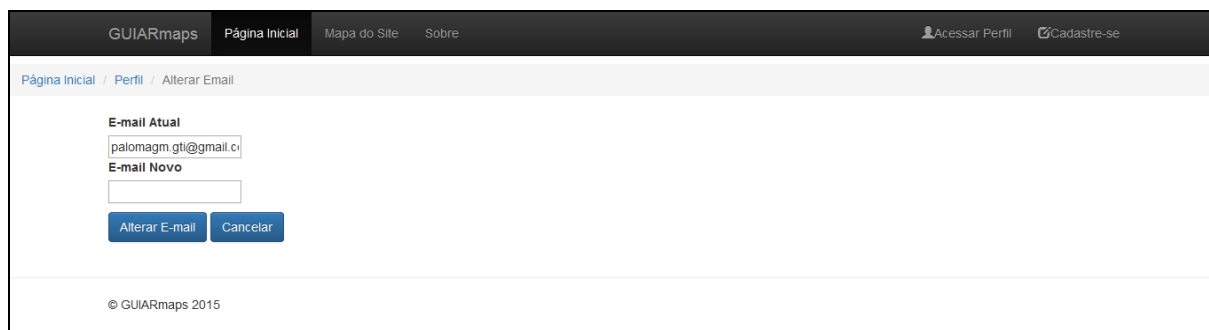


Figura 19 – Tela alteração de e-mail de usuário

Ao seleccionar a função Cadastrar Novo Mapa da tela do perfil será apresentado ao usuário um formulário conforme a Figura 20.

Figura 20 – Tela cadastro de mapa

Ao concluir o preenchimento do formulário de cadastro de mapa e clicar no botão Cadastrar Mapa a tela presente na Figura 21 será apresentada. Nessa tela é possível visualizar incluir e excluir pontos de interesse em um determinado mapa, alterar informações salvas na tela de cadastro de mapa, publicar o mapa criado e salvar o mesmo para edição posterior.

	Descrição	Tipo	Comentário	Propaganda	
<div> <div>+</div> <div>Cadastrar RFID</div> </div> <div> <div>+</div> <div>Associar Pontos</div> </div>	Escada do 2º andar	Escada	Escada do 2º andar que dá acesso ao 3º andar		<div> <div>✎</div> <div>Alterar</div> </div> <div> <div>🗑</div> <div>Excluir</div> </div>

Buttons at the bottom: Publicar Mapa, Salvar Mapa para Edição, Cancelar.

Figura 21 – Tela inclusão de Pontos de Interesse

O sistema permite cadastrar pontos como rampa, escada, banheiro, dentre outras classificações. Caso o usuário não localize o tipo de ponto desejado, o mesmo pode cadastrar

o tipo utilização o botão Incluir Tipo da tela acima que o levará a tela apresentada na Figura 22.

Figura 22 – Tela inclusão de Tipo de Ponto de Interesse

Na tela de perfil também há a opção Visualizar Meus Mapas e Mapas Não Publicados. Em Visualizar Meus Mapas o usuário poderá ver a lista de mapas que já desenvolveu, esteja ele publicado ou não. Já em Mapas Não Publicados é possível visualizar seus ainda não publicados e continuar criação dos mesmos, conforme apresentado na Figura 23.

Figura 23 – Tela de Mapas Não Publicados

4.4. TESTES

Rios (2013) esclarece que o processo de teste de software visa à execução do software em um ambiente controlado para avaliar seu comportamento com base no que foi especificado.

Na fase de construção do protótipo foram realizados diversos testes de codificação, para que falhas fossem corrigidas ainda no desenvolvimento. Após a correção das falhas

encontradas os testes foram realizados novamente para confirmar a execução satisfatória dos componentes criados.

Testes de sistema visam à execução de todo o sistema, ou subsistema, para validar a exatidão e perfeição durante a execução das suas funções (RIOS, 2013).

Para o nível de avaliação de qualidade do protótipo foi aplicado o questionário presente no APÊNDICE C. Silva (2003) destaca que dentre as vantagens da utilização de questionários há o fato de serem instrumentos de baixo custo de elaboração e aplicação; podem ser entregues em um primeiro momento e recolhidos após um determinado tempo, não interrompendo as atividades dos usuários; levam a uma maior precisão e estabilidade nas respostas, pois além dos os usuários se sentirem mais à vontade eles podem responder o questionário no seu tempo.

O modelo do questionário utilizado foi elaborado pela autora como base no modelo desenvolvido por Brustein (2007) para avaliação da aplicação de um software didático pelos alunos. Os itens presentes no questionário foram divididos em grupos com o intuito de medir vários aspectos do protótipo, incluindo características presentes no modelo de qualidade da norma internacional ISO/IEC 9126.

Após a aplicação dos questionários foram realizados cálculos para mensurar o nível de qualidade do protótipo. Os cálculos executados são baseados nos cálculos realizados por Brustein (2007) e na norma ISO/IEC 9126.

Os primeiros valores calculados são as médias das avaliações de cada subitem e então a média final de cada item.

A Tabela 1 mostra como devem ser distribuídas as respostas dos questionários para que sejam totalizados os pontos atribuídos a cada subitem por tipo de apreciação (Insuficiente, Regular, Bom e Excelente) e na coluna "Total" a soma total desses pontos. A coluna "Média Subitem" deve ser preenchida com as médias ponderadas de cada subitem. Para isso os subitens avaliados pelos níveis de pontuação Insuficiente, Regular, Bom e Excelente devem receber respectivamente os pesos 1, 2, 3 e 4, portanto, multiplica-se a quantidade de pontos aplicados em cada subitem pelos pesos de cada nível de pontuação e dividir pela quantidade total de pontuação presente na coluna "Total".

Item	Subitem	Insuficiente	Regular	Bom	Excelente	Total	Média Subitem
Aspectos Visuais	1.1						
	1.2						
	1.3						
Funcionalidade	2.1						
	2.2						
	2.3						
	2.4						
	2.5						
Confiabilidade	3.1						
	3.2						
Usabilidade	4.1						
	4.2						
	4.3						
	4.4						
	4.5						
Eficiência	5.1						
	5.2						

Tabela 1 – Tabela de total de pontuações e médias das avaliações

A média de cada item é a média simples das médias de seus subitens correspondentes. Já a média final é a média simples entre todos os itens. A Figura 24 mostra a escala utilizada no mapeamento da média final encontrada.

Valores	Classificação	
1,00 - 1,74	Insuficiente	Insatisfatório
1,75 - 2,49	Regular	
2,5 - 3,24	Bom	Satisfatório
3,25 - 4	Excelente	

Figura 24 – Níveis de pontuação

Fonte: Adaptado de Brustein (2007)

O resultado dos questionários aplicados considerou o protótipo satisfatório com classificação *Bom* em relação à amostra. O Gráfico 1 apresenta a média final de cada grupo dos itens avaliados.

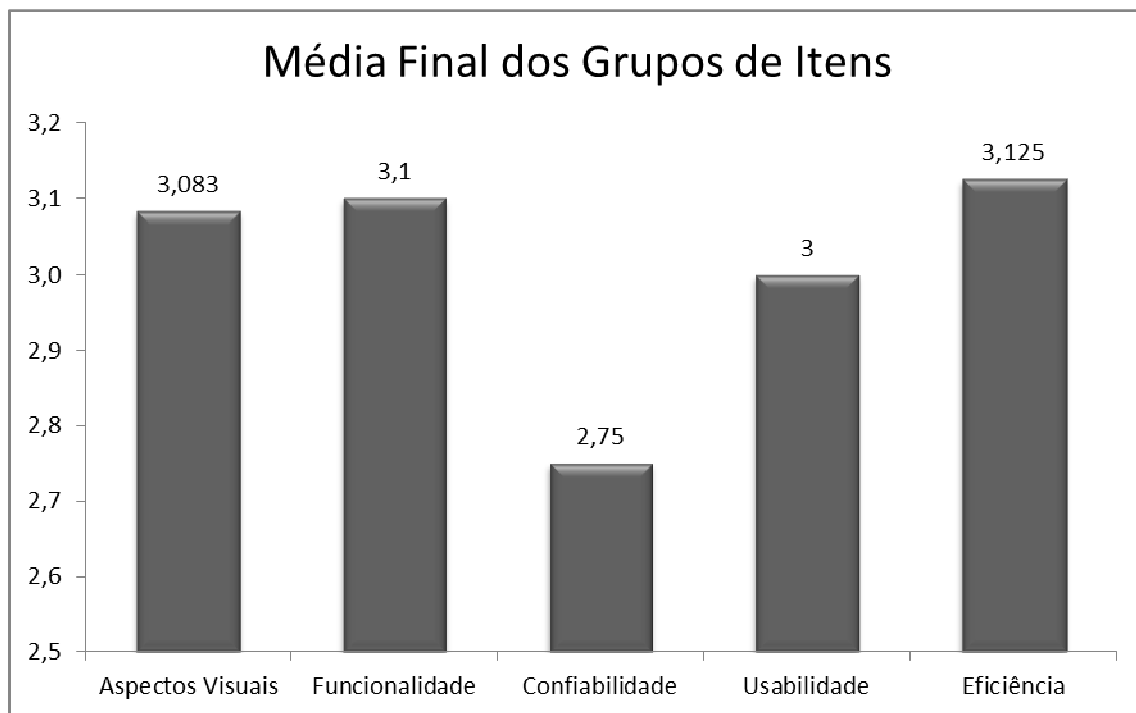


Gráfico 1 – Média final dos grupos de itens avaliados

Apesar de todos os grupos possuírem nível de pontuação *Bom*, os resultados mostram que o ponto forte do protótipo é a Eficiência, ou seja, foi considerado satisfatório o relacionamento entre o desempenho do protótipo e a quantidade de recursos usados. Já a Confiabilidade foi a característica com menor nível de pontuação, apontando assim uma possibilidade de melhoria na tolerância a falhas e recuperabilidade.

Outra importante avaliação realizada foi o de acessibilidade do protótipo. Conforme comentado anteriormente, várias orientações do eMAG foram seguidas no desenvolvimento, visto que o objetivo final da criação do sistema proposto é facilitar a mobilidade autônoma do deficiente visual, nada mais certo do que tornar o acesso independente ao sistema o mais fácil possível.

O protótipo desenvolvido neste trabalho foi criado seguindo normas de HTML e utilização, tendo como base o documento Padrões *Web* em Governo Eletrônico que fornecido pelo Ministério do Planejamento, Orçamento e Gestão. Já o CSS fornecido pelo *Twitter Bootstrap* segue normas compatíveis com as normas estabelecidas pelo W3C.

Recomendações como a 1.6 – Não utilizar tabelas para diagramação, 3.3 - Oferecer um título descritivo e informativo à página, 3.4 – Informar o usuário sobre sua localização na página, 3.5 – Descrever links clara e sucintamente são alguns exemplos utilizados no processo de criação do sistema.

A avaliação manual de acessibilidade do sistema foi realizada utilizando o *Checklist* de Acessibilidade Manual para o Desenvolvedor, disponível no endereço <http://www.governoeletronico.gov.br/acoes-e-projetos/e-MAG/material-de-apoio>, e desenvolvido em acordo com o Ministério do Planejamento, Orçamento e Gestão (MP), representado pela Subsecretaria de Planejamento, Orçamento e Administração e o Ministério da Educação, representado pela Secretaria de Educação Profissional e Tecnológica (SETEC).

O *checklist* foi desenvolvido com base em testes realizados com deficientes visuais e padrões W3C, com o objetivo de orientar o desenvolvedor a alcançar qualidade na acessibilidade, usabilidade e comunicabilidade de sites (CHECKLIST, 2014).

O Quadro 5 apresenta a primeira parte do *checklist* que trata da utilização de links e mostra os resultados obtidos na parte de cadastro de pontos de interesse.

Item	SIM	NÃO	N.A	Local onde não foi respeitado
O site fornece a localização do usuário em um conjunto de páginas?	X			
As âncoras estão sendo usadas corretamente?			X	
Há links indicadores na página?		X		
Os links apresentam descrições curtas e objetivas? Eles identificam o destino ao qual remetem? Abrem o conteúdo na mesma página de navegação ou avisam que irão abrir em uma nova página?	X			
Há atalhos para facilitar a navegação pelo site? Esses atalhos funcionam corretamente?	X			

Quadro 5 – Checklist de utilização de links

Algumas não conformidades foram identificadas após realização da avaliação do sistema e foram prontamente corrigidas. Através das perguntas presentes no *checklist* foram identificadas boas práticas que facilitam a utilização, como o uso de *breadcrumbs*, que fornecem a localização do usuário no sistema.

5. CONCLUSÕES

O avanço tecnológico trouxe um leque de soluções para suporte ao desenvolvimento de aplicações *web*. No desenvolvimento do protótipo confirmou-se que o *Zend Framework* é uma ferramenta confiável e bem documentada. A separação existente entre a parte lógica e a parte visual simplifica a realização de manutenções e a estrutura do *Zend* permite desenvolver aplicações *web* de forma flexível e organizada.

Em busca dos objetivos iniciais, chegou-se ao resultado de software que, em servidor local, possibilita a criação e disponibilização de mapas para suporte à locomoção de pessoas com deficiência visual. O protótipo pode ser alocado em servidor de rede e disponibilizado na rede mundial de computadores e a existência de informações como idioma, latitude e longitude torna admissível a implantação em outros países.

O trabalho evidencia a importância da busca por tecnologias para facilitar a locomoção autônoma de pessoas com deficiência visual e que a implantação do projeto de pesquisa GUIAR no Campus-Centro do IFF pode ajudar no deslocamento diário de dezenas de pessoas nas dependências da instituição.

5.1. TRABALHOS FUTUROS

Durante o desenvolvimento do protótipo identificou-se recursos interessantes que poderiam ser incorporados ao ambiente. Dessa forma, foram sugeridas funcionalidades a serem desenvolvidas em trabalhos futuros. Entre as funcionalidades, destacam-se:

- Internacionalizar todo o sistema, permitindo sua leitura nos principais idiomas existentes;
- Implantação de listas como: os mapas mais recentes, os mais baixados, dentre outras classificações que podem auxiliar os usuários na localização dos mapas desejados;
- Inclusão de campos de comentários para os mapas, que permitirá que os usuários expressem suas opiniões a respeito dos mapas publicados. Opiniões essas que podem ser utilizadas no processo de melhoria.

- Tornar o processo de inclusão de pontos de interesse mais visual, para que o usuário projetista possa visualizar graficamente como está o mapa.

REFERÊNCIAS BIBLIOGRÁFICAS

- BAPTISTELLA, Adriano José. Abordando a arquitetura MVC, e Design Patterns: Observer, Composite, Strategy. 2011. Disponível em: <<http://www.linhadecodigo.com.br/artigo/2367/abordando-a-arquitetura-mvc-e-design-patterns-observer-composite-strategy.aspx>>. Acesso em: 23 maio. 2015.
- BÖCK, Heiko. The Definitive Guide to NetBeans™ Platform 7. Apress, 2011.
- BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. UML: guia do usuário. Elsevier Brasil, 2006.
- BOOTSTRAP. Disponível em: <<http://getbootstrap.com/>>. Acesso em: 21 maio. 2014.
- BRUNSTEIN, Leo. Avaliação da Qualidade de Software Aplicado como Ferramenta de Apoio Didático com Base na Norma ISO 9126. 2007. Dissertação de Mestrado em Engenharia de Produção, Universidade Paulista – UNIP, São Paulo. Disponível em: <http://www3.unip.br/ensino/pos_graduacao/strictosensu/eng_producao/download/eng_leobrunstein.swf>. Acesso em: 20 maio. 2015.
- BRYLA, Bob; LONEY, Kevin. Oracle Database 11g: manual do DBA. Bookman, 2008.
- CARTILHA de Acessibilidade na Web - W3C Brasil - Fascículo 1: Introdução. Disponível em: <<http://acessibilidade.w3c.br/cartilha/fasciculo1/>>. Acesso em: 27 out. 2014.
- CHAVES, Aline Martins; DA SILVA, Gabriel. Proposta de uma arquitetura de software e funcionalidades para implementação de um ambiente integrado de desenvolvimento para a linguagem php. I Jornada Científica e VI FIPA do CEFET Bambuí, v. 31, p. 32-36, 2008. Disponível em: <http://www.cefetbambui.edu.br/str/artigos_aprovados/informatica/68-CO-5.pdf>. Acesso em: 27 out. 2014.
- CHECKLIST Manual de Acessibilidade - Desenvolvedores. Disponível em: <<http://www.governoeletronico.gov.br/biblioteca/arquivos/checklist-manual-de-acessibilidade-desenvolvedores/download>>. Acesso em: 27 out. 2014.
- CONALLEN, Jim. Building Web applications with UML. Addison-Wesley Longman Publishing Co., Inc., 2002.
- CONFORTO, Débora; SANTAROSA, Lucila MC. Acessibilidade à Web: Internet para todos. Revista de Informática na Educação: Teoria, Prática-PGIE/UFRGS, v. 5, n. 2, p. 87-102, 2002. Disponível em: <http://edu3051.pbworks.com/f/ACESSIBILIDADE_WEB_revista_PGIE.pdf>. Acesso em: 26 out. 2014.
- DALL'OGGIO, Pablo. PHP Programando com Orientação a Objetos. Novatec Editora, 2009.

E-MAG - Modelo de Acessibilidade em Governo Eletrônico. Disponível em: <<http://www.governoeletronico.gov.br/biblioteca/arquivos/emag-modelo-de-acessibilidade-em-governo-eletronico/download>>. Acesso em: 27 out. 2014.

FREIRE, André Pimenta. Acessibilidade no desenvolvimento de sistemas web: um estudo sobre o cenário brasileiro. 2008. Tese de Doutorado. Universidade de São Paulo. Disponível em: <<http://www.teses.usp.br/teses/disponiveis/55/55134/tde-06052008-101644/en.php>>. Acesso em: 17 mai. 2015.

GÄRTNER, Vilson Cristiano. Um Ambiente Integrado Para Apoio Ao Desenvolvimento Distribuído De Software. Dissertação de Pós-Graduação em Computação Aplicada, 85 páginas, Universidade do Vale do Rio dos Sinos, Porto Alegre, 2011 Disponível em: <http://idde.vgdata.net/Dissertacao_Final_Entregue_Correcao_Banca.pdf>. Acesso em: 9 jul. 2014.

HEDENGREN, Thord Daniel. Smashing WordPress Themes: Making WordPress Beautiful. John Wiley & Sons, 2011.

JOBSTRAIBIZER, F. Guia Profissional PHP. São Paulo: Digerati Books, 2009.

LISBOA, FLAVIO GOMES DA SILVA. Zend Framework: desenvolvendo em PHP 5 orientado a objetos com MVC. Novatec Editora, 2008.

LOBATO, Fernanda Hoffmann; OLIVEIRA, João Batista Ferri de; SILVA, Thiago Augusto dos Santos. Modelo de Acessibilidade em Governo Eletrônico. XVII Congreso Internacional del CLAD sobre la Reforma del Estado y de la Administración Pública, Cartagena, Colombia, 2012. Disponível em: <<http://www.dgsc.go.cr/dgsc/documentos/cladxvii/lobatofe.pdf>>. Acesso em: 20 maio. 2015.

MEIRELES, A. M. G. Auxílio na Orientação de Invisuais Usando a Tecnologia RFID (SmartVision). Dissertação de Mestrado em Engenharia Electrotécnica e Computadores, 85 páginas, Universidade De Trás-Os-Montes e Alto Douro, Vila Real, 2009. Disponível em: <https://repositorio.utad.pt/bitstream/10348/348/1/msc_amgmeireles.pdf>. Acesso em: 9 jul. 2014.

MELO, Alexandre Altair de, and Mauricio GF NASCIMENTO. PHP Profissional: Aprenda a desenvolver sistemas profissionais orientados a objetos com padrões de projeto. Editora Novatec, 2007.

MENDES, Douglas Rocha. Redes de Computadores. Editora Novatec, 2007.

MINETTO, Elton Luís. Frameworks para Desenvolvimento em PHP. Editora Novatec, 2007.

MOURATO, Luis Patrício; PIRES, Glauber Magalhães; D'EMERY, Richarlyson A. Model View Controller E Facade: Um Estudo De Caso. IX Jornada de Ensino, Pesquisa e Extensão, JEPEX. Universidade Federal Rural de Pernambuco. 2009. Disponível em: <<http://www.eventosufrpe.com.br/eventosufrpe/jepex2009/cd/resumos/R1091-1.pdf>>. Acesso em: 23 maio. 2015.

MUÑOZ, Daniel Mauricio et al. Robô Cão Guia Para Auxílio A Deficientes Visuais Em Recintos Fechados. VI Congresso Nacional De Engenharia Mecânica, 2010. Disponível em: <<http://www.abcm.org.br/pt/wp-content/anais/conem/2010/PDF/CON10-1860.pdf>>. Acesso em: 4 mar. 2015.

NETBEANS IDE - Overview. Disponível em: <<https://netbeans.org/features/index.html>>. Acesso em: 12 jul. 2014.

NIEDERAUER, Juliano. Desenvolvendo websites com PHP. Editora Novatec, 2011.

OLIVEIRA, Cícero David Leite. Desenvolvimento de componentes WEB Services disponibilizados por meio de WEB Servers. 14º Encontro de Iniciação Científica e Pós-Graduação do ITA – XIV ENCITA. 2008. Disponível em: <<http://www.bibl.ita.br/xivencita/COMP01.pdf>>. Acesso em: 10 jul. 2014.

OTTO, MARK. Bootstrap from Twitter, 2011. Disponível em: <<https://blog.twitter.com/2011/bootstrap-twitter>>. Acesso em: 11 jul. 2014.

PRESSMAN, Roger S. Engenharia de software. McGraw Hill Brasil, 2011.

PRESSMAN, Roger S. Engenharia de Software. Makron Books, 2007.

PRIKLADNICKI, Rafael; AUDY, Jorge Luis Nicolas. Desenvolvimento distribuído de software. Elsevier, 2008.

RAMOS, Ricardo Argenton. Treinamento prático em UML. Universo dos Livros Editora, 2006.

RANTHUM, Rogério. Modelagem E Implementação De Um Sistema De Informação Para Otimização De Exames De Diagnósticos Por Imagens. Dissertação de Pós-Graduação em Engenharia de Produção. Universidade Tecnológica Federal Do Paraná. 2005. Disponível em: <<http://www.pg.utfpr.edu.br/dirppg/ppgep/dissertacoes/arquivos/13/Dissertacao.pdf>>. Acesso em: 20 maio. 2015.

REZENDE, D. A. Engenharia de Software e Sistemas de Informação. Brasport, 2005.

RUBINOFF, JEFF. Suporte a Zend Framework no NetBeans IDE para PHP. Disponível em: <https://netbeans.org/kb/docs/php/zend-framework-screencast_pt_BR.html>. Acesso em: 9 out. 2014.

SCHMITZ, Daniel. Criando Sistemas RESTful com PHP e jQuery: Uma abordagem prática na criação de um sistema de vendas. Novatec Editora, 2013.

SERPRO. Portal do SERPRO - Zend Framework, 2008. Disponível em: <http://www4.serpro.gov.br/imprensa/publicacoes/tema-1/antigas%20temas/tema_193/materias/zend-zramework>. Acesso em: 09 jul. 2014.

SILVA, David Vasconcelos Corrêa da et al. Desenvolvimento De Um Protótipo De Sistema Automatizado Para Locomoção Autônoma De Deficientes Visuais. V Congresso Fluminense de Iniciação Científica e Tecnológica, 2013.

SILVA, Simone Vasconcelos. Qualidade de Software–Uma Abordagem Baseada na Satisfação do Usuário. 2003. Dissertação de Mestrado em Ciências de Engenharia –Área de Engenharia de Produção, UENF, Campos dos Goytacazes–RJ. Disponível em: <http://uenf.br/Uenf/Downloads/POS-ENGPRODUCAO_2397_1213624178.pdf>. Acesso em: 20 maio. 2015.

SKVORC, B. Best PHP Frameworks for 2014, SitePoint, 28 dez. 2013. Disponível em: <<http://www.sitepoint.com/best-php-frameworks-2014/>>. Acesso em: 9 jul. 2014

SPURLOCK, Jake. Bootstrap. O'Reilly Media, Inc., 2013.

UFSM , eMAG - Modelo de Acessibilidade em Governo Eletrônico - Núcleo de Acessibilidade UFSM. Disponível em: <<http://w3.ufsm.br/acessibilidade/index.php/noticias/50-emag-modelo-de-acessibilidade-em-governo-eletronico>>. Acesso em: 20 maio. 2015.

WAMPSEVER. Disponível em: <<http://www.wampserver.com/en/>>. Acesso em: 12 jul. 2014.

ZEND Framework. Disponível em: <<http://framework.zend.com/>>. Acesso em: 13 maio. 2014.

APÊNDICES

APÊNDICE A: ESTRUTURA DO BANCO DE DADOS

A estrutura do banco de dados utilizado no sistema, cujas relações foram apresentadas na Figura 7, está descrita conforme os quadros de 6 a 13.

Quadro 6: Usuario – apresenta as informações das pessoas, ou entidades, cadastradas no sistema.

Campo	Tipo de Dados	Descrição
IdUsuario	Numérico	Identificador único do usuário
Nome	Alfanumérico	Nome/razão social do usuário
Tipo	Alfanumérico	Se o usuário é pessoa física ou jurídica
Email	Alfanumérico	Email de contato do usuário
Login	Alfanumérico	Nome utilizado para se autenticar no sistema
Senha	Alfanumérico	Senha utilizada na autenticação no sistema

Quadro 6 - Descrição da tabela Usuario

Quadro 7: Predio – descreve informações dos prédios cadastrados no sistema.

Campo	Tipo de Dados	Descrição
IdPredio	Numérico	Identificador único do prédio
Nome	Alfanumérico	Nome dado ao prédio
Endereço	Alfanumérico	Endereço de localização do prédio
Bairro	Alfanumérico	Bairro onde está localizado o prédio
Cidade	Alfanumérico	Cidade onde está situado o prédio
Uf	Alfanumérico	Estado brasileiro encontra-se o prédio país
País	Alfanumérico	País o prédio está localizado
Cep	Alfanumérico	Código de endereço postal do prédio
Longitude	Alfanumérico	Coordenada de longitude da localização do prédio

Campo	Tipo de Dados	Descrição
Latitude	Alfanumérico	Coordenada de latitude da localização do prédio

Quadro 7 - Descrição da tabela Predio

Quadro 8: Mapa – armazena as informações do mapa criado.

Campo	Tipo de Dados	Descrição
IdMapa	Numérico	Identificador único do mapa
Descrição	Alfanumérico	Título do mapa
Status	Alfanumérico	Status atual do mapa
dataPublicacao	Alfanumérico	Data em que o mapa foi publicado
Versão	Alfanumérico	Versão atual do mapa
IdUsuario	Numérico	Indica qual usuário desenvolveu o mapa
IdPredio	Numérico	Indica a qual prédio o mapa pertence
IdIdioma	Numérico	Indica qual o idioma do mapa

Quadro 8 - Descrição da tabela Mapa

Quadro 9: Idioma – armazena os idiomas em que um mapa pode ser desenvolvido.

Campo	Tipo de Dados	Descrição
IdIdioma	Numérico	Identificador único do idioma
Idioma	Alfanumérico	Título do idioma
Sigla	Alfanumérico	Armazena qual a sigla do idioma

Quadro 9 - Descrição da tabela Idioma

Quadro 10: PontoInteresse - armazena informações sobre os pontos de interesse dos prédios.

Campo	Tipo de Dados	Descrição
IdPonto	Numérico	Identificador único do ponto de interesse
Descrição	Alfanumérico	Título dado ao ponto de interesse

Campo	Tipo de Dados	Descrição
Propaganda	Alfanumérico	Propaganda cadastrada para o ponto
Comentário	Alfanumérico	Comentário cadastrado para o ponto
IdMapa	Alfanumérico	Indica a qual mapa o ponto de interesse pertence
idTipoPonto	Alfanumérico	Indica qual a natureza do ponto de interesse

Quadro 10 - Descrição da tabela PontoInteresse

Quadro 11: TipoPonto – armazena as classificações que os pontos de interesse podem ter.

Campo	Tipo de Dados	Descrição
idTipoPonto	Numérico	Identificador único do tipo do ponto de interesse
Descrição	Alfanumérico	Descreve a natureza do ponto de interesse

Quadro 11 - Descrição da tabela TipoPonto

Quadro 12: RFID – armazena informações das etiquetas RFID

Campo	Tipo de Dados	Descrição
IdRFID	Numérico	Identificador único da etiqueta RFID
codigoRFID	Alfanumérico	Código presente na etiqueta RFID
IdPonto	Numérico	Indica a qual ponto de interesse a etiqueta pertence

Quadro 12 - Descrição da tabela RFID

Quadro 13: ListaPontoEncadeado – armazena a relação entre dois pontos de interesse

Campo	Tipo de Dados	Descrição
idListaPontoEncadeado	Numérico	Identificador único da relação entre os pontos
IdPontoA	Numérico	Indica qual o primeiro ponto da relação
IdPontoB	Numérico	Indica qual o segundo ponto da relação
Distancia	Numérico	Distância entre os pontos de interesse

Quadro 13 - Descrição da tabela ListaPontoEncadeado

APÊNDICE B: IMPLANTAÇÃO DO PROTÓTIPO NO SERVIDOR

Para o correto funcionamento do protótipo, o servidor *Windows* deve possuir os de PHP, *Apache* e MySQL. Esses serviços podem ser facilmente obtidos através do *WampServer*.

O roteiro abaixo fornece suporte para preparar o servidor utilizando o *WampServer* e realizar a execução do protótipo:

Instalação e configuração do *WampServer*

1. Realize download do software de acordo com a versão do sistema operacional no endereço <http://www.wampserver.com/en/#download-wrapper>;
2. Executar o arquivo de download e seguir as orientações do processo de instalação.
3. Acessar o menu do *Wamp*, presente no canto inferior da tela, clicar no diretório *Apache*, e abrir o arquivo `httpd.conf`. Localizar a palavra *rewrite* e retirar o símbolo # presente antes do termo *LoadModule*.

Importação do banco de dados

1. Acessar o *phpMyAdmin* através do menu do *Wamp*;
2. Clicar na opção Importar do *phpMyAdmin* e através do botão Selecionar arquivo localizar o backup da base de dados do protótipo;
3. Clicar no botão para concluir a importação dos dados.

Importação do protótipo

1. A pasta do protótipo deve ser descompactada e salva na pasta WWW que foi criada no caminho instalação do *WampServer*, que por padrão é C:\wamp.
2. Acessar a pasta *public* do protótipo através do servidor apache do *Wamp*.

APÊNDICE C: QUESTIONÁRIO DE AVALIAÇÃO DE QUALIDADE

1. Aspectos Visuais

1.1. As telas apresentam somente informações necessárias e utilizáveis, sensíveis ao contexto?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

1.2. As telas apresentam contrastes e cores, facilitando a leitura?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

1.3. As telas exibem as mensagens com bom aspecto visual, utilizando, com moderação, negrito, itálico e sublinhado?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

2. Funcionalidade

2.1. A interface mantém uma padronização própria em relação a configuração de janela?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

2.2. A interface mantém uma padronização própria em relação a formatação de ícones?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

2.3. As funções implementadas no protótipo estão todas implementadas corretamente?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

2.4. As funções verificadas no protótipo geram resultados corretos ou conforme o esperado?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

2.5. O protótipo evita acesso não autorizado, acidental ou deliberado?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

3. Confiabilidade

3.1. Qual a frequência de falhas por defeitos que o protótipo apresenta?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

3.2. O protótipo mantém um nível de desempenho especificado nos casos de falhas no software ou violação nas interfaces especificadas?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

4. Usabilidade

4.1. Qual a frequência de erros gramaticais apresentados na interface?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

4.2. Qual a frequência de erros ortográficos apresentados na interface?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

4.3. O usuário reconhece com facilidade no protótipo o conceito lógico e sua aplicabilidade?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

4.4. O usuário aprende com facilidade a aplicação do protótipo?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

4.5. O usuário opera e controla cada operação do software com facilidade?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

5. Eficiência

5.1. É satisfatório o tempo de resposta do protótipo, o tempo de processamento e velocidade na execução de suas funções?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica

5.2. É satisfatória a quantidade de recursos usados pelo protótipo e a duração de seu uso na execução de suas funções?

☐Excelente ☐Bom ☐Regular ☐Insuficiente ☐Não se Aplica