DAS Departamento de Automação e Sistemas
CTC Centro Tecnológico
UFSC Universidade Federal de Santa Catarina

Aplicação web escalável e customizável para sistema educacional

Relatório submetido à Universidade Federal de Santa Catarina

como requisito para a aprovação na disciplina

DAS5511: Projeto de Fim de Curso

Leonardo Flores Zambaldi

Florianópolis, Agosto de 2015

Aplicação web escalável e customizável para sistema educacional

Leonardo Flores Zambaldi

Esta monografia foi julgada no contexto da disciplina
DAS5511: Projeto de Fim de Curso
e aprovada na sua forma final pelo
Curso de Engenharia de Controle e Automação

Marcelo Ricardo Stemmer

Assinatura do Orientador

Banca Examinadora:

Vinicius Flores Zambaldi
Orientador na Empresa

Prof. Marcelo Ricardo Stemmer
Orientador no Curso

Prof.
Avaliador

aluno1
aluno2
Debatedores

Resumo

Aplicações web têm se tornado cada vez mais importantes em nossas vidas. A conveniência do acesso à qualquer momento, as simples interfaces e o acesso massivo à internet tornam estas aplicações elementos estratégicos para todos os setores da economia. Há diversos desafios de conhecimento, desenvolvimento, experiência e constante atualização, sendo assim, é hoje uma habilidade fortemente requisitada mudialmente. A educação, por exemplo, está passando por uma revolução nunca antes vista na história: iniciativas para aproximar alunos e seus pais, plataforma adaptativas e sistemas que facilitam o contato professor-aluno são inovações deste meio. O trabalho presente se dá neste exato contexto: melhorar o ensino de uma escola de programação por meio de uma aplicação customizável com tarefas, tutoriais, sistema de avaliação, acompanhamento de desempenho, obtenção de feedback individualizado e rápido, comparação de resultados, tudo feito de forma intuitiva, escalável e segura. Foram então realizados estudos de necessidades da empresa, requisitos do sistema e de tecnologias nos melhores sistemas pelo mundo. Finalmente, uma solução foi construída e será detalhada nos capítulos posteriores.

Abstract

Web applications have become increasingly more important in our lives. The 24/7 access convenience, simple interfaces and massive reach to the internet have been transforming these applications into strategic elements for all economic activities. There are several challenges regarding those systems: development, experience and frequent updates, therefore it has become a very much needed knowledge all over the world. The education sector, for example, has been drastically changed. New ways to increase the engagement between students and their parents, adaptative platforms and systems allowing to decrease the student-teacher communication gap are being developed. This thesis goes on the same direction as these initiatives: enhance a programming school education by developing a customizable web application which provides homework, tutorials and graded exercises following students results, comparing their development, obtaining precise and fast feedback. The platform has to be constructed in a scalable, intuitive and secure way. Hence, several studies were held: company necessities, system requirements and technologies compatibilities. Finally, the system has been built and the information regarding it will be detailed within the next chapters.

Lista de Figuras

Sumário

Capítulo 1

Introdução

A utilização de plataformas computacionais no contexto educacional têm se tornado gradativamente mais frequente nas instiuições de ensino. Por meio delas, muitos processos são melhorados obtendo economia, fácil acesso e rapidez das atividades da educação. De fato, as instituições de ensino mais importantes no Brasil já utilizam bastante esse instrumento. A UFSC, por exemplo, utliza o Moodle há algum tempo no cursos presenciais e é também integrada com outros sistemas de controle administrativos.

Em meados de 2014 a escola de programação para crianças e jovens, I Do Code, foi criada em Florianópolis. Pelo fato dos sistemas apresentarem diversos benefícios, consideramos o uso dessas plataformas com atividades concentradas no ensino de programação. Nas aulas iniciais já ficou evidente a necessidade de um ambiente para acompanhamento das atividades dos alunos, exibição de conteúdo personalizado, organização das tarefas, fornecimento de tutoriais, repositório de textos e vídeos, exibição de gráficos e avaliação on-line. Na medida em que os alunos foram evoluindo e desenvolvendo suas habilidades, assim como o curso foi se desenvolvendo com atividades gradativamente mais complexas, tornou-se necessária então a criação desta plataforma.

Este projeto de conclusão de curso consiste na pesquisa, planejamento, elaboração e descrição da aplicação web para o curso de programação para a escola I Do Code. Nos capítulos seguintes mostramos a descrição e histórico da I Do Code, fundamentação teórica do projeto em que todos os elementos do planejamento são enfatizados, como sistemas web, tecnologias e linguagens de programação. Em seguida, a metodologia empregada é descrita e detalhada como os elementos de engenharia de software e funcionalidades. Logo após, apresentamos os resultados obtidos. O trabalho termina com um capítulo conclusivo analisando resultados, dificuldades encontradas e futuras abordagens relacionadas ao projeto.

Capítulo 2

A Empresa

A I Do Code é uma escola que tem como missão empoderar as novas gerações de crianças e jovens. Acreditamos que é necessário fazer com que eles entendam melhor o mundo tecnológico em que vivemos. Portanto como cursos principais eles aprendem diversas finalidades da programação, como: criar jogos, aplicativos, websites e robôs.

A seção a seguir mostra porque criamos a empresa e acreditamos que com essa iniciativa podemos ajudar milhares de crianças e jovens a crescer como pessoas mais conscientes e criadores de tecnologia.

2.1 Motivação

A idéia que originou a empresa surgiu a partir do encontro de várias constatações.

A primeira é a observação que as novas gerações crescem conectadas diariamente ao meio digital. Este meio transformou-se no local da zona de conforto dos jovens, onde eles passam a maior parte do seu tempo.

A segunda observação é em relação as atividades dedicadas ao mundo digital: o modo em que estes jovens utilizam os mecanismos atualmente é improdutivo: basicamente redes sociais e jogos. Com certeza há utilidade nesses, todavia o tempo demasiado é um disperdício de potencial dos jovens.

A terceira é que atividades cognitivas oferecidas aos jovens são extremamente limitadas. São poucas ou muitas vezes não existem tarefas que realmente os fazem pensar construtivamente, desenvolvendo seu raciocínio.

A quarta observação é que seria interessante que durante a adolescência, principalmente, os jovens conseguissem fazer algo que impactaria clara e diretamente seu futuro, o que novamente, dificilmente ocorre.

Finalmente, a última observação é o rumo que a educação mundial está caminhando. Países desenvolvidos como Coréia do Sul, Finlândia e Estados Unidos perceberam estes pontos e também a escassez, que tende a aumentar no futuro, da demanda por pessoas que entendem a tecnologia e mais especificadamente a programação.

Por que então não juntar o mundo digital dos jovens com atividades produtivas de concentração, raciocínio lógico e criatividade? Em meados de 2014 foi criada a I Do Code, ela visa ajudar na solução de todos estes problemas.

A seguir detalhamos o funcionamento e particularidades da escola.

2.2 A escola

Atualmente a I Do Code tem 3 unidades em Florianópolis e São José nos seguintes bairros: Itacorubi, Centro e Campinas. Percebemos que era essencial para o crescimento da empresa que cobrissemos grande parte do território da região, pelo fato de grande parte dos alunos serem crianças e portanto não terem locomoção independente.

O momento atual é de expansão em cada uma destas regiões. Os alunos via de regra gostam muito de ir para as aulas e aprender o conteúdo, portanto acreditamos que para o sucesso precisamos fortemente fazer campanhas futuras de conscientização. Grande parte dos alunos tem os pais já no setor de tecnologia, por isso é simples entender como o que fazemos será muito impactante no futuro destes jovens.

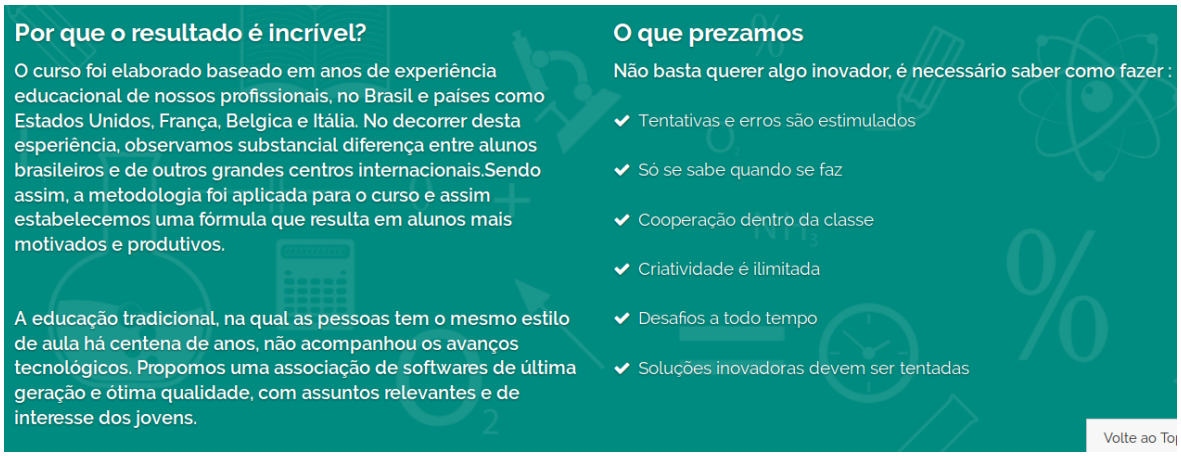

Figura 2.1. Filosofia I Do Code

Adotamos uma didática substancialmente diferente do ensino tradicional. Já é sabido que o modelo atual de escola não é o ideal principalmente pela passividade do aluno neste ambiente, adicionalmente a programação exige que o aluno continue estudando em casa e aprenda muitas coisas individualmente. Por isso adotamos pro-

cessos de ensino que vem se provando melhores e mais motivantes para o aprendizado como, por exemplo, a educação por projeto (veja a figura 2.1 na página 4 para mais detalhes).

Em nosso curso principal, por exemplo, ensinamos crianças a partir de 8 anos a criar jogos, aplicativos e websites, aprender robótica e finalizamos o curso com uma experiência de empreendedorismo (veja a figura 2.2 na página 5).

Figura 2.2. Website I Do Code - www.idocode.com.br

Existem ainda palestras e cursos de férias, os quais são modos rápidos e simples de jovens terem o primeiro contato com a programação e entender a importância dela.

2.3 Futuro

Cada dia desde a fundação acreditamos ainda mais que estamos no caminho correto para ajudar as pessoas. É incrível poder ver crianças com real vontade de aprender, criar e melhorar em diversos aspectos de sua vida. As idéias para o futuro estão no sentido de poder ajudar cada vez mais os jovens.

Buscaremos fazer palestras, oficinas e novos cursos para que cada vez mais este caminho da tecnologia e ensino inovador seja o escolhido pelas pessoas. Para isso buscaremos também parcerias como universidades e empresas, por exemplo. A iniciativa da escola tenta suprir uma grande deficiência do país que é mão-de-obra no setor de tecnologia, assim com os aprendizados, vários estarão aptos a fazer um estágio quando terminarem o curso. A idéia é que não só os alunos tenham maior interesse pela área de tecnologia, como também transformem-se nos futuros trabalhadores desta área ou até os futuros empreendedores.

Acreditamos que essa iniciativa fará com que o jovem cresça mais questionador, independente, criativo, melhore sua concentração, seu entendimento de tecnologias, escolha melhor sua profissão e seja mais preparado para o futuro num mundo que muda cada vez mais rápido, portanto um grande futuro pela frente.

Capítulo 3

Fundamentação Teórica

Neste capítulo apresentaremos a teoria relacionada à aplicações web. Há uma gama de pontos que devem ser abordados quando é considerado este tipo de aplicação, pois ele nada é que um pedaço de software on-line. Portanto, grande parte dos desafios relacionados a outros tipos de software, também estão aqui presentes. Para melhor entendimento do contexto, serão tratados assuntos relacionados à softwares na internet e também várias das particularidades e diferenças em relação aos softwares tradicionais (nativos).

3.1 Aplicações Web

As aplicações web são softwares projetados para funcionar através dos browsers ou navegadores. Elas utilizam a internet para a transferência de dados, normalmente comunicam-se com um servidor da aplicação e as têm suas próprias tecnologias, como as linguagens HTML e CSS. Confome [7], a grande ascensão e popularização das aplicações web deu-se principalmente a partir do ano 1995, foram vários os pontos que possibilitaram este crescimento, destacamos:

- Inovações em infra-estrutura dos provedores de internet. Houve uma significativa redução de custos para os fornecedores e consequentemente para os clientes. Foi um fator marcante pelo fato da aquisição de novos clientes ocorrer em grande escala.
- Outro efeito da melhora na infra-estrutura foi o aumento da velocidade para os usuários. Isto permitiu que melhores aplicativos fossem desenvolvidos: mais funcionalidades, melhores interfaces e sistemas mais complexos. Um dos grandes problemas que desenvolvedores enfrentam é construir programas baseados na

configuração que os computadores e internet de seus clientes têm. No setor de jogos, por exemplo, há tecnologias inovadoras e incríveis que não são usadas pelo fato de exigirem computadores de alto desempenho, não sendo acessível a muitos possíveis usuários.

- Novas ferramentas e popularização das tecnologias de desenvolvimento. O número de trabalhadores neste campo cresceu numa velocidade muita alta e portanto o ecossistema estava pronto para os dias de hoje: há empresas inteiras que funcionam como uma aplicaçõe web, como é o caso do Facebook com bilhões de usários.

Outros fatores não exclusivos de aplicações web também foram fundamentais:

- Aumento da quantidade de software disponível. Com o tempo, este conjunto de softwares tornou os computadores mais atrativos para inúmeras finalidades. Foi recentemente que ocorreu a educação e adoção em massa de computadores. O computador passou a ser entendido como o meio em que as pessoas poderiam ter acesso a ferramentas que facilitariam suas vidas.

- Computadores pessoais mais potentes e menores custos. Como já dito na seção da web, o efeito foi similar. O custo menor perimitiu a adoção em massa e pelo fato dos computadores terem um desempenho maior, mais aplicações foram desenvolvidas e uma nova geração de funcionalidades foi incorporada as possibilidades de uso do computador.

A seguir trataremos das diferenciações entre softwares web e tradicionais, mostraremos diversas razões que mostram o porque foi escolhido este ambiente para o trabalho e também a razão de atualmente ocorrer o desenvolvimento massivo dos mesmos.

3.1.1 Diferenças: Software Web x Software tradicional (Nativo)

Como já dito, construir softwares web(SW) é algo muito mais recente que os softwares tradicionais(ST), ou seja, aqueles que são instalados diretamente em sistemas operacionais. Apesar de ambos os tipos de software apresentarem várias semelhanças pelo fato de que ao final de tudo serem puramente códigos, conforme [8], eles têm diversas diferenças na utilização, abrangência, desempenho e mais. Listamos algumas delas a seguir.

1. Segurança. Os ST têm um grande problema com segurança, pois dependem do estado do sistema operacional(SO) do usuário. Portanto se o SO apresentar problemas ou arquivos maliciosos devido à qualquer outra aplicação, a segurança do mesmo é diretamente afetada. Por outro lado, os ST geralmente são usados offline, então não estão à mercê de diversos tipos de ataque que os SW podem sofrer (ver seção segurança para mais detalhes sobre ataques).

2. Instalação. Esta é uma grande vantagem dos SW, tudo que é necessário para um software web funcionar é um navegador (alguns SW requerem instalação de plugins web). O que é preciso em termos de software está instalado no computador servidor, portanto neste ponto SW é mais prático e tira qualquer problema que o usuário possa ter com a instalação em SO.

3. Pré-requisitos para uso. Vantagem para SW, pois é apenas necessário um navegador atualizado para usá-los. ST apresentam várias desvantagens neste quesito, desde o tipo de arquitetura 32 e 64 bits, até os requisitos de hardware - os ST necessitam de espaço em HD e versão correta do sistema operacional, por exemplo.

4. Distribuição/Alcance. Sem dúvidas a distribuição dos SW é melhor. Os ST precisam ser inicialmente instalados e posteriormente atualizados. Os SW por sua vez, são instalados e atualizados no servidor. O usuário muitas vezes nem percebe estas mudanças e elas podem ser mais frequentes e numerosas.

5. Desempenho. Vantagem para ST. Nos SW muitas vezes são necessárias que transmissões junto ao servidor sejam realizadas, portanto é mais lento. Adicionalmente a quantidade de dados transmitidas não pode ser tão grande, pois isto pode demorar muito e a aplicação perde a idéia de tempo real, a qual é praticamente uma obrigação das aplicações web atualmente. Os ST são muito melhores, pois eles trabalham sem necessidade de transmissão de dados fora do computador cliente e tem o hardware do computador inteiro à seu dispor, sendo por isso possíveis aplicações como jogos.

6. Acesso. Vantagens para os dois lados dependendo da aplicação. ST podem ser usados em qualquer local, sem necessitar de internet, basta apenas o computador em que a aplicação está instalada, estar disponível. Os SW por outro lado, podem ser usados de qualquer computador, sem a necessidade de instalação, porém é necessário o acesso à internet.

7. Aplicações específicas. Algumas aplicações precisam de um servidor e acesso online, por isso precisam ser SW, entre elas estão: internet banking, rede sociais, mecanismos de busca massivos. Por outro lado algumas aplicações precisam processar e transmitir enormes quantidades de dados ou apenas executar offline, como jogos, criação e edição de arquivos sigilosos, execução de softwares para prever o tempo (grandes quantidades de dados), conforme [4].

Passamos agora a um grande problema de aplicações web: a segurança. Hoje, a internet tem um espaço grande em nossas vidas, compartilhamos informações em redes sociais, fazemos compras com cartões de crédito e temos informações pessoais em diversos sistemas on-line, por isso cada vez mais a segurança é de suma importância em softwares web.

3.1.2 Segurança

Segurança é uma grande preocupação em sistemas de todo tipo. Na internet isto é intensificado, pois há maneiras de que por exemplo, conteúdo e senhas sejam interstein1998webceptados. Dentre as propriedades de sistemas web, conforme [11] há 3 principais que devem ser entendidas e respeitadas para seu bom funcionamento:

- Integridade. Esta propriedade diz respeito as mensagens que são transmitidas, fora e dentro do sistema. Para um sistema ser considerado íntegro, as mensagens que são enviadas devem ser obrigatoriamente as mesmas no momento do recebimento. Um exemplo clássico de sistema não íntegro é a alteração de mensagens enviadas numa transferência bancária. Estas mensagens podem assim ser destinadas a uma conta diferente da escolhida pelo usuário. Colocando de outra forma, em um sistema íntegro não é possível que fraudes por modificação de mensagens sejam realizadas.

- Privacidade. Esta propriedade diz respeito a quem tem acesso ao conteúdo das mensagens. Para um sistema ser considerado privado, nenhuma entidade diferente do servidor e do cliente que gerou aquela mensagem pode ter acesso ao conteúdo. Uma das técnicas que cria sistemas privados é a criptografia. Atualmente, há vários sistemas que usam criptografia até mesmo na transmissão de mensagens com dados não sensíveis, sendo que os dados sensíveis são aqueles relacionados à informações sigilosas, como senhas.

- Autenticidade. Esta propriedade diz respeito às comunicações feitas na aplicação. Para um sistema ser considerado autêntico, uma requisição que é feita ao servi-

dor precisa obrigatoriamente ocorrer. Devem haver mecanismos que para uma dada requisição, consigam identificar o recebimento, reenviando-a caso ocorra algum problema e repetir o processo. Existem diversas formas e implementações para esta parte, porém ela geralmente funciona com temporizadores. Eles comparam o tempo decorrido desde o envio da mensagem com o tempo esperado da transmissão e tem suas decisões baseadas nisto.

Dependendo da sensibilidade do tipo de dado e do sistema em questão existe a necessidade de proteção contra uma variedade enorme de ataques. Detalhamos aqui os mais comuns. Alguns deles devem ser implementados desde as primeiras versões, outros apenas com o aumento da escala da aplicação.

- Ataque Ddos. Neste tipo de ataque vários computadores escravos são controlados por um mestre, veja a figura 3.1 na página 12 para melhor entendimento. O objetivo deste mestre é inundar um servidor de requisições para que seu serviço fique indisponível quando algum usuário comum tente fazer uso do servidor, conforme [10].

  O que ocorre neste caso, é que grande parte dos servidores web não são projetados para operação em grande escala, como o serviço do Google por exemplo. Portanto quando milhares de requisições por segundo são feitas ao servidor, ele não consegue atender a esta demanda, a rede fica congestionada e o sistema torna-se extremamente lento ou até indisponível. Isto ocorre artificialmente (não sendo um ataque malicioso) no Brasil em datas como a Black Friday: os sites de ecommerce não conseguem atender a demanda de milhões de pessoas.

- Ataque Scanner de Portas. Neste tipo de ataque, as portas de acesso ao servidor são testadas para identificação de vunerabilidades (figura 3.2 na página 13). O que o scanner faz é tentar requisições por diversas portas, receber a resposta e processá-la, sendo que o dado recebido indica se a porta é passível de ataque. Este processo visa identificar o caminho mais fácil para invasão de computadores.

- Vírus e Cavalo de Tróia . São pedações de software que têm por objetivo infectar um computador, neste caso pode ser o servidor responsável pela aplicação. A diferença fundamental entre os dois tipos é que os vírus são responsáveis por causar a inutilização do sistema. O cavalo de tróia é ainda mais severo, pois seu objetivo é criar defeitos no sistema, podendo criar uma ponte de acesso diretamente ao computador infectado.

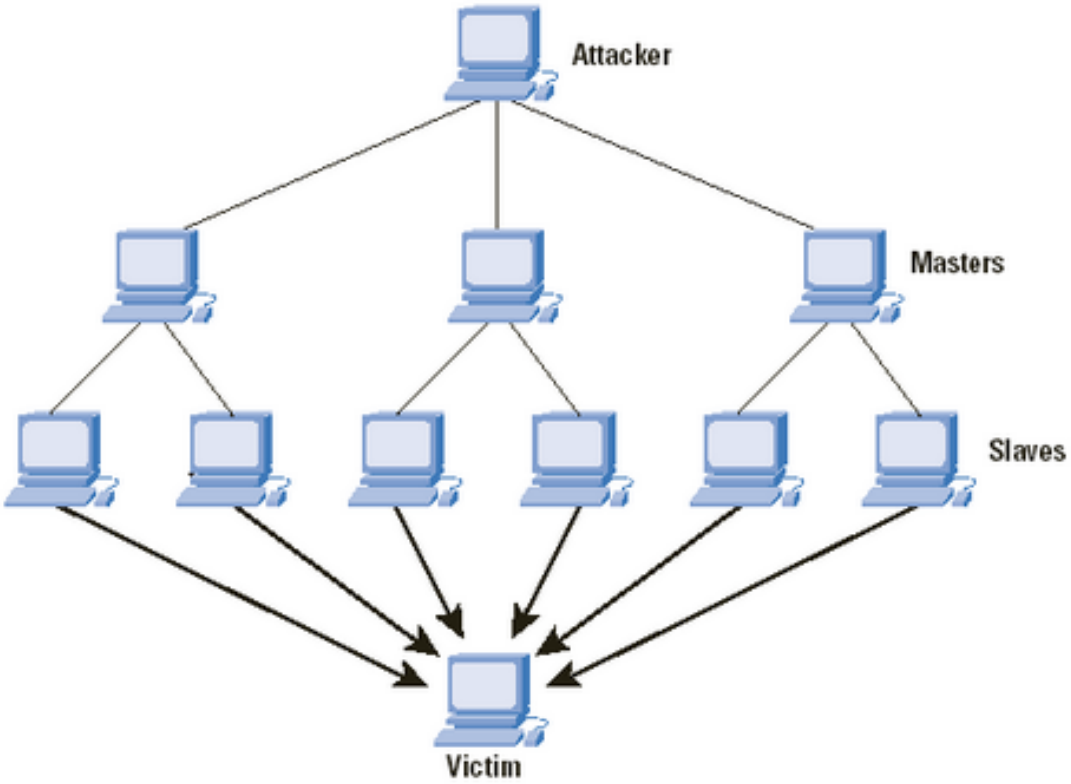

Figura 3.1. Ataque DDos

- Injeção Sql. Sql é a linguagem mais usada atualmente para manipulação de banco de dados. Este ataque se caracteriza por fazer uma requisição maliciosa ou alterar as que estão sendo feitas ao banco de dados, portanto é possível que os bancos de dados sejam completamente alterados ou até destruídos. Por causa disto é necessário ter mecanismos para tratar mensagens que são enviadas por formulários, por exemplo (conforme [3]).

- Transbordamento de Buffer. Neste tipo de ataq ue existem basicamente duas abordagens. A primeira delas é que a memória do computador seja inundada com dados adicionais irrelevantes de modo que não exista memória suficiente para outras aplicações. Assim o computador ficará extremamente lento e muitas vezes precisa ser reiniciado para liberar todos estes espaços de memória.

  Na segunda abordagem a idéia é fazer com que dados e informações do sistema armazenandos em memória no computador sejam sobreescrevidos por outros dados ou até o espaço destinado ao armazenamento de um código seja alterado, tornando os dados corrompidos e o programa impassível de ser utilizado.

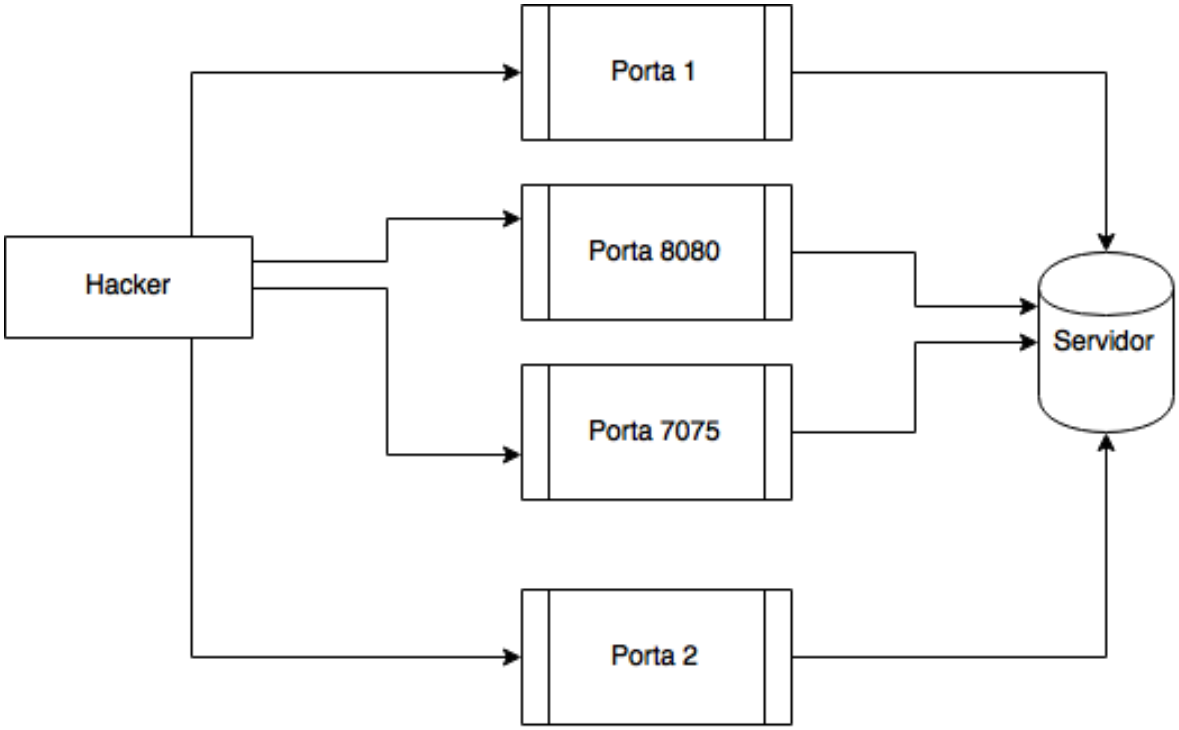

Figura 3.2. Ataque Scanner de Portas

- Cross-site scripting. Este ataque é uma exploração de falhas em websites. Um usuário pode executar qualquer comando em Javascript para tentar obter alguma informação sigilosa do sistema. O JavaScript é executado no computador do usuário, sendo por isso, passível de alterações (veja a figura 3.3 na página 14). Um modo de combater o ataque é não permitir que dados estejam disponíveis por ação dos scripts.

  Existem diversos outros tipos de ataques, listamos anteriormente alguns dos principais que profissionais de segurança precisar lidar diariamente para fazer com que o sistema fique invunerável. Outros ataques que merecem destaque: Clickjacking, Cross-site Request Forgery, Directory Traversal, Drive-by download e envenenamento de DNS.

Passamos agora as definições da arquitetura. A próxima seção afeta diretamente o desempenho, custos, complexidade e também a segurança, normalmente quanto maior o número de componentes envolvidos, maior o número de possíveis vunerabilidades do sistema, conforme [6].

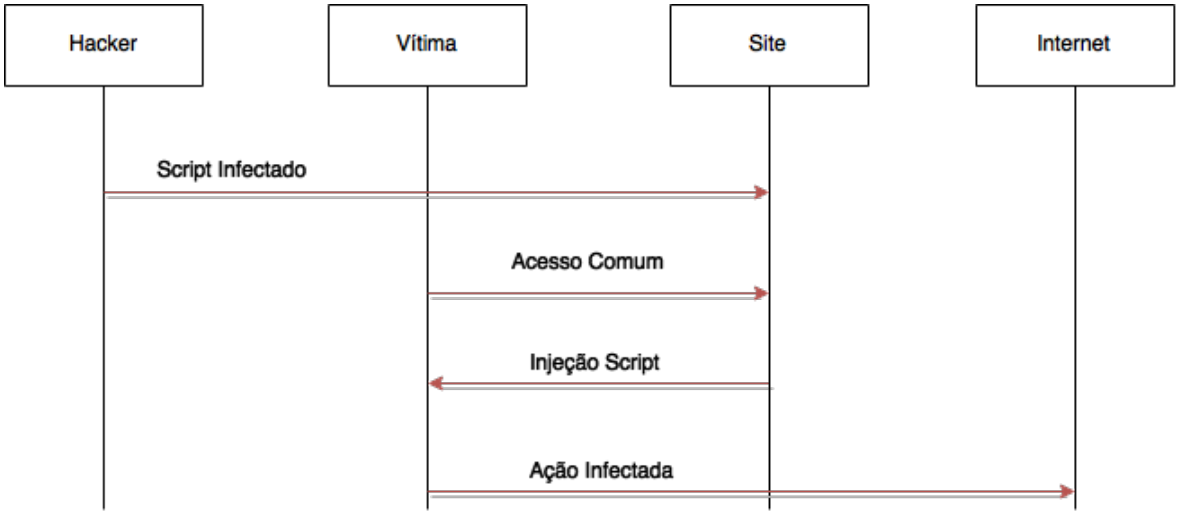

Figura 3.3. Ataque Cross-site Scripting

3.1.3 Arquitetura

Neste trabalho adotamos a arquitetura básica de aplicações para web, o cliente-servidor. Basicamente um usuário do sistema é considerado um cliente, podendo portanto existirem diversos clientes, o servidor é um computador remoto que pode ser acessado pela internet. Há também um banco de dados conectado ao servidor para armazenamento de dados e realização de consultas.

Nesta arquitetura é necessário um caminho de ida e volta em todos os elementos adjacentes do sistema e somente eles (veja a figura 3.4 na página 15). A seguir mostramos um fluxo de mensagem comum do sistema, a divisão é feita de acordo com a ordem de execução, cada uma com suas particularidades.

1. Primeiramente pode-se interagir com a página web. Quando o cliente requisita alguma página nova, envio de formulário, atualização, entre outros é realizada uma requisição HTTP.

2. De acordo com esta requisição várias coisas podem ocorrer. Se é por exemplo algo mais simples que não envolve um servidor, como por exemplo a troca de imagens em um slider ou visualização de testemunhos de clientes, esta requisição é tratada com linguagens que funcionam do lado do cliente, como o Javascript. A resposta é devolvida ao cliente na forma de uma página atualizada.

   Se por outro lado o cliente faz um login no sistema, é necessário validar o usuário e sua respectiva senha de acordo com as informações do banco de dados, portanto o navegador faz uma requisição ao servidor com as informações.

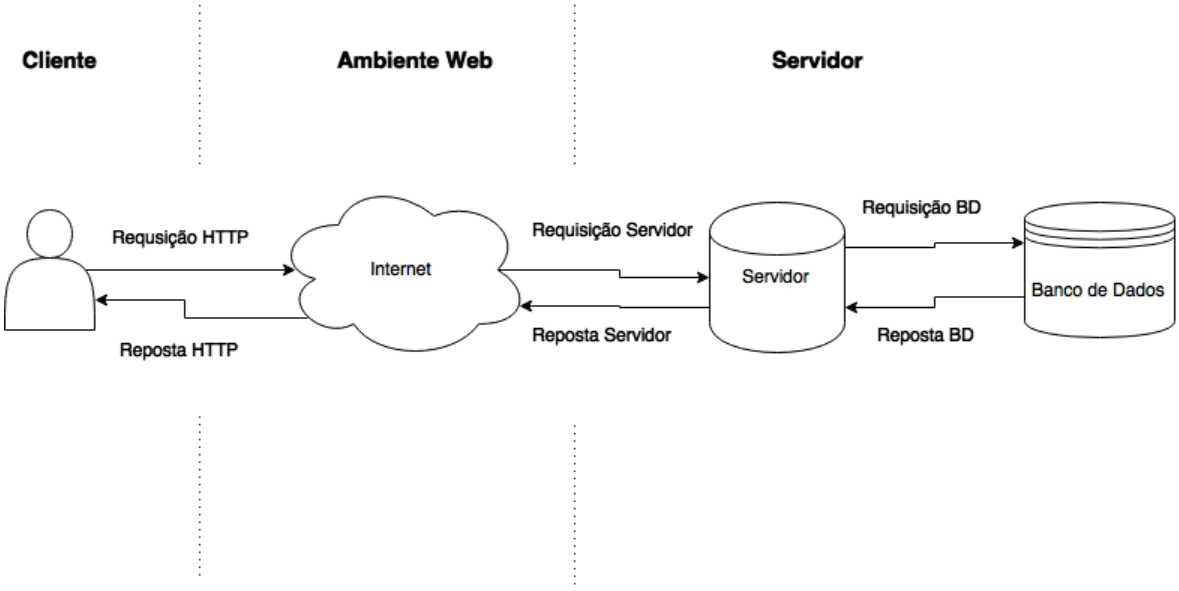

Figura 3.4. Arquitetura Cliente Servidor

3. O servidor quando recebe esta requisição entende seu tipo e faz o tratamento. Se é algo que não necessita do banco de dados, como o fornecimento de uma página estática, o servidor localiza este arquivo e envia-o como resposta.

   Por outro lado se é algo como um login no sistema, é necessário fazer uma requisição ao banco de dados para que os valores nas tabelas sejam verificados e a resposta seja dada como 'ok', se os dados estejam corretos ou 'inválido', se os dados forem diferente dos que estão na base de dados.

   Dependendo da aplicação, esta etapa de tratamento da requisição do browser, execução de vários programas e contato com diversos bancos de dados pode ocorrer inúmeras vezes.

4. O banco de dados recebe esta requisição do servidor que é na maioria dos casos uma consulta estilo SQL. Ele acessa a tabela correspondente, faz as alterações nos dados (caso seja um novo cadastro no sistema por exemplo) ou até retorna os resultados para serem usados pela programação presente no servidor (caso seja apenas uma consulta de dados).

5. Quando volta do banco de dados, a mensagem vai sendo processada pelas etapas seguintes (servidor e internet), até que chegue no cliente e tudo a página web é atualizada quando o login é autorizado, por exemplo.

Salientamos ainda que cada diferente processo é único: possui próprios caminhos,

tempo de execução, interações e resultados diferentes. Uma requisição que necessita do banco de dados deve passar por todas as conexões, por exemplo.

Após apresentada a arquitetura básica, passamos agora a aspectos de escalabilidade que possibilitarão que o sistema funcione em larga escala. É notável dizer que a escalabilidade pode fazer com que existam alterações na arquitetura do sistema, porém sua estrutura básica será a mesma apresentada neste capítulo.

3.1.4 Escalabilidade

Escalabilidade é hoje um fator extremamente importante em sistemas web. Para poder obter o ganho de escala inerentes a estes sistemas, deve-se construir um software especialmente projetado. Há diversas partes do sistema que precisam funcionar em conjunto para que, por exemplo, vários usuários possam utilizar o sistema sem problemas relacionados a concorrência ou até alta carga do servidor.

Todavia a escalabilidade não está limitada ao acesso de vários usuários, é também necessário que todas as partes, desde a infra-estrutura até as partes de software estejam em conformidade com a proposta. Em suma para um sistema ser dito escalável, ele precisa ser passível de processar um crescente número de usuários de maneira uniforme. A seguir explicitamos alguns dos termos relacionados à escalabilidade.

- Nó de sistema. São as partes únicas constituintes do sistema. Um computador pode ser considerado um nó.

- Escalar verticalmente. Ocorre em um sistema quando, não há mudança do número de nós, mas sim na capacidade de desempenho do nó. Um sistema pode escalar verticalmente de diversos modos: quando aumentamos a memória ou o espaço em disco rígido ou ainda o processador instalado em um nó.

- Escalar horizontalmente. Ocorre quando um nó é adicionado ao sistema. Escalar horizontalmente pode ser feito de diversos modos também, um exemplo simples é ter mais um computador como servidor para executar as operações caso algum esteja ocupado ou ainda criar uma réplica do banco de dados para que operações possam ser feitas em paralelo.

Há inúmeras técnicas e variações que podem ser feitas para escalar horizontalmente e aqui listamos algumas das mais importantes.

- Cache de Banco de Dados. A técnica de cache é extremamente importante para sistemas com muitas requisições. Qualquer técnica de cache tem por objetivo

principal fazer com que resultados de maior frequência de requisição, sejam separados dos demais e seu acesso seja mais rápido. A implementação do cache em banco de dados faz com que as operações de leitura tenham menor tempo de resposta e muitas vezes a carga do sistema também torna-se muito menor pelo acesso rápido e eficiente ao cache (veja a figura 3.5 na página 17).

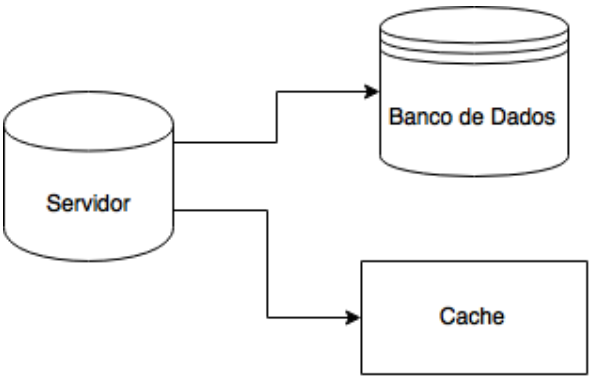

Figura 3.5. Cache

- Replicação banco de dados. A técnica de replicação é muito usada em escalabilidade de sistemas. Ela permite um grau maior de escalabilidade na leitura, como também aumenta a confiabilidade da aplicação.

  Nela os bancos de dados são replicados em clones, podendo estar localizados em servidores distintos (veja a figura 3.6 na página 18). Isso faz com que, se a carga é alta na utilização de algum banco, algum outro possa ser utilizado ou ainda se algum banco apresentar problemas, algum backup possa ser escolhido como o banco principal imediatamente.

  Dentro da categoria de replicação, ainda há diversas técnicas que são usadas dependente do sistema em questão, como as variações do sistema mestre-escravo.

- Memcached. O Memcached é um sistema que faz a aplicação de técnicas de cache em memória RAM. Ele é extremamente utilizado em aplicações que dependem fortemente ao acesso de banco de dados, pois é neste caso que sua efetividade é maior. Ele basicamente faz com que dados da aplicação, como objetos, fiquem facilmente disponíveis. Assim se estes objetos são requisitados não é preciso fazer outros acessos adicionais a banco de dados, por exemplo (veja a figura 3.7 na página 18).

- Balanceamento de carga. O balanceamento pode ser utilizado em diversas partes de sistemas web. Ele é um sistema que visa balancear a demanda pelos recursos

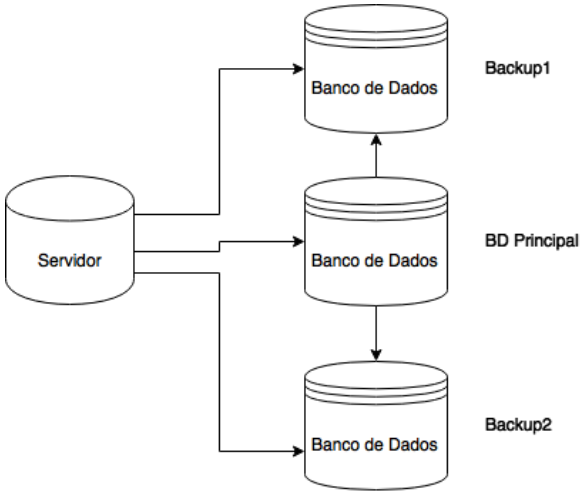

Figura 3.6. Replicação do Banco de Dados

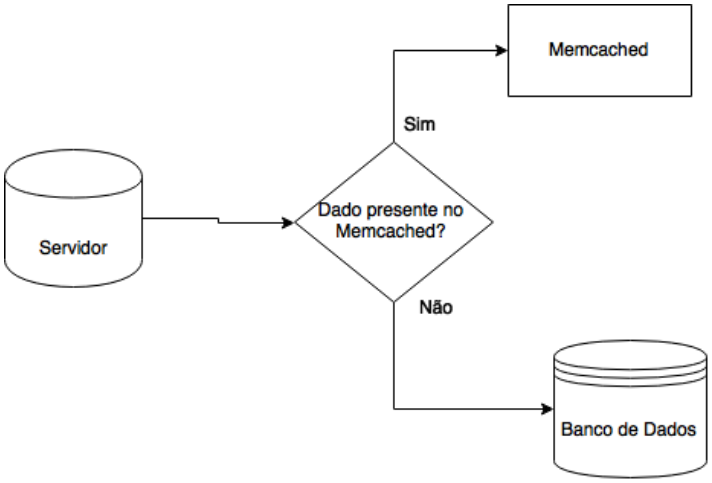

Figura 3.7. Memcached

entre os elementos que processam estas requisições. Isto faz com que nenhum elemento seja sobrecarregado, maximizando o desempenho e minimizando tempos de reposta. O balanceamento é normalmente feito com servidores, que são as unidades que processam as requisições vindas da internet (veja a figura 3.8 na página 19).

Seguimos então para as tecnologias usadas na aplicação. É importante salientar que várias tecnologias foram rejeitadas para este trabalho. As que estão presentes seguem todos os critérios definidos nos itens anteriores, ou seja, permitem a escalabilidade, tem as integrações bem especificadas e afins.

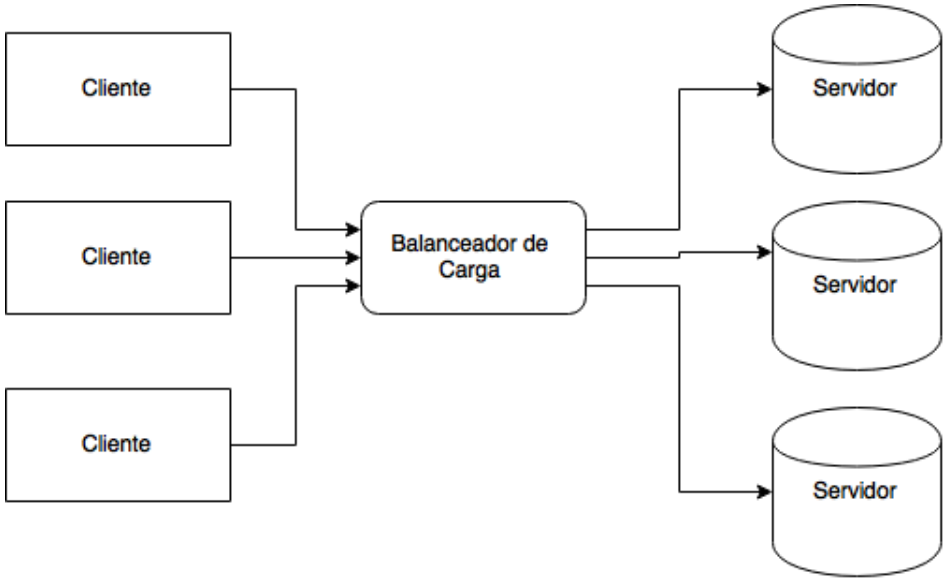

Figura 3.8. Balanceamento de Carga

3.2 Tecnologias

Em sistemas web, as tecnologias escolhidas para desenvolvimento são de suma importância. É necessário que as diversas partes do sistema sejam selecionadas para que o desenvolvimento seja mais rápido e simples. Adicionalmente elas precisam ser passíveis de integração, os programadores precisam ter domínio dos assuntos e também que exista o capital disponível para os diversos investimentos.

Um dos grandes benefícios atuais é que grande parte das tecnologias usadas no mundo são livre de custos, tem código aberto e há uma imensa comunidade de ajuda on-line. Os gastos da operação são relacionados a serviços privados, como é o caso da hospedagem de sites.

3.2.1 Linguagens

Qualquer software deve ser programado em alguma linguagem de programação e normalmente são necessárias várias. Cada linguagem tem sua especialidade e algumas delas são obrigatórias em determinados contextos ou se transformaram em um padrão tão dominante que via de regra são tratadas como única opção. É o caso de várias das linguagens deste trabalho: o HTML é usado para estruturação de páginas web, já o CSS é usado para a estilização das páginas e assim por diante. A seguir cada linguagem usada será explicitada.

3.2.1.1 HTML

A linguagem HTML é um padrão para a internet. É com HTML que os navegadores entendem os códigos e geram as páginas web. Qualquer website hoje existente usa HTML como estrutura básica. Diferente de muitas linguagens tradicionais, ele tem como linguagem base as tags (etiquetas). Um elemento HTML é formado por uma tag e os atributos que esta tag aceita. As linguagens mais tradicionais usam declarações de comandos e funções, como é por exemplo o padrão de linguagens como Python,Java ou C++. O HTML é usado a todo momento em aplicações web.

3.2.1.2 CSS

CSS é outra lingugem muito usada na internet, sendo como o HTML, um padrão da W3C (a principal organização de padronização da internet). Enquanto o HTML dá a estrutura e o conteúdo das páginas, ela é empregada conjuntamente ao HTML para definir a estilização, aparência e layout das páginas. Além de ser muito útil por fazer essa separação de conteúdo e formato, ela é interessante pelo fato de qualquer página estruturada por HTML poder usar o mesmo arquivo de definições CSS, sendo assim, é fácil fazer com que milhares de páginas de um site siga as mesmas regras de estilo.

A sintaxe em CSS especifica os diferentes estilos de propriedade de uma página web. O documento CSS pode ser entendido mais facilmente como um conjunto de regras que fala como o site deve se comportar. As regras são compostas de seletores e declarações. Os seletores são os elementos, os quais as regras de declaração serão aplicadas.

3.2.1.3 Python

Python é uma das linguagens que mais vem crescendo no mundo, por sua simplicidade e o uso para educação em programação. Ela é considerada uma evolução em relação as linguagens mais tradicionais por ser mais intuitiva, as declarações em Python se parecem muito mais com a linguagem que usamos normalmente para falar. A linguagem pode ser usadas para inúmeras finalidades, principalmente para aplicações web e como script para administração de sistemas ou até em pesquisas científicas e computação gráfica, que exigem alto esforço computacional.

Ela é orientada a objetos e também suporta o paradigma de linguagem funcional. Várias empresas no mundo usam Python em seus sistemas, como: Youtube, Dropbox, IBM e até jogos como Civilization IV.

3.2.1.4 JavaScript

JavaScript é a linguagem padrão na web para execução de scripts do lado do cliente. Os scripts então funcionam sem ter de interagir com o servidor a todo momento. Sendo assim muito importante, pois alivia a carga de servidores e torna a execução de muitos processos mais rápidos, pois acaba com o tempo da transmissão cliente-servidor. Para execução de código javascript no lado do servidor, vêm se usando o Node.js para aplicações de alta escalabilidade. É uma linguagem orientada a objetos com suporte para programação funcional.

Passamos agora aos sistemas utilizados. Juntamente com as linguagens, eles fazem com que o software possa ser construído de um modo simples, fácil e rápido.

3.2.2 Sistemas

Nesta seção mostramos os sistemas utilizados no trabalho, eles vão desde os frameworks para aplicações específicas, até o hardware que foi necessário. É importante notar que estes sistemas são vastamente difundidos pelo mundo, tem extensa documentação e integração com diversos outros sistemas.

3.2.2.1 Django

Django é um framework para aplicações web. Ele é escrito na linguagem Python, prima por sua capacidade de segurança, rapidez e escalabilidade. É um framework muito difundido no mundo atualmente, pela facilidade do desenvolvimento, sendo possível criar formulários, trabalhar com url amigáveis e vários outros tipos de suporte web simples e muito úteis.

3.2.2.2 AngularJS

Da mesma forma que o Django, o AngularJS é também um framework. Ele é escrito em JavaScript e pode ser usado diretamente nos documento HTML. Ele torna mais simples o dinamismo em páginas web. Basicamente ele faz a atualização do modelo sempre que a visão muda e vice versa, conforme [5]

3.2.2.3 MySql

Mysql é um SGBD (sistema de gerenciamento de banco de dados) que usa a linguagem SQL para fazer consultas, ele é utilizado em todo tipo de aplicação. Atualmente é considerado o banco de dados open source mais popular no mundo, tem integração simples com Django. Diversas empresas usam Mysql: Google, Twitter, Ebay, Paypal.

3.2.2.4 AWS

A Amazon oferece a AWS (Amazon Web Services), que possui diversos serviços disponíveis pela web. Permite que desenvolvedores construam aplicações para serem utilizadas inteiramente na nuvem. É uma plataforma que é escalável e o desenvolvimento é possível em várias linguagens como: Python, Ruby, PHP, entre outras, conforme [1]. É usado por diversas grande empresas e aplicações como SoundCloud, GitHub, Dropbox, Mozilla.

Capítulo 4

Metodologia

O capítulo da fundamentação teórica diz respeito principalmente ao estudo e detalhamento das partes estruturais e tecnologias do sistema. Neste capítulo faremos o estudo dos métodos utilizados no trabalho, ou seja, mostramos o modo como várias partes da fundamentação teórica são postas em conjunto e usadas como base para a implementação.

4.1 Engenharia de Software

A engenharia de software busca por meio de mecanismos gerais, fazer com que todo o software possa ser especificado. Esse detalhamento deve ser feito em diversas instâncias de acordo com o estágio em que o software se encontra. Desde requisitos, projeto, construção, testes e gerência, todo processo é passível de ser especificado em textos, relatórios, diagramas e outros. A finalidade da engenharia, é que o software seja construido de modo mais organizado, tenha maior produtividade e qualidade.

Aqui vamos nos ater principalmente à área destinada aos requisitos, os quais são o ponto de partida do processo da engenharia.

4.2 Modelagem UML

A Linguagem de Modelagem Unificada (UML) é uma linguagem padrão na indústria de software. Resumidamente, esta linguagem visa especificar, documentar e estruturar o sistema, conforme [2]. A principal finalidade desta modelagem é fazer com que as pessoas engajadas no desenvolvimento entendam visualmente e de uma forma sistemática, quais e como são as relações entre os elementos do sistema. Pelo fato de sistemas

de todo tipo poderem ser modelados via UML, é simples também que pessoas possam entrar mais facilmente no andamento do projeto ou também que elementos sejam incluídos e modificados com bastante facilidade.

4.2.1 Casos de Uso

O levantamento dos casos de uso do sistema é uma técnica de descrição que visa tornar mais claros e exemplificados os requisitos de um novo sistema. Eles são portanto simples e de alto nível para que usuários e clientes possam ajudar em sua confecção. Os casos em si, são uma lista de cenários que mostram como o sistema deve se comportar e interagir com os usuários ou atores (veja a figura 4.1 na página 24).

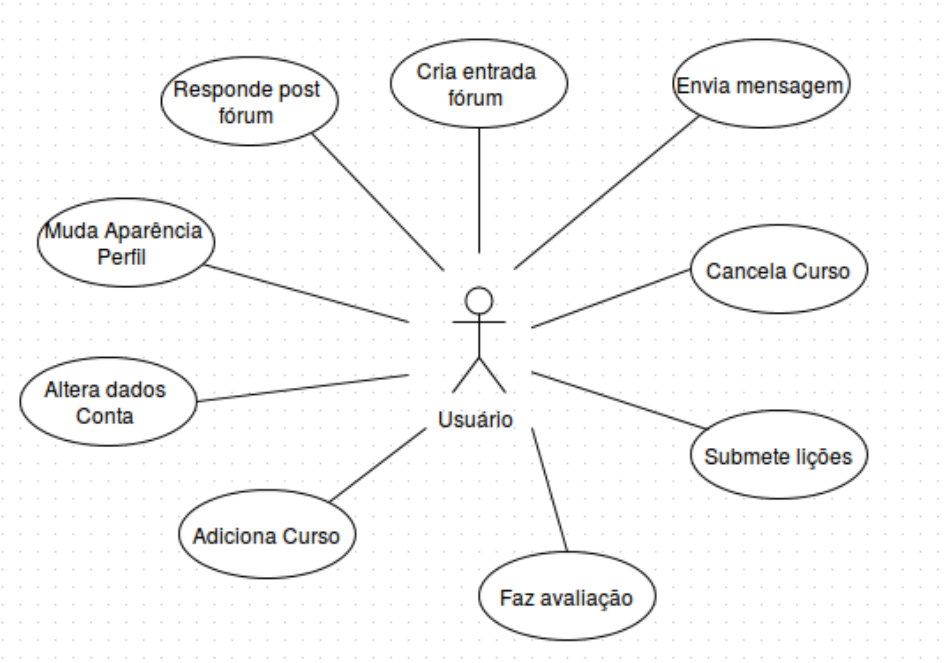

Figura 4.1. Diagrama de Casos de Uso para Usuário

4.2.2 Análise de Requisitos

A etapa de análise de requisitos visa em identificar as necessidades do software, projeto e cliente para que em seguida estas sejam especificadas. Neste processo é comum que existam conflitos e divergências, sendo que existe a necessidade de algumas iterações até que todas as partes envolvidas no processo cheguem a um consenso e o processo da engenharia de software siga à próxima etapa.

Basicamente, cada requisito deve ser uma necessidade que os sistema deverá ter.

Os requisitos podem ser divididos em funcionais e não funcionais, como veremos a seguir.

4.2.2.1 Requisitos funcionais

Requisitos Funcionais	
RF01 - Cadastro de Alunos	RF12 - Criar Exercícios
RF02 - Remoção de Alunos	RF13 - Mudar Perfil
RF03 - Mudar Aluno de Turma	RF14 - Visualizar Perfil
RF04 - Adicionar Curso para Aluno	RF15 - Alterar Dados
RF05 - Mudar descrição Curso	RF16 - Escrever no Fórum
RF06 - Adicionar Curso	RF17 - Criar Post Fórum
RF07 - Acessar Tarefa Aluno	RF18 - Fazer Login Sistema
RF08 - Avaliar Tarefa Aluno	RF19 - Sair do Sistema
RF09 - Conceder medalha aluno	RF20 - Avaliar Instrutor
RF10 - Conceder certificado aluno	RF21 - Avaliar Curso
RF11 - Criar Provas	RF22 - Visualizar notas

Figura 4.2. Requisitos Funcionais

São estes requisitos que irão especificar como o sistema deve performar. Cada requisito deve então listar um função do software ou de alguma de suas partes. Uma função é descrita como um conjunto de entradas, seu comportamento e as saídas. (veja a figura 4.2 na página 25)

4.2.2.2 Requisitos não funcionais

Diferentemente dos requisitos funcionais, os não funcionais são relacionados à especificações técnicas do software, ou seja, são as propriedades e restrições que o sistema deve seguir. Há diversas categorias que necessitam de especificação, entre elas: custo, confiabilidade, desempenho, usabilidade, segurança, manutenção, portabilidade, conforme [9].

Dependendo do sistema, alguns dos requisitos podem fazer com que o projeto fique inviável ou até existirem requistos concorrentes, daí faz-se a necessidade de entendimento profundo das restrições e a partir delas especificar as propriedades. (veja a figura 4.3 na página 26)

Requisitos Não Funcionais	
RNF01 - Sistema deve responsivo para funcionamento em smartphones e tablets	RNF07 - Sistema deve ser integrado com os frameworks Django e AngularJs
RNF02 - O sistema deverá ter alta disponibilidade: > 99%	RNF08 - O banco de dados deve ser Mysql
RNF03 - Sistema deve funcionar em qualquer plataforma com navegador	RNF09 - O Sistema deverá ser o mais leve possivel em sua interface
RNF04 - Sistema será desenvolvido nas linguagens Python, Javascript, HTML, CSS	RNF10 - Sistema só pode funcionar se usuário estiver conectado na internet
RNF05 - Sistema deverá ser capaz de integração AWS	RNF11 - O tempo de qualquer atualização de páginas não pode ultrapassar 5 segundos
RNF06 - Sistema deve ser privado, não mostrar informações não habilitadas pelo usuário em seu perfil	RNF12 - Sistema deve ser intuitivo, como textos claros e botões coloridos

Figura 4.3. Requisitos Não Funcionais

4.2.3 Diagrama de Classe

Este diagrama é mais uma forma de modelagem do sistema. Ele visa representar o sistema em forma de classes e relacionamentos, por causa disso é uma base para vários outros diagramas. As classes e relações entre elas são o ponto de partida para que os objetos sejam formados. (veja a figura 4.4 na página 27)

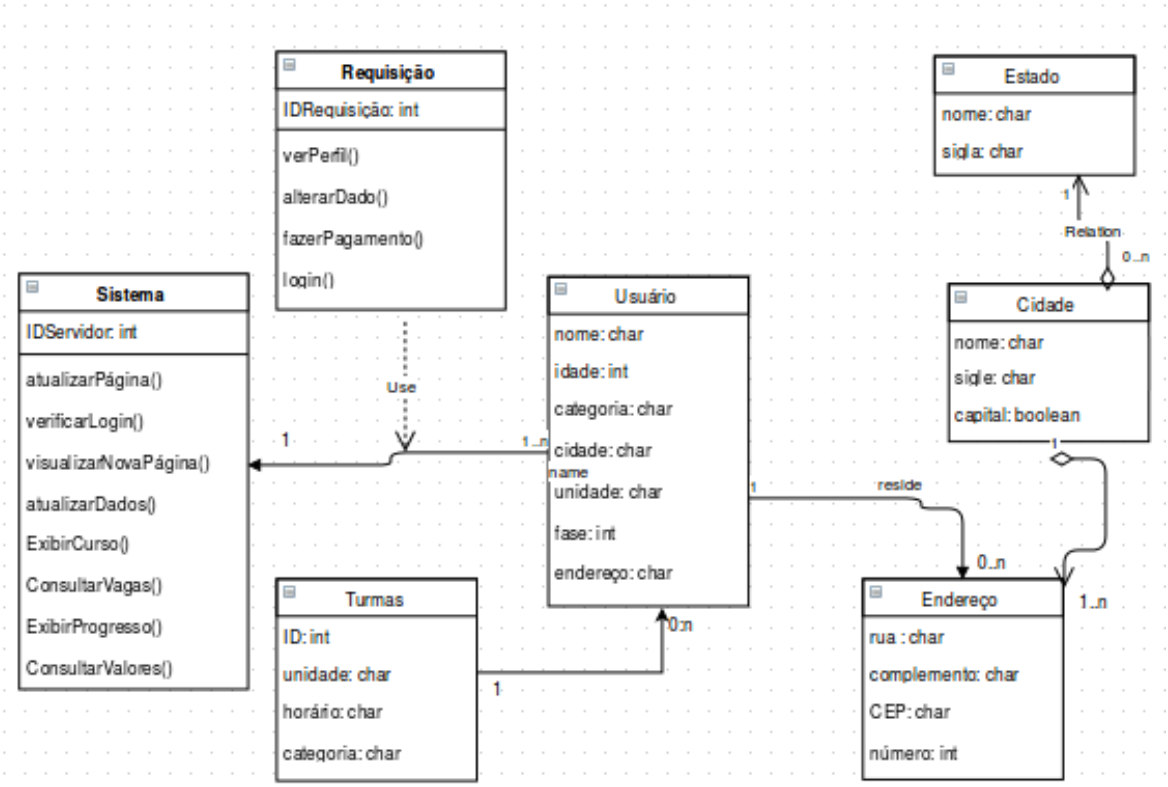

Figura 4.4. Diagrama de Classe Usuário

Capítulo 5

Implementação

Nesta seção descrevemos as principais funcionalidades implementadas. Estas foram derivadas dos requisitos do sistema levantados, conforme descrevemos na seção 4.2.2. particularmente, mostramos o sistema que é composto por um **painel**, onde ficam um resumo das atividades e cursos do aluno, uma página da **conta** do aluno, do **fórum** do curso, acompanhamento das **matérias** e por fim, questionários de **avaliação**.

5.1 Visão Geral

A parte do *front-end* foi projetada para suprir dois dos requisitos: interface ser *responsiva*, isto é, dinâmica e mutável automaticamente para diferentes tamanhos de telas, o que possibilita o uso do sistema em celulares e tablets; sistema ser *leve*, isto é, consumir pouca banda de rede durante a navegação do usuário. Esse requisito possibilita o uso do sistema em conexões com qualidade baixa, além de economizar banda móvel (3G/4G) de usuários que optam pelo uso dessas redes. Nas seções seguintes descrevemos cada parte do sistema juntamente com sua funcionalidade.

Menus

Existem dois **menus** no sistema: um lateral e um superior. Ambos ficam sempre fixos e ocupam uma área reduzida da tela, assim o miolo da página é única parte que sofre alterações. Adicionalmente o menu lateral é comprimido quando navegamos para uma página posterior à inicial. Menus fixos possibilitam uma redução na banda utilizada para navegação no site, uma vez que todas alterações de conteúdo ocorrem dinamicamente no miolo da página via *javascript* e não é necessário o recarregamento da página a cada click em uma funcionalidade. A compressão do menu lateral foi concebida

com objetivo de auxiliar a navegação no site quando usuários utilizam dispositivos com telas menores.

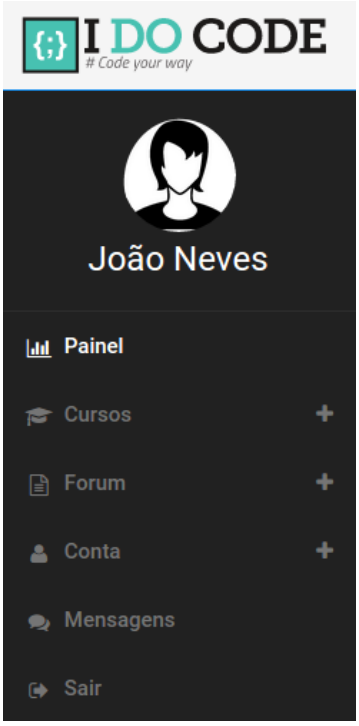

Figura 5.1. Inferface do menu fixo lateral do sistema.

A interface do menu lateral é apresentada na figura 5.1. Primeiramente existe o logo da empresa, que ao ser clicado direciona o usuário para página inicial, em seguida o avatar do usuário com seu nome, e por fim, os links do menu, consistindo no **painel**, que mostra um sumário para o aluno dos eventos recentes; **cursos**, que apresentam as matérias que já foram e estão sendo feitas; **fórum**, onde ficam as discussões e dúvidas dos alunos; **conta**, onde o aluno pode editar suas informações e avatar; **mensagens**, para comunicação privada entre alunos e professores; **sair**, para finalizar a seção corrente.

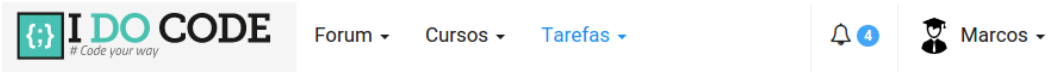

Figura 5.2. Inferface do menu fixo superior do sistema.

A interface do menu superior é apresentada na figura 5.2. Este menu apresenta três links com atalhos para o fórum e cursos, que foram descritos no parágrafo acima e

para **tarefas**, onde o aluno pode acompanhar e realizar suas tarefas. Adicionalmente a barra apresenta um ícone de notificações, onde o aluno é sinalizado de novas tarefas ou respostas para suas perguntas no fórum. Por fim, existe um atalho para a página pessoal do aluno no canto direto.

Painel principal

O painel principal funciona como um *dashboard*, onde as principais informações do curso são resumidas. Em particular, o painel é composto de *widgets* que mostram um sumário sobre os cursos que o aluno está fazendo, o resultado das tarefas, as próximas pendências (*deadlines*), últimas mensagens do fórum, os prêmios (medalhas) do aluno e, por fim, os certificados obtidos durante o curso. A figura 5.3 apresenta a interface completa do **painel**, juntamente com os menus lateral e superior já descritos. Em seguida, apresentamos cada funcionalidade (widget) do painel em detalhes.

Figura 5.3. Interface completa do painel do sistema.

Widget cursos recentes

A figura 5.4 apresenta a interface do widget **cursos recentes** do **painel**. Este widget mostra ao aluno os últimos três cursos/matérias que este está realizando, assim como o progresso do aluno em cada matéria. Também há um link para a página principal dos cursos, onde estão todos os cursos/matérias já realizados.

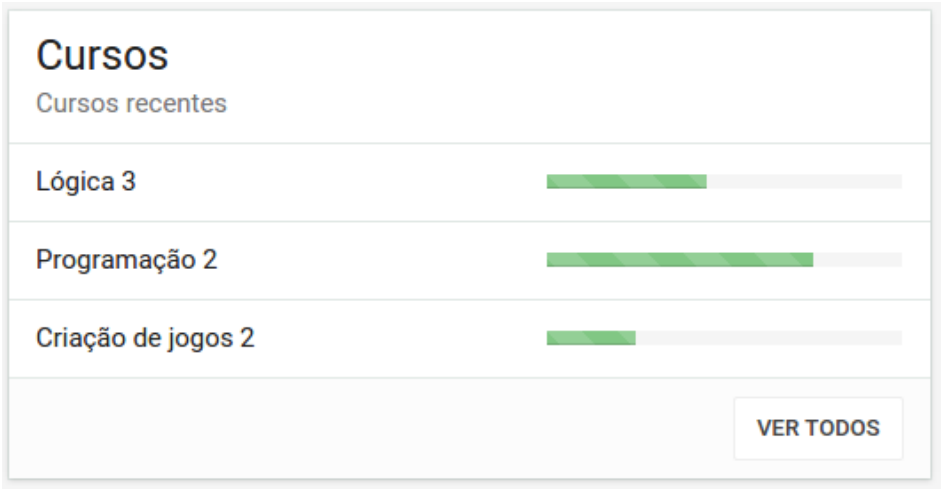

Figura 5.4. Interface do widget cursos recentes do painel.

Widget tarefas recentes

A figura 5.5 apresenta a interface do widget **tarefas recentes** do **painel**. Este widget mostra ao aluno os resultados (notas e conceitos) das últimas três tarefas finalizadas e avaliadas pelos instrutores. Há também um link para todas as avaliações do aluno, bem como links para o curso correspondente à cada tarefa. As notas naturalmente vão de zero a dez e os conceitos são divididos em *ótimo*, para notas entre 8 e 10, *bom*, para notras entre 6 e 8, *regular*, para notas entre 4 e 6, e por fim, *ruim*, para notas entre zero e quatro.

Widget Atividade no Fórum

A figura 5.6 apresenta a interface do widget **atividade no fórum** do **painel**. Este widget apresenta até cinco tópicos do forum. Os tópicos apresentados são ordenados por tempo da última atividade e não por data de criação do mesmo. Assim, caso um aluno responda uma pergunta de um tópico antigo que não aparecia no widget, este será movido para primeira posição na lista do widget de atividades no fórum. Com esse widget os alunos podem ir diretamente para os últimos tópicos ao clicar na pergunta em questão, assim como seguir para página do curso relativo à questão e até mesmo à página do perfil do aluno que fez a pergunta.

Widget Prêmios

A figura 5.7 apresenta a interface do widget **prêmios** do **painel**. Este widget apresenta as medalhas (*badges*) obtidas pelo aluno desde o momento da criação da

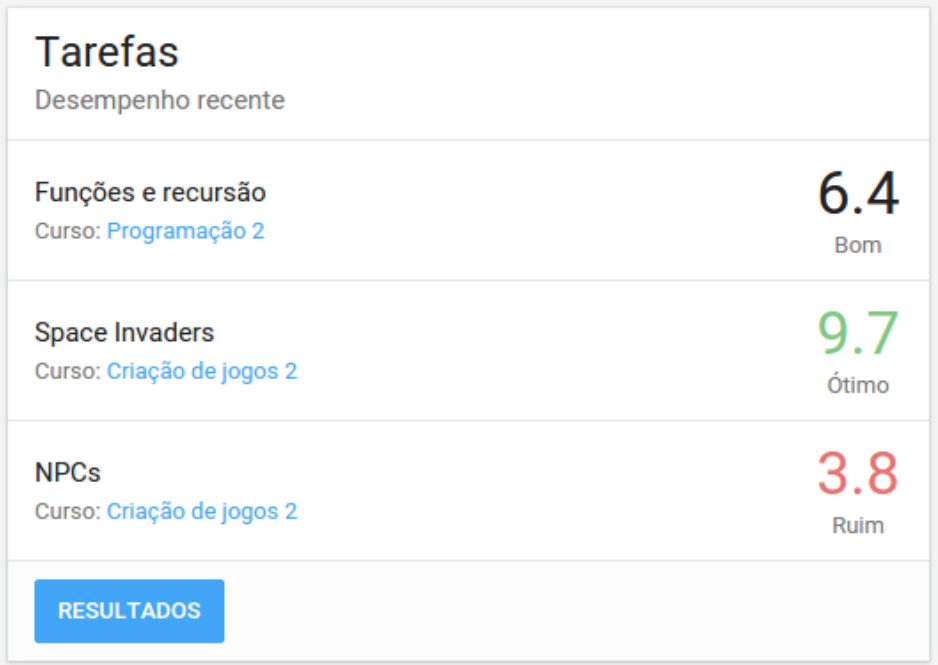

Figura 5.5. Interface do widget tarefas recentes do painel.

conta. São apresentadas até 8 medalhas do aluno e cada uma é conquistada de acordo com um desafio. Como exemplo, a medalha azul com troféu é atribuída ao aluno que vencer um concurso de jogos da escola, a medalha vermelha com um ruby é dada ao aluno que vencer uma competição de arte e a medalha roxa com estrela é atribuída ao aluno que responder 10 vezes no forum. O objetivo das medalhas e promover o chamado *gamification*, conforme apresentado na seção dos requisitos (seção 4.2.2).

Widget certificados

A figura 5.8 apresenta a interface do widget **certificados** do **painel**. Este widget traz um resumo dos últimos 10 certificados obtidos pelo aluno. Cada certificado é relativo a um curso ou grupo de matérias finalizadas pelo aluno e seguem uma graduação de cores, indo do branco até o preto assim como as faixas das artes marciais (o último curso dá a "faixa preta"ao aluno).

5.2 Outras Funcionalidades

Nesta seção descrevemos as principais funcionalidades do sistema apresentadas na seção anterior, juntamente com sua explicação e interface. Inicialmente é apresentada a página do controle do aluno, em seguida as páginas derivadas do painel principal,

Figura 5.6. Interface do widget atividade no fórum do painel.

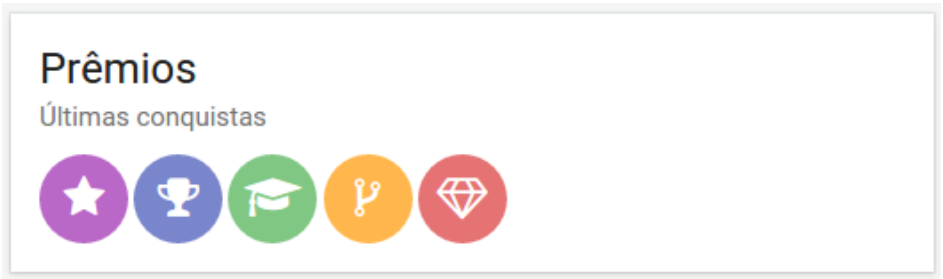

Figura 5.7. Interface do widget prêmios do painel.

começando pelo fórum, em seguida o acompanhamento dos cursos, e até a avaliação on-line.

Conta do aluno.

Uma página do sistema é destinada à conta do aluno, onde este pode fazer diversas alterações em seus dados. Em particular, o aluno pode alterar seu nome que será usado para o fórum, assim como um avatar que pode ser sua foto ou uma imagem que o represente. Adicionalmente, é possível alterar o email usado para correspondência e entrar com um website, que é mostrado quando outros alunos entram em seu perfil. Outras preferências como quais informações mostrar em seu perfil podem ser alteradas pela aba "Outros detalhes". Essas informações variam desde seu email, até medalhas e certificados obtidos. Um resumo da interface é mostrada na figura 5.9.

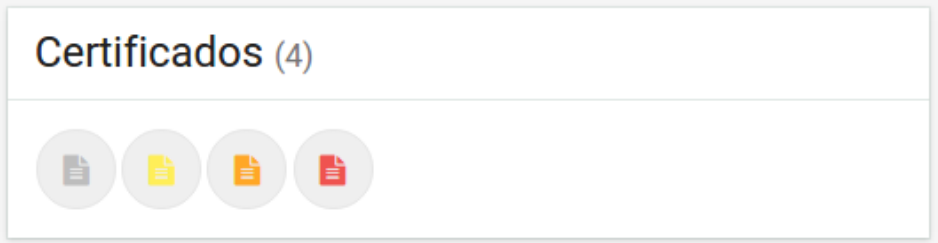

Figura 5.8. Interface do widget certificados do painel.

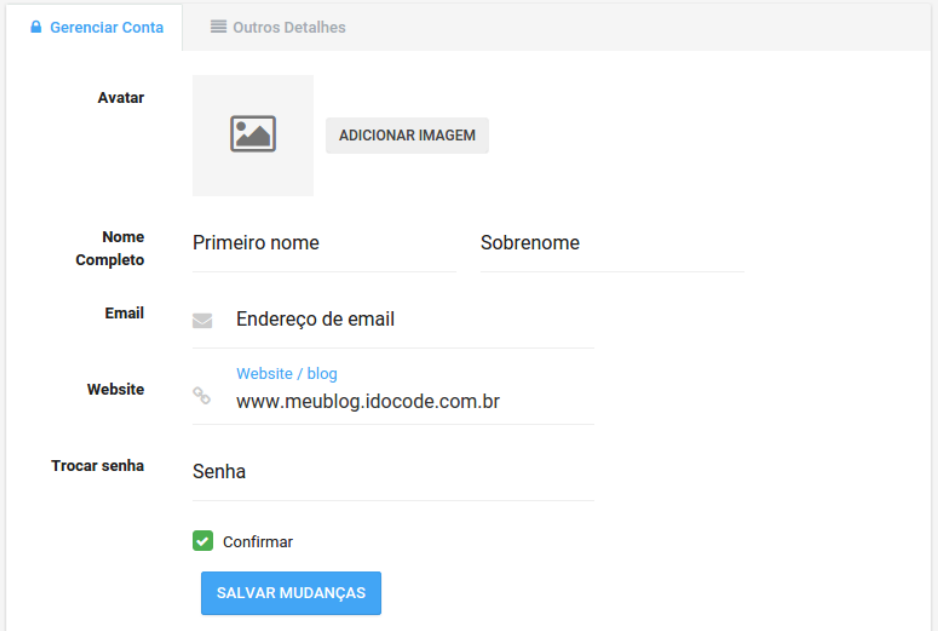

Figura 5.9. Interface da área da conta do aluno.

Fórum

O fórum é a principal ponte de comunicação do sistema, onde os alunos e instrutores podem interagir publicamente a fim de trocar experiências e/ou resolver dúvidas. Ao entrar na página do fórum, o menu lateral é comprimido a fim de maximizar a experiência dos usuários de dispositivos móveis que apresentam telas reduzidas. Existem duas páginas principais para o fórum, uma que mostra todas as questões vigentes, com um limite de cinco questões por página e outra que mostra a pergunta em si com as respostas dos usuários.

A figura 5.10 mostra a página inicial do fórum, onde para cada questão, são apresentadas informações a respeito do aluno que a criou, do tempo desde sua criação, o tílulo da pergunta e o número de respostas. Nesta página também é possível criar

uma nova questão para ser adicionada ao forum pelo botão "Novo Tópico". Por fim, é possível navegar entre todas as páginas existentes no fórum pelo recurso de paginação mostrado na parte inferior da figura 5.10.

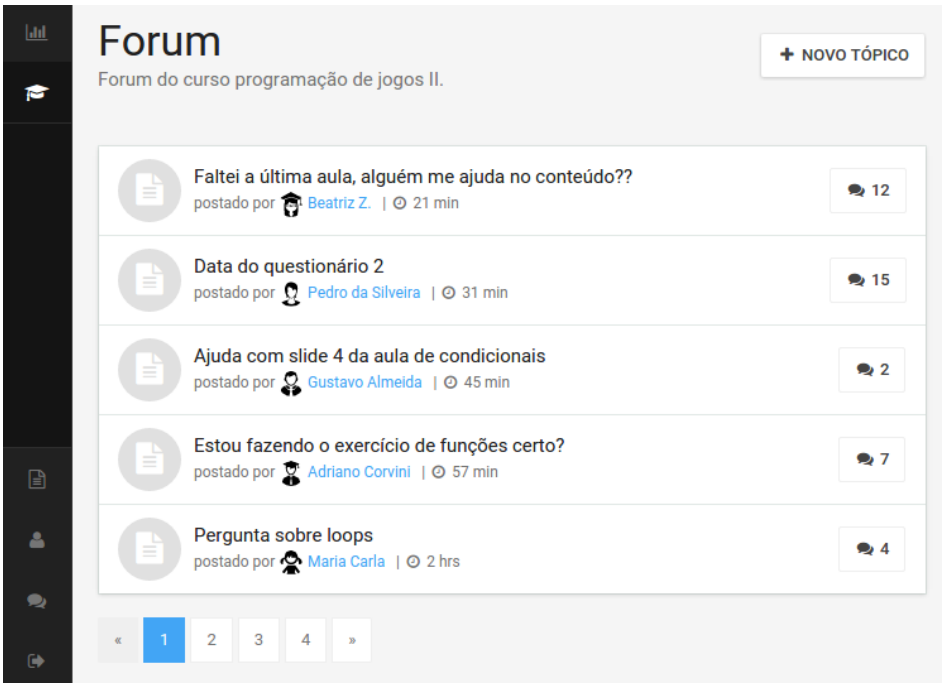

Figura 5.10. Interface da página principal do forum.

A figura 5.11 mostra uma das páginas secundárias do fórum. O usuário é direcionado a esta página ao clicar em um dos tópicos na página principal do fórum. Nesta página são apresentadas informações a respeito do aluno que postou a pergunta, o tempo desde sua criação, o tílulo da pergunta e por fim, a lista das respostas. Cada resposta é composta do avatar do usuário que a criou, juntamente com a identificação de aluno/instrutor e da resposta em si. Nesta página também é possível criar uma nova resposta à pergunta em questão pelo botão "Responder". Por fim, é possível navegar entre todas as páginas existentes no fórum pelo recurso de paginação mostrado na parte inferior da figura 5.11.

Acompanhamento dos cursos.

A página de acompanhamento dos cursos mostra todas matérias e cursos finalizados pelo aluno, bem como os cursos atuais que se encontram em andamento. Para os cursos em andamento também é mostrada uma barra de progresso para o aluno acompanhar sua evolução. A página apresenta os cursos por ordem cronológica

Figura 5.11. Interface da página específica das pergunta do forum.

e limita a seis cursos por páginas, de acordo com o requisito que visa reduzir a quantidade de dados necessários para renderização das páginas, todavia, é possível navegar para matérias antigas usando os recursos de paginação. Por fim, é possível clicar em qualquer curso na página para ir à página exclusiva do curso. A figura 5.12 apresenta a interface da visão geral do acompanhamento dos cursos.

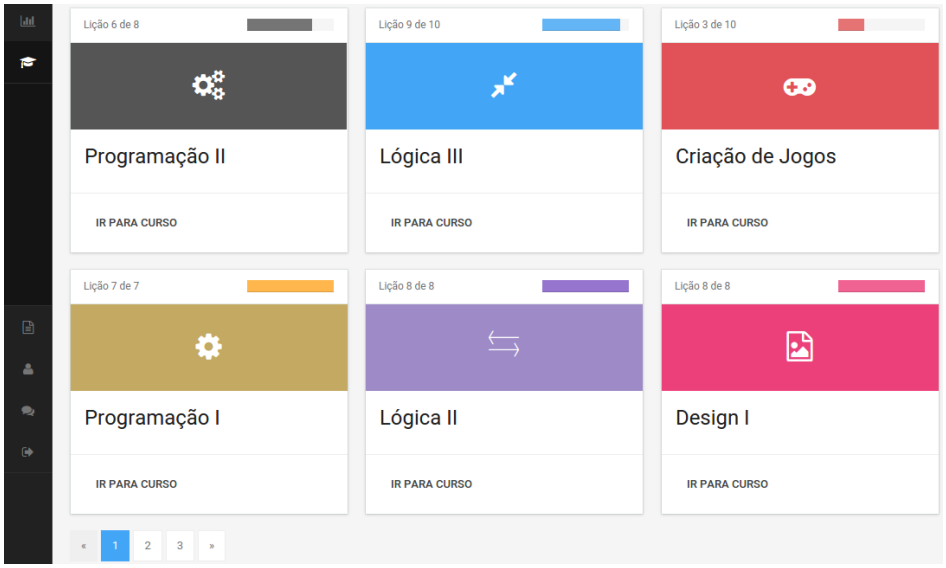

Figura 5.12. Interface da página de acompanhamento geral dos cursos.

Ao clicar em um dos cursos na página de acompanhamento, o usuário é levado à página específica do curso. Esta página apresenta o título e uma descrição geral da matéria, bem como seu currículo. O currículo de cada matéria é dividido em capítulos, e cada capítulo é dividido em tópicos específicos. Cada tópico pode ter um link para um slide ou um vídeo que explica o conteúdo brevemente e uma marcação que indica se o aluno já estudou o tópico ou não (isso é usado para medir o progresso do aluno na matéria). A figura 5.13 apresenta a interface da página específica do curso.

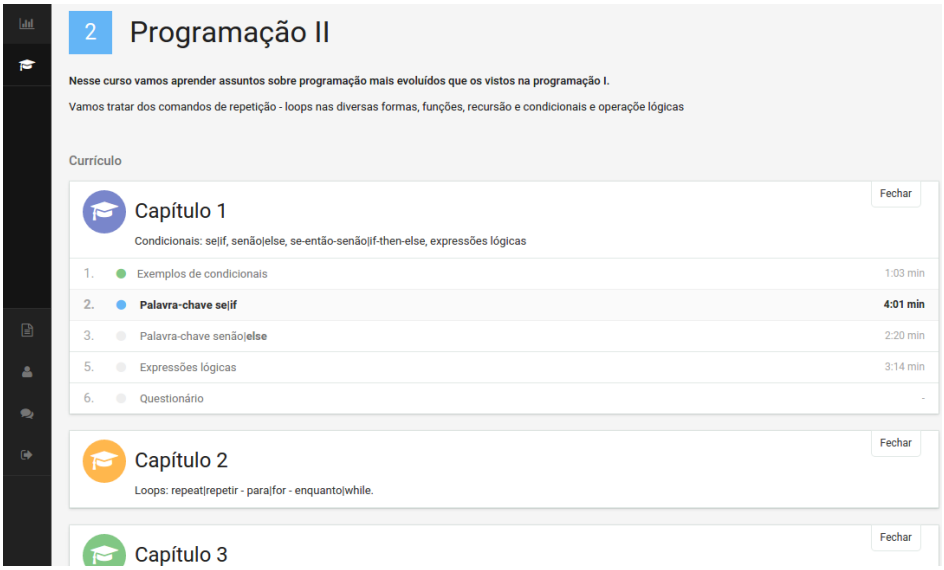

Figura 5.13. Interface da página específica dos cursos/matérias.

Avaliação on-line.

A página de avaliação on-line é responsável pelo *front-end* do sistema de avaliação dos alunos. Esta página apresenta uma parte distinta das já descritas: a barra lateral direita. Nesta barra, o aluno pode observar o tempo que ainda tem para finalizar a tarefa, assim como as questões que já fez e que estão por vir. Na parte central da página é mostrada a questão vigente, com a pergunta, respostas possíveis e no final um botão para o aluno confirmar sua resposta. Na parte superior o aluno pode observar sua pontuação corrente e o número de questões da avaliação que está fazendo. Essas tarefas são posteriormente apresentadas no widget de tarefas – desempenho recente. A figura 5.14 apresenta a interface da página de avaliação on-line do desempenho do aluno.

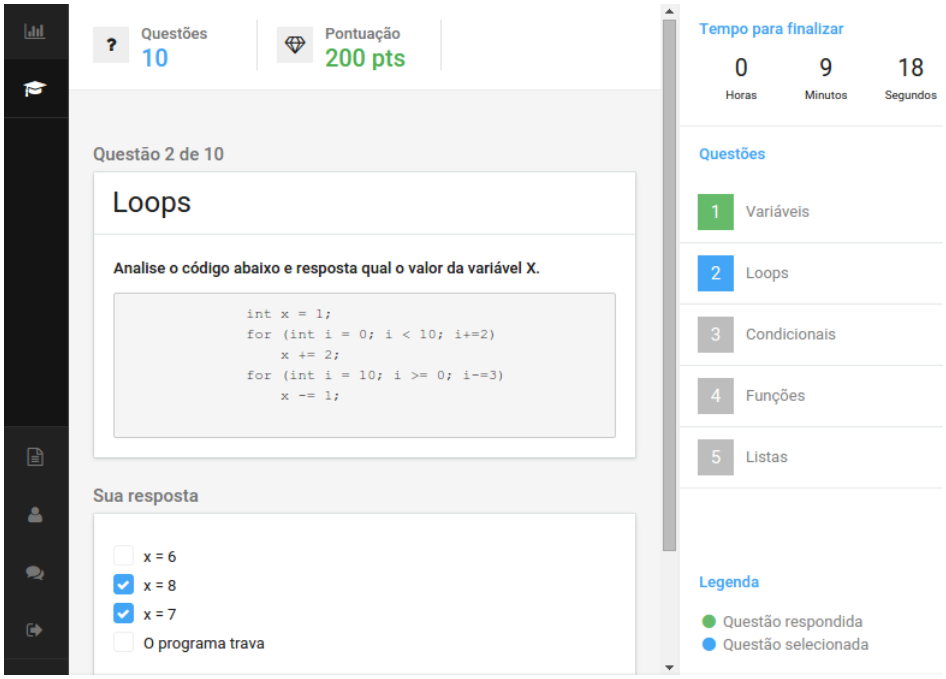

Figura 5.14. Interface da página de avaliação on-line.

Capítulo 6

Conclusões e Perspectivas

O objetivo principal deste trabalho de conclusão de curso foi planejar, desenvolver e avaliar uma plataforma para acompanhamento dos cursos de programação para a escola I Do Code. Todo o trabalho técnico envolvendo programação, assim como planejamento das funcionalidades do ambiente computacional e sua adaptação para o caso específico, constituiu um investimento intenso de pesquisa em várias áreas que envolveu muita leitura e comprometimento.

Projetos web são particularmente desafiadores, pois envolvem diversas áreas do conhecimento. Os principais desafios se deram em relação aos seguintes temas: muitas integrações estabelecidas, extenso conhecimento sobre linguagens web, criação de interfaces, usabilidade intuitiva e aspectos teóricos como segurança e suas soluções. São muitos os passos necessários para ir da idealização de uma aplicação até sua finalização, o resultado aqui mostrado foi somente possível pelos anos anteriores estudando programação, construindo websites, participando de competições de programação e vivenciando experiências de empreendedorismo, para entender por exemplo, como funciona o ciclo de um software.

Embora o objetivo principal tenha sido a plataforma em si mesma, o estabelecimento da empresa, assim como todo a formatação do negócio, foi um grande desafio que tem se mostrado muito produtivo e estimulante. Neste contexto, o curso de graduação em Engenharia de Controle e Automação teve um papel decisivo para o sucesso do projeto como um todo. As habilidades desenvolvidas no curso, por meio das várias disciplinas desde os Cálculos passando por robótica, engenharia de software até a programação foram muito importantes.

Outra experiência muito significativa foi o intercâmbio nos Estados Unidos. O ambiente das universidades americanas é especial, promove muita conversas construtivas e foi o fator essencial que permitiu a formatação das idéias, idealização do conceito

e entender mais a importância que isso terá no futuro dos jovens.

6.1 Trabalhos Futuros

Como todo software, há sempre partes que podem criadas e outras melhoradas.

Pontos relacionados a usabilidade e design serão os primeiros intensamente observados. Como cada classe de usuário é diferente, é interessante entender o modo que a aplicação é vista e entendida. Quais são os pontos que a interface não está intuitiva, e também o posicionamento de elementos. Em relação ao design, o que ocorre na maioria dos softwares é que algumas partes não estão adequadas, seja pelo tamanho, cor ou fonte, existe uma composição correta de fatores que são necessários para que a utilização seja adequada.

Outro aspecto importante são as funcionalidades. Por um lado alterações devem ser feitas nas existentes, por outro novas deverão ser criadas de acordo com a necessidade. Em todo software é comum adequações, seja em pontos simples como as extensões de arquivos suportadas para upload no servidor, ou até mudanças estruturais. Uma das funcionalidades já idealizadas é a adaptação da plataforma para adição de elementos dinâmicos de recomendação de conteúdos, para que tarefas sejam recomendadas a quem precisa, tudo de um modo automático.

Como já explicitado no trabalho, umas das vantagens de software web é que alteração podem ser rapidamente feitas e testadas, portanto acreditamos que esta versão inicial seja apenas um ponto de partida para inúmeras otimizações que serão feitas com os anos a seguir.

Referências Bibliográficas

[1] Amazon, A. (2010). Amazon web services. *Available in: http://aws. amazon. com/es/ec2/(November 2012)*.

[2] Booch, G.; Rumbaugh, J. & Jacobson, I. (2006). *UML: guia do usuário*. Elsevier Brasil.

[3] Cheswick, W. R.; Bellovin, S. M. & Rubin, A. D. (2005). *Firewalls e segurança na Internet*. Bookman.

[4] Conte, T.; Mendes, E. & Travassos, G. H. (2005). Processos de desenvolvimento para aplicações web: Uma revisão sistemática. Em *Proceedings of the 11th Brazilian Symposium on Multimedia and Web (WebMedia 2005)*, volume 1, pp. 107--116.

[5] Green, B. & Seshadri, S. (2013). *AngularJS*.

[6] Mendes, A. (2002). Duas arquiteturas de software.

[7] Merrill, D. (2006). Mashups: The new breed of web app. *IBM Web Architecture Technical Library*, pp. 1--13.

[8] Palmer, J. W. (2002). Web site usability, design, and performance metrics. *Information systems research*, 13(2):151--167.

[9] Pressman, R. S. (2011). *Engenharia de software*. McGraw Hill Brasil.

[10] Ribeiro, B. & Albuquerque, R. (2002). *Segurança no desenvolvimento de software*. Elsevier Brasil.

[11] Stein, L. D. (1998). *Web Security: A step-by-step reference guide*. Addison-Wesley Longman Publishing Co., Inc.