UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE COMPUTAÇÃO
CURSO DE CIÊNCIA DA COMPUTAÇÃO

EDUARDO GOMES PIETRANTONIO

DESENVOLVIMENTO DE RUBRICAS PARA VALIDAÇÃO DE DESIGN RESPONSIVO

TRABALHO DE CONCLUSÃO DE CURSO

MEDIANEIRA

2021

EDUARDO GOMES PIETRANTONIO

DESENVOLVIMENTO DE RUBRICAS PARA VALIDAÇÃO DE DESIGN RESPONSIVO

Trabalho de Conclusão de Curso apresentado ao Departamento Acadêmico de Computação da Universidade Tecnológica Federal do Paraná como requisito parcial para obtenção do título de "Bacharel em Ciência da Computação".

Orientador: Prof. Dr. Fernando Schütz

MEDIANEIRA

2021

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Diretoria de Graduação e Educação Profissional
Coordenação do Curso de Ciência da Computação

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

TERMO DE APROVAÇÃO

DESENVOLVIMENTO DE RUBRICAS PARA VALIDAÇÃO DE DESIGN RESPONSIVO

Por

EDUARDO GOMES PIETRANTONIO

Este Trabalho de Conclusão de Curso foi apresentado às 10:30h do dia 13 de Agosto de 2021 como requisito parcial para a obtenção do título de Bacharel no Curso de Ciência da Computação, da Universidade Tecnológica Federal do Paraná, Câmpus Medianeira. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Dr. Juliano Rodrigo Lamb
UTFPR - Câmpus Medianeira

Prof. Dr. Ricardo Sobjak
UTFPR - Câmpus Medianeira

Prof. Dr. Fernando Schütz
UTFPR - Câmpus Medianeira

A folha de aprovação assinada encontra-se na Coordenação do Curso.

RESUMO

PIETRANTONIO, Eduardo Gomes. DESENVOLVIMENTO DE RUBRICAS PARA VALIDAÇÃO DE DESIGN RESPONSIVO. 50 f. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade Tecnológica Federal do Paraná. Medianeira, 2021.

Dados do Sistema Global para Comunicações Móveis mostram que o número de usuários de smartphones ultrapassou a marca de 5,1 bilhões. Com o número de diferentes dispositivos em crescimento e mais usuários utilizando os dispositivos para diferentes tarefas, alguns conteúdos não são apresentados adequadamente em todos os dispositivos, por esse motivo Marcotte (2010) propôs um design que facilitasse o desenvolvimento de um sistema para suportar diferentes dispositivos, chamado de design responsivo. Com esse modelo o código escrito adapta o layout para diferentes dispositivos sem que fosse necessário escrever diferentes códigos. Uma das técnicas utilizadas é o método *mobile first* onde se desenvolve primeiramente para telas menores, indo para telas maiores. Este trabalho teve como objetivo analisar os métodos de design responsivo e utilizá-los na criação de sistemas com o auxílio de técnicas da engenharia de software. De maneira a avaliar esses designs, após a criação dos mesmos, a forma de validação da qualidade do design responsivo criado foi feita a partir de rubricas, criadas com critérios definidos a partir das necessidades desses sistemas, como *media queries*, *breakpoints*, *mobile first*, entre outras. Assim a utilização dessas rubricas podem ser adaptadas para diferentes sistemas, realizando pequenas alterações específicas de acordo com o domínio escolhido para o sistema.

Palavras-chave: engenharia de software, aplicações web, sites da web

ABSTRACT

PIETRANTONIO, Eduardo Gomes. DEVELOPMENT OF RUBRICS FOR RESPONSIVE DESIGN VALIDATION . 50 f. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade Tecnológica Federal do Paraná. Medianeira, 2021.

Data from the Global System Mobile for Communications show that the number of smartphone users has exceeded the 5.1 billion mark. With the number of different devices growing and more users using the devices for different tasks, some content is not presented properly on all devices, for this reason Marcotte (2010) proposed a design that would facilitate the development of a system to support different devices, called responsive design. With this model, the written code adapts the layout to different devices without having to write different codes. One of the techniques used is the mobile first method, where it is developed primarily for smaller screens, going to larger screens. This work aimed to analyze the responsive design methods and use them in the creation of systems with the aid of software engineering techniques. To evaluate these designs, after their creation, the way of validating the quality of the responsive design created was based on rubrics, created with criteria defined from the needs of these systems, such as media queries, breakpoints, mobile first, among others. Thus, the use of these items can be adapted for different systems, making small specific changes according to the domain chosen for the system.

Keywords: software engineering, web applications, websites

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CSS	Cascading Style Sheets
CSIEEE	Institute of Electrical and Electronics Engineers
ES	Engenharia de software
GSM	Global System Mobile for communication
HTML	Hypertext Markup Language
SWEBOK	Software Engineering Body of Knowledge
UML	Unified Modeling language

SUMÁRIO

1 INTRODUÇÃO

De acordo com dados da Global System Mobile for Communications (GSM, do inglês, Sistema Global para Comunicações Móveis), associação que reúne a indústria de telefones celulares, em 2020 o número de usuários de smartphones chegou em 5,1 bilhões, com a grande maioria desses usuários tendo acesso à internet (GSM Association, 2020). Isso mostra a força desse setor e torna fundamental que o uso desses aparelhos seja o mais agradável possível.

Os usuários utilizam esses dispositivos para as mais diversas tarefas, mas nem sempre o conteúdo é apresentado de forma adequada para os dispositivos em que são abertos, pois o sistema não é criado de forma adequada.

Uma técnica utilizada por empresas é criar dois sites, onde um possui o endereço normal, como “nomedosite.com.br” e a versão mobile é feita inserindo um “m” antes do endereço principal, por exemplo “m.nomedosite.com.br”. Essa forma de lidar com versões diferentes de sites não é ideal, pois é necessário criar dois códigos diferentes para o mesmo site.

Os navegadores de celulares e computadores conseguem adaptar a exibição da mesma tela para diferentes dispositivos caso o desenvolvimento do sistema seja feita de forma responsiva, não necessitando utilizar algum programa específico para rodar o design responsivo.

Os sistemas responsivos servem para tornar mais adequado o uso em diferentes aparelhos. Seu objetivo é de ter o seu formato adaptado de um smartphone, tablet, monitores desktop, aproveitando toda sua tela, minimizando a dificuldade do usuário em navegar no sistema (MARCOTTE, 2010). Para uma empresa que deseja prestar o melhor serviço para seus clientes, possuir tais sistemas responsivos atualmente se torna imprescindível. Com designs responsivos isso é possível sem ter a necessidade de se construir códigos específicos para cada dispositivo.

Neste contexto, a interface gráfica de uma aplicação que seja construída por meio de técnicas responsivas adapta-se à tela do dispositivo identificado, quando esta é requisitada.

A engenharia de software web está sendo utilizada de forma emergente. Ela é multidisciplinar, pois engloba características de diversas áreas, como análises de sistemas e design, engenharia de software, interação humano-computador, interface, entre várias outras coisas. Ela busca entender as necessidades específicas da Web e resolver problemas de

desenvolvimento, visando resolver os problemas de forma mais rápida e com um ciclo de vida menor, devidos as diversas mudanças que possam ocorrer (PRESSMAN; LOWE, 2009).

As rubricas são esquemas para se classificar produtos ou comportamentos (BIAGIOTTI, 2005), que podem ser utilizadas em diversos assuntos e áreas, incluindo a área da computação.

Este trabalho teve como objetivo a criação de rubricas para realizar a avaliação e validação do sistema responsivo, definindo critérios necessários para os mesmos e fazendo a validação do sistemas criado.

Apesar de já existirem metodologias de criação do design responsivo, não foram encontradas metodologias a partir da utilização de rubricas que podem ser reutilizadas para diferentes sistemas.

Este trabalho buscou utilizar uma abordagem mais documentada do processo de validação utilizado no mercado, em que não é tão detalhado.

1.1 OBJETIVOS GERAL E ESPECÍFICOS

O objetivo geral deste trabalho é desenvolver artefatos para avaliação de sistemas responsivos, baseados em métodos e técnicas da engenharia de software tradicional e engenharia de software web.

São objetivos específicos deste trabalho:

- Desenvolver um estudo de caso experimental para aplicar o design responsivo;
- Aplicar as técnicas responsivas e criar protótipos com características responsivas;
- Desenvolver rubricas para fazer a validação da interface gráfica desenvolvida;
- Avaliar o site desenvolvido a partir das rubricas definidas, classificando os designs dos sistemas propostos, de acordo com a pontuação obtida.

1.2 JUSTIFICATIVA

A exigência dos usuários vem crescendo na medida em que mais pessoas têm acesso à tecnologia e com isso passam a se deparar com problemas em seu uso. O número de smartphones vem crescendo e chegou a ultrapassar o número de computadores pessoais no Brasil (MEIRELLES, 2019).

Apesar disso, muitos sistemas ainda não vem se adaptando na mesma velocidade, e isso vale para os vários sistemas não responsivos presentes no mercado, fazendo com que sua utilização nos smartphones não sejam agradáveis para o usuário.

Existem diferentes técnicas que são voltadas a validação da interface gráfica e usabilidade, como no livro de Engenharia Web (PRESSMAN; LOWE, 2009), mas que não avaliam especificamente o design responsivo, que pode tornar mais fácil a criação e avaliação desses sistemas para que possam ser utilizadas para melhorar o que foi criado e um lançamento do sistema com maior qualidade.

O intuito desse trabalho é utilizar as rubricas após a fase de protótipos e criação de um sistema, para após analisar os resultados obtidos, verificar a possibilidade de alterações para que facilite o desenvolvimento de melhorias e tornar mais agradável a utilização desses sistemas pelo usuário final tanto em seus dispositivos móveis quanto em seus outros aparelhos.

1.3 ORGANIZAÇÃO DO TRABALHO

Esse documento foi organizado da seguinte forma. O Capítulo 2 mostra brevemente a história da engenharia de software e sua utilização no desenvolvimento de softwares. Após isso, é apresentada outra versão da engenharia de software que foi aperfeiçoada para sistemas web, a engenharia web. É apresentado o que é e como é feito o desenvolvimento de um sistema responsivo, com ênfase nas necessidades de métodos utilizados na sua construção e a apresentação de rubricas utilizadas em outros domínios. A metodologia utilizada se encontra no Capítulo 3, nele são descritas todas as etapas para o desenvolvimento do projeto. No Capítulo **??** são apresentados os resultados e discussões do trabalho. As considerações finais e trabalhos futuros são apresentados no Capítulo 5.

2 LEVANTAMENTO BIBLIOGRÁFICO

Nessa seção é apresentada a utilização da engenharia de software, assim como seu funcionamento e disciplinas que são utilizadas como ajuda na sua criação, pois é a base sobre a engenharia de software web. Também são apresentadas a definição e utilização da engenharia de software web, além de desenvolvimento de sistemas responsivos e onde esses temas se encaixam no desenvolvimento de um software. Breve apresentação sobre as rubricas com exemplos feitos por outros autores e trabalhos correlatos que servem de apoio para este trabalho.

2.1 ENGENHARIA DE SOFTWARE

É possível encontrar software em diversas partes do cotidiano, seja em celulares, computadores, tablets, notebooks e até mesmo em dispositivos menos comuns, como brinquedos, aparelhos na cozinha, entre outros.

A Engenharia de Software (ES) é um conjunto de técnicas e metodologias e compreende a aplicação de alguns princípios da engenharia no desenvolvimento e criação de um software (PRESSMAN, 2005). É uma área de conhecimento da computação que visa principalmente a especificação, desenvolvimento e manutenção de sistemas de software, onde são aplicadas tecnologias, práticas e conhecimento de diversas disciplinas (BOURQUE; FAIRLEY, 2014).

A ES visa melhorar a qualidade de desenvolvimento e criação de softwares, buscando aplicar técnicas para que a experiência seja melhor aproveitada. Ela acaba sendo uma área multidisciplinar, e para que essas práticas sejam implementadas de uma melhor forma, foi criado um guia de referência para aplicar esses métodos.

2.1.1 Corpo de conhecimento da engenharia de software

O Software Engineering Body of Knowledge (SWEBOK, do inglês, corpo de conhecimento da engenharia de software) é um guia de mecanismo desenvolvido pela Sociedade de Computação IEEE (CSIEEE, do inglês, sociedade de computação do instituto de engenheiros eletricistas e eletrônicos), sua primeira versão foi criada em 2004 para aquisição de conhecimento necessário para se colocar em prática as técnicas da engenharia de software em seus sistemas. Nele é fornecido o corpo exigido de conhecimento e práticas recomendadas. A versão utilizada neste trabalho é a terceira, lançada em 2014.

O SWEBOK foi estabelecido com os seguintes objetivos (BOURQUE; FAIRLEY, 2014):

- Promover uma visão consistente da ES globalmente, que foi apoiada por um processo de desenvolvimento que envolveu aproximadamente 150 revisores de 33 países;
- Especificar o escopo e esclarecer o lugar da engenharia de software em relação a outras disciplinas;
- Caracterizar o conteúdo da disciplina de ES;
- Fornecer acesso tópico ao corpo de conhecimento da engenharia de software;
- Fornecer uma base para o desenvolvimento curricular e para o material de certificação e licenciamento individual.

Tabela 1 – As 15 áreas de conhecimento do SWEBOK

Requisitos de software
Design de software
Construção de software
Teste de software
Manutenção de software
Gerência de configuração de software
Gerência da engenharia de software
Processos da engenharia de software
Ferramentas e métodos da engenharia de software
Qualidade de software
Práticas profissionais em engenharia de software
Economia da engenharia de software
Fundamentos da computação
Fundamentos da matemática
Fundamentos da engenharia

Fonte: Adaptado de Bourque e Fairley (2014).

Como apresentado na Tabela 1, são 15 as áreas de conhecimento do SWEBOK, definidas pela CSIEEE. Elas são utilizadas para boas práticas de ES, sendo utilizadas da seguinte forma:

- **Requisitos de software:** É uma área que se preocupa com o levantamento, análise, especificação e validação dos requisitos de software, sendo uma área de extrema importância para que um sistema não tenha péssimo desempenho, já que um requisito de software deve resolver um problema da vida real (KOTONYA; SAWYER, 2014).
  Esse processo é preferencialmente feito na seguinte ordem, de acordo com Kotonya e Sawyer (2014):
  - **Fundamentos dos requisitos de software**: definição, requisitos do produto e processo, requisitos funcionais e não-funcionais;
  - **Processo de requisitos**: modelos, atores, suporte e gerenciamento e a qualidade e evolução do processo;
  - **Levantamento de requisitos**: fontes e técnicas de levantamento;
  - **Análise dos requisitos**: classificação, modelo conceitual, design de arquitetura e alocação, ajuste e análise formal dos requisitos;
  - **Especificação dos requisitos**: sistema de definição do documento, sistema de especificação dos requisitos, especificação dos requisitos do software;
  - **Validação dos requisitos**: revisão dos requisitos, prototipação, validação do modelo e testes de aceitação.
- **Design de software:** O design de um software é considerado tanto o processo de declaração da arquitetura, componentes, interface de um sistema, como também o produto final que é o resultado disso (SUN, 2014).
  Na realização desse processo, são abordados diversas formas de design, de acordo com Sun (2014), são definidos em:
  - **fundamentos do design de software**: conceitos gerais de design, contexto, processo e princípios;
  - **problemas chave em design de software**: concorrência, controle e manipulação de eventos, persistência dos dados, distribuição de componentes, tolerância de falhas, interação e apresentação, segurança;
  - **estrutura e arquitetura de software**: pontos de visão arquiteturais e estruturais, estilos arquiteturais, padrões de design, arquitetura de designs de decisão;
  - **design da interface do usuário**: princípio de design da interface de usuário, problemas de design de interface, processo de design da interface, localização e internacionalização;

  - **análise e avaliação da qualidade do design**: atributos de qualidade, técnicas de avaliação e análise da qualidade, medidas;
  - **notações de design de software**: descrição estrutural e comportamental.
- **Construção de software:** A construção de um software é relacionada pela criação com detalhes de um software funcional, atendendo alguns critérios como, combinação de código, verificação, teste de unidade e integração e depuração (PENG, 2014).

  De acordo com Peng (2014), esse processo é feito da seguinte forma:
  - **fundamentos da construção**: minimizar a complexidade do software, antecipar mudanças, construído para verificação, reuso e padrões na construção;
  - **gerenciar a construção**: construção em modelos de ciclo de vida, planejamento e medição da construção;
  - **considerações práticas**: construção de design, linguagens, codificações, construção de teste, construção para reuso, qualidade, integração;
  - **tecnologia de construção**: manipulação de erros, manipulação de exceções e tolerância a falhas, modelos executáveis, entre outras.
- **Teste de software:** Consiste na verificação de que um programa seja executado e se porte do mesmo modo que era esperado durante sua criação. De acordo com Bertolino e Marchetti (2014) as formas de testes devem atender alguns requisitos, tais como, ser dinâmico, ou seja executando o sistema e inserindo entradas selecionadas, vendo seu comportamento:
  - **deve ser finito**: existem muitas formas de testar, mesmo em programas mais simples, que poderiam durar até anos, dessa forma deve se limitar os testes, priorizando os critérios mais importantes;
  - **deve ser selecionado**: os dados de teste serão selecionados, de forma a maximizar as verificações mais importantes, escolhendo os melhores testes;
  - **deve ser esperado**: quando ocorre a verificação se o resultado obtido após os testes, é o resultado esperado ou não, caso não seja o esperado, será necessário fazer mudanças.
- **Manutenção de software:** Após o desenvolvimento do software que satisfaça todos os requisitos necessários, o sistema pode mudar, e após estar em operação ocorrem mudanças de ambiente, novos requisitos, entre outras coisas, tornando necessária essa manutenção (APRIL; KAJKO-MATTSSON, 2014).

  A manutenção do software faz parte do ciclo de vida de um software. Apesar do desenvolvimento do software levar muito mais atenção do que a parte de manutenção (APRIL; KAJKO-MATTSSON, 2014).

- **Gerência de configuração de software:** Em um sistema há a combinação de elementos organizados para que se atinja o objetivo final. Para se configurar um sistema são várias as questões que se devem levar em conta, como versões de hardware, características de software, entre outras (CHAMPAGNE; APRIL, 2014).

  A gerência dessas configurações de software é um processo de apoio ao ciclo de vida do software, desenvolvimento e manutenção;
- **Gerência da engenharia de software:** É uma atividade de gerência da aplicação, onde é feito o planejamento, coordenação, medição, controle e relatório, para garantir que os produtos e os serviços da ES sejam entregues de forma efetiva, eficiente e que seja útil aos clientes (MCDONALD, 2014).

  Esse processo é realizado em três níveis: gerenciamento de infraestrutura e organizacional, gerência de projeto e gestão de medição do programa (MCDONALD, 2014);
- **Processos da engenharia de software:** Consiste de um conjunto de atividades relacionadas entre si em que transformam as entradas em saídas, ao mesmo tempo em que consome os recursos para efetuar essa transformação (REILLY; FAIRLEY, 2014).

  Eles têm a função de desenvolver, manter e operar o software, requisitos, design, construção, testes, gerência de configuração, entre outros processos de ES (REILLY; FAIRLEY, 2014);
- **Ferramentas e métodos da engenharia de software:** Possui como objetivo deixar a atividade de criação mais sistemática, repetitiva e no final mais propensa ao sucesso. A utilização de modelos permite uma melhor aproximação para solucionar e analisar os problemas.

  Segundo (SIOK, 2014), esses métodos e ferramentas são divididos em quatro etapas principais:
  - **Modelagem**: onde são discutidos as práticas gerais de modelagem e os princípios;
  - **Tipos de modelos**: onde são definidos os modelos e suas características que serão utilizados na prática de ES;
  - **Análise de modelos**: onde é escolhida a técnica de análise que será usada para validar todas as coisas necessárias;
  - **Métodos de ES**: onde é escolhido o método que será usado na ES.
- **Qualidade de software:** Pode ser considerada a capacidade de seu software atender as expectativas e necessidades que foram constatadas na fase de planejamento do software.

  Verificar se os requisitos são atendidos de acordo com as necessidades dos clientes, e se era aquilo que o mesmo esperava encontrar ao fim do projeto (BLAINE; BISWAS, 2014);

- **Práticas profissionais em engenharia de software:** É a parte que trata sobre todo o conhecimento, especialidades e atitudes que esses profissionais devem possuir para aplicar a ES de forma responsável, prática e ética.

  Os engenheiros de software devem possuir essas habilidades, pois precisam lidar com problemas únicos, por isso é necessária uma prática profissional para cuidar desses assuntos (SHEFFIELD; ZOU, 2014);
- **Economia da engenharia de software:** Deve passar também pelo engenheiro de software a parte econômica do processo de construção do projeto, apesar de essa não ser sua especialidade, pois ele trata tanto da parte técnica quanto da parte de negócios.

  É preciso levar muitas coisas em consideração para se aproveitar o máximo possível, como a gestão de valor agregado, taxa de retorno integrado, o retorno mínimo que seria considerado o aceitável, o ciclo de vida do desenvolvimento e do produto em si, o retorno do investimento feito, retorno do capital que foi empregado e o custo total das operações (EBERT, 2014);
- **Fundamentos da computação:** A importância da computação na ES é clara, não é possível um software, sem o local onde ele possa ser executado, portanto o conhecimento sobre esses fundamentos é essencial para todos os engenheiros de software que entendam sobre os fundamentos da computação, em ordem de realizar um bom software (ZOU, 2014).
- **Fundamentos da matemática:** É focado no sentido da lógica e raciocínio que são os principais focos em um engenheiro de software precisa se dedicar (CHAKI, 2014).

  Matemática em sua definição é algo que indica uma precisão, não podendo ter uma interpretação ambígua ou errado de alguma coisa que foi definida nas etapas anteriores do projeto. Portanto o engenheiro de software precisa fazer as etapas da forma mais clara e precisa possível, para não haver esses tipos de erro;
- **Fundamentos da engenharia:** De acordo com a Bourque e Fairley (2014), a engenharia é a aplicação de uma abordagem quantitativa sistemática e disciplinada para estruturas, máquinas, produtos, sistemas ou processos.

  Como dito anteriormente, a ES aplica conceitos e princípios da engenharia tradicional para a criação de software, portanto são utilizados diversos fundamentos dessa área de conhecimento, que sejam mais voltados para a computação e criação de software.

  De acordo com Bandyopadhayay e Willshire (2014) isso inclui utilizar alguns métodos, por exemplo:
  - Análise estatística;
  - Medição;

- Design de engenharia;
- Modelagem, prototipação e simulação;
- Padrões;
- Análise de causa principal.

Com a aplicação desse conhecimento, será possível ao engenheiro de software manter um software mais efetivo e eficiente (BANDYOPADHAYAY; WILLSHIRE, 2014).

A engenharia de software é muito abrangente e como apresentado anteriormente envolve uma diversidade de áreas de conhecimento que devem ser aplicadas e relacionadas com o projeto.

2.1.2 Disciplinas Relacionadas

Abrangendo diversas áreas de conhecimento como a ES faz, isso abre espaço para que muitas disciplinas sejam aplicadas e que precisem ser atendidas e respeitadas pelo engenheiro de software, tornando um trabalho ainda mais complexo.

Como apresentado na Tabela 2 as disciplinas que mais se destacam em um projeto de ES, podem ser descritas a partir das áreas de conhecimento apresentadas anteriormente.

Tabela 2 – As disciplinas relacionadas do SWEBOK

Engenharia da computação
Ciência da computação
Gestão geral
Matemática
Gestão de projetos
Gestão da qualidade
Engenharia de sistemas

Fonte: Adaptado de Bourque e Fairley (2014).

Por exemplo, nas engenharias de computação e de sistemas são aplicados os fundamentos da engenharia no geral e em suas áreas específicas. A ciência da computação que serve como base da criação da parte mais prática do software. As gestões geral, de projetos e de qualidade, servem para auxiliar o gerenciamento do projeto como um todo e em suas particularidades, controlar a qualidade do produto apresentado, garantindo que seja feito com qualidade, por meio também dos testes como visto anteriormente. E também os fundamentos

da matemática, utilizando toda sua lógica e raciocínio para criar um software menos ambíguo e sem problemas de interpretação, facilitando a vida do cliente e dos desenvolvedores.

2.2 ENGENHARIA DE SOFTWARE WEB

Assim como a engenharia de software tradicional, a engenharia de software web visa facilitar o desenvolvimento e a manutenção de softwares complexos, por meio de métodos, ferramentas e modelos específicos (BROD; KÄFER, 2009).

Engenharia web não é um clone da ES, apesar de ambas envolverem elementos parecidos. As diferenças vão além de que a ES visa softwares para programas de computador e a engenharia web no desenvolvimento de aplicações para a Internet, já que a engenharia web utiliza novas formas de abordar o desenvolvimento para aplicações baseadas em web (RINE, 2010).

2.2.1 A Web e aplicações Web

A web é algo indispensável para os negócios, seja para o comércio, mídia, governo, indústria, entretenimento, entre tantos outros elementos que estão presentes no cotidiano.

Nos primórdios da World Wide Web (Rede mundial de computadores), as aplicações web eram formadas apenas por arquivos de hipertextos conectados, apresentando para o usuário apenas simples textos e gráficos. À medida em que foi evoluindo, a Hypertext Markup Language (HTML, do inglês, Linguagem de Marcação de HiperTexto) foi expandida com novas ferramentas e tecnologias para desenvolvimento com maior capacidade de computação para criação das aplicações (PRESSMAN; LOWE, 2009).

2.2.2 Processo de Engenharia Web

De acordo com Pressman e Lowe (2009), os processos dominantes de desenvolvimento para as equipes são compostos por alguns critérios:

- **Confiabilidade:** aplicações que tem bom funcionamento, não sofrem problemas para funcionarem, não passam dados incorretos;
- **Usabilidade:** boas aplicações são aquelas que são fáceis de usar, em que os usuários não tenham maiores problemas ao utilizar, pois deve ser intuitiva. Uma aplicação com má usabilidade será facilmente substituída por outra se não satisfazer as necessidades do usuário;
- **Segurança:** deve lidar de forma segura com os dados dos usuários e outras informações para evitar processos, perdas financeiras, entre outros problemas legais;
- **Avaliabilidade:** os usuários vão sempre esperar que uma aplicação fique ativa 100% do tempo, então é extremamente importante que a aplicação fique o máximo de tempo possível online pois qualquer período de tempo, por menor que seja pode gerar perda de receitas para os clientes;
- **Escalabilidade:** o software deve ter sempre um espaço para crescer, para que se possam aplicar melhorias de acordo com algumas necessidades que surjam por meio de pedidos dos usuários e clientes;
- **Manutenção:** aplicações web estão sendo atualizadas constantemente, então é necessário ter uma facilidade na correção, adaptação e mudanças na página;
- **Eficiência:** ter um bom tempo de resposta, gerar páginas e os elementos da página de maneira rápida.
- **Intensidade da rede:** as aplicações ficam em uma rede, que deve servir a um conjunto diversificado de clientes;
- **Simultaneidade:** uma grande quantidade de pessoas pode acessar a aplicação Web simultaneamente e esses números irão variar constantemente;
- **Carga imprevisível:** a quantidade de usuários pode variar muito de um dia para outro;
- **Estética:** uma parte muito importante de uma aplicação Web é a sua aparência, principalmente dependendo do conteúdo disponibilizado por essa aplicação.

2.2.3 Principais diferenças entre ES tradicional x Engenharia Web

As principais diferenças entre a abordagem clássica da ES e a Engenharia Web, conforme Brod e Käfer (2009) são:

- **Ciclo de vida do produto desenvolvido:** essa é uma das principais diferenças já que a ES tradicional costuma ter um ciclo de vida maior, com uma evolução mais lenta, com menos mudanças. A engenharia web evolui constantemente, sempre com mudanças no desenvolvimento, possuindo um ciclo de vida menor;
- **Público alvo:** Na ES tradicional, muitas vezes uma aplicação tem um público definido de usuários em algumas situações, onde algo pode ser desenvolvido propriamente para esse público. Na engenharia web os usuários podem ser mais diversificados, sem uma quantidade definida e fica mais difícil imaginar o perfil de todos usuários que possam utilizar o sistema;
- **Tecnologias utilizadas:** A ES tradicional costuma utilizar tecnologias e técnicas mais estáveis já que esse tipo de software é desenvolvido ao longo de mais tempo, sem muitas alterações. Enquanto na engenharia web são mais constantes as mudanças, com a utilização de tecnologias que sofrem mais mudanças, pois estão sempre evoluindo.

A principal diferença observada é que a engenharia web tem um clico de vida rápido, onde mudanças acontecem o tempo todo e o sistema deve estar sempre preparado para alterações, além de seu público alvo estar sempre mudando e algumas tecnologias próprias para a verificação e aplicação dessas técnicas, enquanto a ES tradicional utiliza um modelo mais estático.

2.3 DESIGN RESPONSIVO

Quando iniciou-se o desenvolvimento para web, os sites eram desenvolvidos sem preocupação quanto à sua aparência em diferentes dispositivos. Em geral a maioria dos seus usuários acessavam aos sites desenvolvidos de computadores com configurações muito similares, fazendo com que o formato da tela não se distanciasse do formato original (GIURGIU; GLIGOREA, 2017).

Conforme as tecnologias estão avançando, novos dispositivos para exibição e

processamento vão surgindo, sendo mais difícil possuir um padrão específico de design.

Com isso Marcotte (2010), propôs que para facilitar o desenvolvimento, seja feito apenas um sistema que seja responsivo, ao invés de um design para cada dispositivo desejado. Este código escrito adaptaria o layout para as diferentes telas, utilizando tecnologias que permitissem o desenvolvimento responsivo. Esse design responsivo busca substituir os métodos utilizados anteriormente, em que eram criados sites diferentes para versões diferentes entre mobile, desktop, entre outros, dificultando a manutenção e atualização de todas essas versões.

A Figura 1 demonstra uma ilustração que simula um comportamento adequado de um sistema responsivo em diferentes dispositivos, que possuem diferentes tamanhos de telas e resoluções.

Figura 1 – Exemplo de modelos responsivos
Fonte: (GIURGIU; GLIGOREA, 2017)

Na criação de um sistema responsivo existem técnicas para facilitar o seu desenvolvimento, uma dessas técnicas é o método de *mobile first*, método proposto por Wroblewski (2011).

2.3.1 *Mobile first*

A metodologia *mobile first* tem como objetivo iniciar o desenvolvimento a partir de dispositivos com telas menores, para após isso adaptar e aplicar as mudanças necessárias para adaptação de telas maiores (WROBLEWSKI, 2011).

Como estará sendo trabalhado com a limitação da tela ser menor, é possível dar mais

foco para as funcionalidades mais importantes e essenciais, fazendo assim com que melhore a experiência do usuário (DEDIS, 2017).

Na Figura 2 é possível ver a sequência em que é feito o desenvolvimento, iniciando pelas telas menores, indo para a tela intermediária e após isso sendo pensado os elementos para as telas maiores.

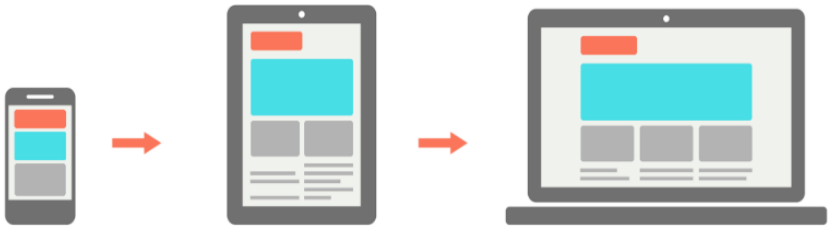

Figura 2 – Exemplo do método *mobile first*

Fonte: (SILVA, 2014)

O objetivo do design responsivo é justamente fazer com que os designs respondam bem para qualquer dispositivo, sem que fiquem itens escondidos na tela onde o usuário não tenha visão (PROSTT, 2013).

De acordo com Marcotte (2010) os três componentes fundamentais para o desenvolvimento do design responsivo são:

- Layout fluído;
- *Media queries*;
- Imagens e mídias flexíveis.

2.3.2 Layout fluído

O layout fluído ou também conhecido como grid flexível é o principal elemento de um web design responsivo (LOPES, 2013). O grid flexível juntamente com as imagens flexíveis são capazes de resolver praticamente todo o necessário no design responsivo (MOYEEN et al., 2017). Para que seja possível obter um layout fluído em um projeto web, a principal coisa a ser feita é não utilizar medidas absolutas no Cascading Style Sheets (CSS, do inglês, Folha de Estilo em Cascata) (ZEMEL, 2012).

As quatro formas de medidas no CSS mais utilizadas de acordo com Zemel (2012):

- Pixel (px): Unidade fixa mais utilizada em CSS;
- Ponto: São mais comumente utilizados para CSSs de impressão e também são unidades de tamanho fixo;
- Ems (em): É uma unidade de medida escalável, por exemplos em tamanhos de fonte, se uma fonte tem tamanho de 12pt, 1em equivale a 12pt e como ele é escalável por natureza, 2em seria igual a 24pt, por exemplo;
- Porcentagem: Essa unidade também é escalável e é muito parecida com a unidade de medida em, apesar de ser mais utilizada para lidar com larguras, margens, espaçamentos, etc.

A diferença entre elas pode ser vista na Figura 3, onde utilizando as medidas flexíveis é possível ter um melhor controle do seu design podendo ser adaptado para diferentes dispositivos, onde do lado esquerdo é utilizada a unidade de medida fixa (pixels) e do lado direito unidade flexível (porcentagem).

Trabalhar com medidas fixas não vai de acordo com o conceito de web adaptável, pois caso seja necessário redimensionar o seu navegador os componentes do site ficariam fixos, ou seja, utilizar layout fluído é trabalhar com medidas flexíveis (TOMAZINI et al., 2018).

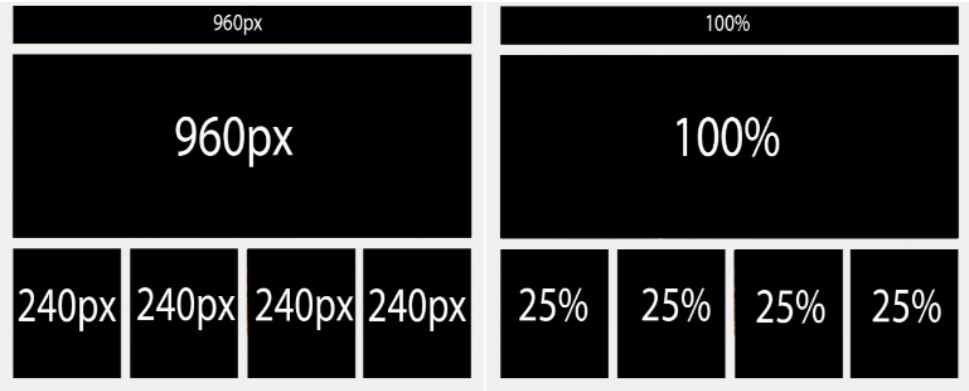

Figura 3 – Layout com unidades de medidas fixas x flexíveis

Fonte: (SILVA, 2014)

De acordo com Silva (2014) deve-se tomar cuidado ao deixar a página ocupar 100% das telas, pois isso pode apresentar resultados ruins, principalmente nos extremos em caso de telas muito grandes ou pequenas. Para solucionar isso é utilizado limites máximo e mínimo para o tamanho, essas unidades fixas devem ser utilizadas apenas para limitar o tamanho, enquanto o restante deve ser feito utilizando medidas flexíveis.

2.3.3 *Media queries*

Esse método é utilizado para aplicar os estilos CSS dependendo das características do dispositivo em que a página será aberta, sendo as *media queries* responsáveis por escolher o melhor layout de acordo com o dispositivo (SILVA, 2014). Ela funciona como uma consulta à mídia para qual dispositivo está requisitando a abertura da aplicação e escolhendo as melhores configurações para abrir a página. Portanto é definido um tamanho de tela e definidas as configurações que serão utilizadas nos dispositivos com esse tamanho, criando assim uma *media querie*.

As *media queries* surgem para que os elementos possuam comportamentos diferentes, podendo até mudar de posição, se necessário.

(a) Layout sem *media queries* (b) Layout com *media queries*

Figura 4 – *Media queries*

Fonte: (LOPES, 2013)

É possível ver na Figura 4 (a) que apesar das imagens se adaptarem ao tamanho da tela, os textos acabam se sobrepondo, o que deixa os elementos da página ilegíveis. Enquanto que na Figura 4 (b) as imagens e textos se adaptam de acordo com o dispositivo utilizado.

Seria impossível criar uma *media querie* para cada dispositivo existente, já que sempre surgem novos dispositivos e é com esse intuito que foram criados os *breakpoints* (LOPES,

2013).

Cada site necessita de seus próprios *breakpoints* e de acordo com França (2015), esses são os passos para encontrá-los:

- Abra a página original no navegador;
- Redimensione a janela até encontrar um ponto em que o design pareça inadequado;
- Ao encontrar esse ponto em que a janela "quebre", em caso da utilização da técnica *mobile first* comece com a janela pequena e vá aumentando, caso não tenha utilizado, faça o inverso;
- Por fim, recarregue a página, verifique se as mudanças melhoraram o design e procure o próximo *breakpoint* até chegar ao resultado desejado.

2.3.4 Imagens e mídias flexíveis

Assim como o conteúdo precisa se adaptar às mais diferentes resoluções, o restante dos recursos também precisa acompanhar essa adaptação, como as imagens, sendo necessário que essas mídias e imagens também sejam flexíveis (FRANÇA, 2015). Como as imagens são feitas por pixels, que é uma unidade de medida fixa, elas não podem ser redimensionadas simplesmente com a resolução da tela, mas ainda não há uma solução padrão para resolver esse tipo de problema (SILVA, 2014).

Uma solução utilizada é dizer que as imagens terão uma largura máxima de 100% e a imagem será contraída ou esticada de acordo com a resolução que se encontra (DEDIS, 2017).

De acordo com Silva (2014) isso acaba não sendo uma solução perfeita pois se a resolução da imagem for muito menor que o dispositivo, ela será esticada e será possível ver os pixels. Uma forma de resolver isso é utilizar várias versões dessa imagem com resoluções diferentes de acordo com os dispositivos que possam abri-la e a imagem mais apropriada será escolhida a partir das *media queries* (SILVA, 2014).

2.3.5 *Meta tag viewport*

O viewport é o espaço disponível para que a página seja renderizada no navegador,

dependendo do tamanho da tela do navegador (LOPES, 2013).

Essa *meta tag viewport* tem a função de especificar para o dispositivo em qual escala ele deve renderizar seu design e caso não seja especificado, o próprio navegador irá encolher a página para caber na tela do dispositivo menor (DEDIS, 2017).

Portanto ao tentar utilizar esse método sem especificar, você terá um resultado satisfatório utilizando a página em seu computador, mas será uma tentativa frustrada ao tentar abrir em um dispositivo móvel (WARD, 2017).

Como pode-se observar na Figura 5, quando não especificado a *meta tag viewport*, a página se adapta ao tamanho da menor tela, mas prejudicando a experiência do usuário com menor tamanho dos elementos utilizados, sendo necessário utilizar o zoom para visualização.

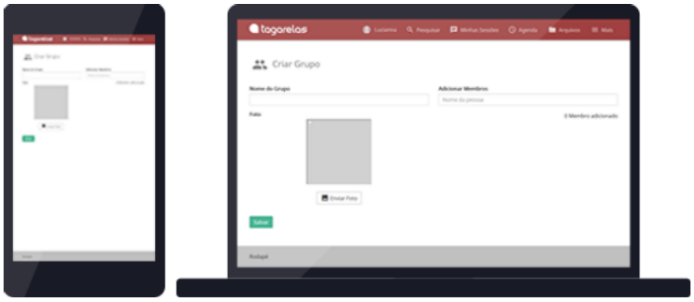

Figura 5 – Exemplos da página sem *meta tag viewport*

Fonte: (DEDIS, 2017)

Já na Figura 6 pode-se observar que os elementos presentes na página ficam bem mais visíveis e melhorando assim a experiência do usuário.

Figura 6 – Exemplo da página com *meta tag viewport*

Fonte: (DEDIS, 2017)

2.4 RUBRICA

Rubricas podem ser consideradas como esquemas para se classificar produtos ou comportamentos, por exemplo, redações, ensaios, trabalhos de pesquisa, apresentações, atividades, entre outras coisas (BIAGIOTTI, 2005).

Uma rubrica funciona como uma ferramenta para comunicar, classificar e avaliar o desempenho de algo pela utilização de uma pontuação, de acordo com os critérios definidos do que se espera sobre determinado assunto (FERRAZ et al., 2016). Cada rubrica é criada de acordo com a pontuação e critérios definidos pelo criador, podendo variar de acordo com o que está sendo avaliado e quem está avaliando.

De acordo com Biagiotti (2005), as rubricas necessitam de algumas características para que possam ser consideradas uma boa ferramenta de avaliação, algumas delas são:

- Facilidade: as rubricas devem facilitar a avaliação de itens complexos;
- Objetividade: ajudar a avaliar de forma objetiva os itens desejados;
- Gradativa: deve ser utilizado para itens individuais, conjuntos e também o produto como um todo;
- Transparência: deixar esse processo todo transparente, para que todos possam ter controle do que está sendo avaliado;
- Associativa: associar essa avaliação, para verificar se a partir do item avaliado, o objetivo final do produto será alcançado;
- Reutilização: devem poder ser utilizadas para diferentes produtos, sofrendo apenas adequações de acordo com as diferenças entre os produtos avaliados;
- Padronização: deve seguir um padrão de avaliação para que facilite o entendimento da avaliação de cada item.

Como pode-se observar na Tabela 3, é possível definir nas rubricas quais serão os critérios de avaliação a partir dos itens que serão avaliados e de acordo com a avaliação, é dada uma pontuação para os requisitos atendidos pelo sistema, podendo receber uma nota máxima, se o item avaliado cumprir totalmente o critério, uma nota mediana se o item avaliado cumprir apenas parcialmente o critério ou uma nota mínima caso o item avaliado não cumpra o critério definido pelo avaliador.

Tabela 3 – Exemplo Rubrica com critérios de avaliação

Itens avaliados	Pontuação máxima	Pontuação média	Pontuação mínima
Item de avaliação 01	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 02	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 03	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 04	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 05	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 06	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 07	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 08	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 09	Valor recebido	Valor recebido	Valor recebido
Item de avaliação 10	Valor recebido	Valor recebido	Valor recebido
Pontuação total	Valor total	Valor total	Valor total

Fonte: Autoria Própria.

Enquanto que na Figura 4 pode-se observar que após dadas as pontuações de acordo com cada item avaliado e com os requisitos cumpridos por eles, é dado a nota na rubrica para verificar se o produto avaliado cumpriu o que se esperava dele e se ele pode ser avaliado positivamente de acordo com os critérios definidos pelo avaliador da rubrica. Nesse caso foi apresentado o funcionamento de uma rubrica de acordo com as avaliações estabelecidas por Alves e Matos (2017) para classificar os vídeos enviados por estudantes e contribuir no processo de ensino alguns conceitos elementares de uma unidade curricular de Engenharia de Software. Foi definido pelo avaliador a nota 0,6 como nota máxima, 0,3 como nota mediana e 0,1 como nota miníma.

Tabela 4 – Exemplo de uma rubrica avaliada

Itens avaliados	Grupos							Total
	1	2	3	4	5	6	7	
a) Entrega da atividade	0,6	0,6	0,6	0,6	0,6	0,6	0,6	4,2
b) Personagens	0,6	0,6	0,6	0,3	0,3	0,1	0,1	2,6
c) Situação/história	0,6	0,6	0,6	0,6	0,6	0,6	0,1	3,7
d) Conceitos utilizados no enredo	0,6	0,6	0,6	0,6	0,6	0,6	0,1	3,7
e) Apresentação/formatação do vídeo/ recursos audiovisuais	0,6	0,1	0,1	0,1	0,1	0,1	0,1	1,2
Total	3,0	2,5	2,5	2,2	2,2	2,0	1,0	-----

Fonte: Adaptado de Alves e Matos (2017).

Uma outra forma de visualização apresentada na Figura 5 é dada por meio da rubrica completa, em que são citados os critérios de avaliação.

Tabela 5 – Exemplo de uma rubrica com critérios definidos

	Exemplar	Proeficiente	Parcialmente Proeficiente	Insatisfatório
Métrica/Categoria	**3 pontos**	**2 pontos**	**1 ponto**	**0 pontos**
Foco na tarefa e participação	Permanece na tarefa durante todo o tempo sem necessidade de lembretes	Permanece na tarefa a maior parte do tempo. O grupo pode contar com esta pessoa.	Permanece raramente na tarefa. O grupo não pode contar com esta pessoa.	Dificilmente permanece na tarefa. Deixa que os outros façam o trabalho.
Confiança e compartilhamento de responsabilidades	Consistentemente pontual para todos os encontros do grupo, cumpre todo o trabalho no tempo.	Usualmente pontual para todos os encontros do grupo, cumpre quase sempre o trabalho no tempo.	Algumas vezes, se atrasa para os encontros do grupo, frequentemente finaliza o trabalho após o deadline.	Atrasa-se para todos ou a maioria dos encontros do grupo, perde todos os deadlines.
Ouvir, questionar e discutir	Respeitosamente escuta, interage, discute e apresenta questões para todos os membros do grupo durante discussões e ajuda a direcionar o grupo para alcançar um consenso.	Respeitosamente escuta, interage, discute e apresenta questões para todos os membros do grupo durante discussões.	Tem alguma dificuldade para escutar e discutir respeitosamente, e tende a dominar as discussões.	Tem grande dificuldade em escutar, discute com companheiros de equipe, e não está disposto a considerar outras opiniões. Impede grupo de alcançar consenso.
Pesquisa e compartilhamento de informação	Sempre compartilha ideias e informações, agrega valores através de relevantes assuntos fomentando a pesquisa no grupo	Interage compartilhando somente informação	Apenas participa as vezes compartilhando e visualizando informações	Não se interessa por novos assuntos e não fornece informações
Resolução de problemas	Ativamente procura e sugere soluções para os problemas.	Refina soluções sugeridas por outros.	Não sugere ou refina soluções, mas está disposto a experimentar soluções sugeridas por outros	Não tenta resolver problemas ou ajudar os outros a resolver problemas.
Trabalho em equipe/parceria	Contribuiu com os membros da equipe igualmente para o projeto acabado.	Assiste o grupo/parceiro no projeto acabado.	Termina tarefa individual, mas não ajuda o grupo/parceiro durante o projeto.	Pouco contribuíram para o esforço de grupo durante o projeto.
Hábitos de Trabalho	Completa as tarefas atribuídas e não depende de outras pessoas para fazer o trabalho.	Completa a maioria das tarefas atribuídas.	Frequentemente atrasa o trabalho.	Não completa as tarefas no tempo, depende sempre dos outros.

Fonte: Adaptado de Ferraz et al. (2016).

Neste estudo feito por Ferraz et al. (2016) pode-se ver as métricas utilizadas, dando pontuação de 3 pontos para uma avaliação exemplar, 2 pontos para proficiente, 1 ponto para parcialmente proficiente e 0 ponto para insatisfatório. Ele explica os motivos para escolha da pontuação em cada categoria que será avaliada, onde torna-se possível entender a razão de cada pontuação. Quando for realizar a avaliação dos alunos, ele usará essas métricas e categorias

para definir uma pontuação para cada um deles e analisar as suas notas. Esse caso exemplifica bem o uso das rubricas com detalhes sobre suas categorias e pontuações.

Desta forma uma rubrica pode ser desenvolvida e utilizada para definir critérios para que determinado sistema seja considerado responsivo a partir da avaliação desses critérios.

2.5 TRABALHOS CORRELATOS

Em um estudo realizado por Li e Zhang (2019), também com intuito de resolver o problema de compatibilidade das páginas em diferentes tipos de dispositivos, plataformas e tamanhos de telas, basearam o estudo na pesquisa de design responsivo, utilizando utilizando tecnologias relacionadas ao HTML5 e CSS3. Foi feito um site para uma empresa utilizando as técnicas de grid flexível, imagens flexíveis, *meta tag viewport*, entre outras técnicas e se provou um método viável e eficaz.

Em um outro estudo realizado por Dedis (2017), foram analisados os padrões responsivos mais populares na web para o layout de página, navegação, tabela e formulário. Com o uso de tecnologias que fazem o auxílio no desenvolvimento de páginas com design responsivo, como o Less[1], Grunt[2] e Bootstrap[3], essas tecnologias foram utilizadas para fazer um estudo de caso sobre o sistema Tagarelas.

Durante esse estudo foi utilizado o sistema Tagarelas que é um projeto que funciona com objetivo de fornecer informações e canais e bate-papo para ajudar professores a planejar e realizar uma dinâmica educacional para apoiar a conversação e desempenho dos alunos durante

[1] http://lesscss.org/
[2] https://gruntjs.com/
[3] https://getbootstrap.com/

a sessão.

Foi demonstrado a utilização dos padrões responsivos, aplicando técnicas de design responsivo considerados mais adequadas para o portal Tagarelas utilizando como um *framework* de *front-end* o Bootstrap por possuir componentes que facilitam o desenvolvimento.

Em outro estudo realizado por Silva (2014) foi apresentado também um estudo sobre o design responsivo com o objetivo de fazer um site ser visualizado independentemente de qual dispositivo ele esteja sendo visualizado, seja de um tablet, smartphone, notebook, desktop, televisão ou outros dispositivos.

Também foi escolhido pelo autor o portal Tagarelas e ele traz um estudo de caso documentando toda a mudança do layout do portal de modelo de design fixo para modelo responsivo, utilizando as técnicas de design responsivo.

Foi manipulado apenas o aspecto visual da página e diferentemente do estudo de Dedis (2017), não foi utilizado um framework, para que não fosse negligenciada a demonstração das técnicas responsivas.

É aplicado técnicas apresentadas neste trabalho como *media queries*, layout flexível, *breakpoints*, entre outras.

Em outro estudo realizado por Ferraz et al. (2016), nesse caso focado em rubricas para avaliação de alunos de uma disciplina de engenharia de software, auxiliando por meio de aplicações para gestão de informação e comunicação, tendo um ambiente para dar suporte a aprendizagem.

Também auxiliar o professor da disciplina a acompanhar e avaliar seus alunos participantes, com a ferramenta sendo utilizado foi possível concluir o processo de ensino-aprendizagem.

A rubrica foi utilizada selecionando critérios definidos pelo autor para avaliar os alunos participantes e atribuídos pontos para validar a aprendizagem desses alunos de acordo com o comportamento deles para validar ou não seu aprendizado.

3 MATERIAL E MÉTODOS

Neste capítulo é descrita a metodologia utilizada para o desenvolvimento deste projeto. São descritas as etapas do projeto e os principais fundamentos e tecnologias a serem empregados.

3.1 MATERIAL

Nesta seção são abordados os materiais que serão utilizados no desenvolvimento do projeto.

3.1.1 Estudo de caso

O livro Engenharia Web de Pressman e Lowe (2009), trata sobre um dos temas propostos neste trabalho e apresenta um estudo de caso para uma empresa e um sistema experimental que pode ser utilizado como exemplo para realização do estudo de caso pretendido para os sistemas responsivos. Nesse caso foi feito um protótipo sobre um sistema, desenvolvido com base nas aplicações das técnicas responsivas e após sua conclusão, pode ser aplicado a rubrica para avaliar o design e por meio de notas verificar a qualidade do design responsivo desse sistema.

Os passos seguidos no estudo de caso criado do livro, serviram como exemplo para a criação do estudo experimental criado para este trabalho, mesmo sendo um sistema diferente.

3.1.2 Astah UML

O Astah UML[4], da empresa Change Vision, Inc é um software cuja principal função é dar suporte para o desenvolvimento de diagramas na unified modeling language (UML, do inglês, linguagem de modelagem unificada). Ele permite a criação de diagramas utilizados na engenharia de software para ajudar na criação dos diagramas necessários para a realização desse trabalho. Possui uma versão gratuita para estudantes, com as funcionalidades necessárias para o desenvolvimento do trabalho.

3.1.3 Justinmind

O Justinmind Software[5], da empresa Justinmind é uma ferramenta disponível para criar protótipos tanto para desktops quanto para dispositivos móveis, possui uma versão gratuita que foi utilizada para desenvolver o protótipo. É uma ferramenta utilizada para criação de protótipos responsivos, possuindo diversos exemplos em seu site.

3.2 MÉTODOS

Nesta seção são detalhados os métodos utilizados no desenvolvimento do projeto, assim como a sequência dos mesmos. A sequência em que foi feito o trabalho está representada no fluxograma presente na Figura 7, onde é apresentado cada passo que foi seguido.

Inicialmente foi criado um estudo de caso e diagramas para esse trabalho seguindo alguns conceitos do livro Engenharia Web de Pressman e Lowe (2009), onde foram escolhidos os mais propícios para esse trabalho, pois pelo conteúdo total do livro e estudo realizado pelo mesmo, não é possível segui-los totalmente.

O estudo de caso deste trabalho é um site para pessoas que gostam de ter a experiência de comer fora de casa, mas que ao mesmo tempo não desejam chegar ao local e ter que esperar por diversos minutos ou em alguns casos, até mesmo horas para sua refeição chegar até sua

[4] http://astah.net/editions/uml-new
[5] https://www.justinmind.com/

mesa.

Os diagramas escolhidos foram o de classes para representar o usuário, estabelecimento e a reserva, todos com seus respectivos atributos e métodos e também o diagrama de atividade, utilizado para mostrar a atividade de um processo do sistema, para facilitar a visualização e a criação da tela e do sistema.

Posteriormente, foi criado uma página inicial para exemplo da utilização dos métodos de design responsivo também será apresentada, assim como exemplos de protótipos criados a partir dessa página, para facilitar a visualização da aparência final de um sistema responsivo.

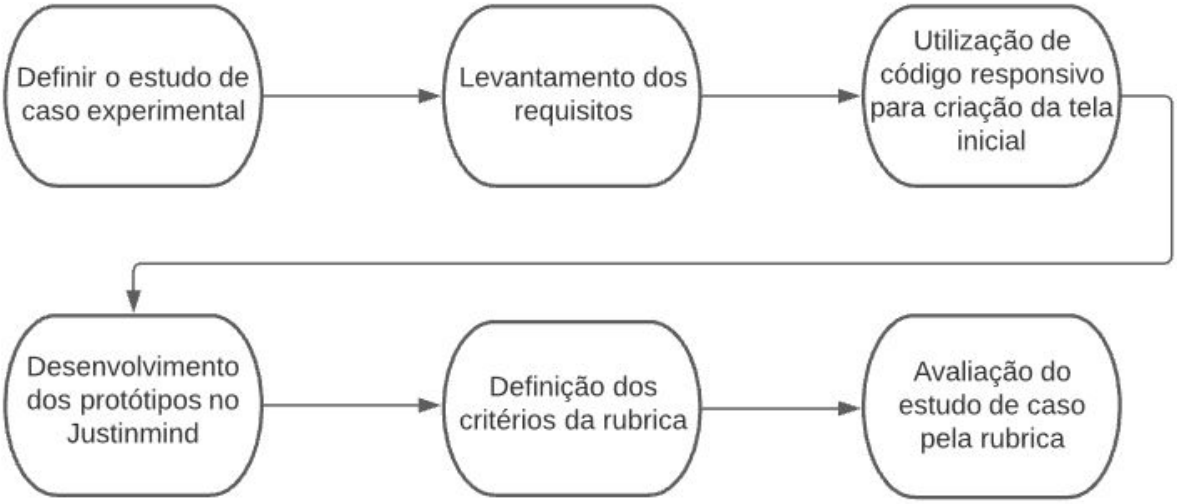

Figura 7 – Fluxograma de sequência de trabalho

Fonte: Autoria Própria.

Os critérios definidos para a rubrica de avaliação do sistema são apresentados na Figura 6, onde a pontuação máxima é definida como 1 ponto, a pontuação média como 0,5 ponto, e a pontuação mínima como 0 ponto. Não existem regras para essa pontuação, portanto a escolha do valor é livre para cada autor.

Para finalizar, são apresentadas as rubricas para avaliar esse estudo de caso criado e as notas referentes a cada propriedade do mesmo.

Tabela 6 – Rubrica com critérios definidos para avaliar o estudo de caso

	Exemplar	Parcialmente satisfatório	Nulo
Métrica/Categoria	**1 ponto**	**0,5 ponto**	**0 ponto**
Utilização da meta tag viewport	O sistema utiliza a meta tag viewport	-	O sistema não utiliza a meta tag viewport
Utilização do método mobile first	O design é iniciado a partir dos dispositivos menores	O design é iniciado a partir dos dispositivos menores, mas depois não segue a ordem de tamanho	O design não segue a sequência do dispositivo menor para o maior
Utilização da unidade de medida flexível "Em" (layout fluído)	A forma de medida de todas as fontes são feitos pela medida flexível Em	Algumas fontes utilizam a forma de medida flexível Em e existem fontes que utilizam outra forma de medida	Não possuem fontes que utilizam a unidade de medida flexível Em
Utilização da unidade de medida flexível "%" (layout fluido)	Todas as formas de medidas utilizadas são "%"	Algumas das formas de medidas utilizadas são em "%", mas possuem outras formas	Não utiliza a forma de medida flexível "%"
Utilização das media queries	Foram utilizadas todas as media queries necessárias	Foram utilizadas apenas algumas media queries	Não foram utilizadas media queries
Breakpoints descobertos	Foram descobertos todos os breakpoints	Foram descobertos apenas alguns breakpoints	Não foi descoberto nenhum breakpoint
Utilização de imagens e mídias flexíveis	As imagens e mídias são flexíveis	Apenas algumas imagens são flexíveis	As imagens não são flexíveis
Utilização de imagens em diferentes tamanhos	Possui todas as imagens necessárias em diferentes tamanhos	Possui algumas imagens necessárias em diferentes tamanhos	Não possui imagens necessárias em diferentes tamanhos
O objetivo principal do sistema é mantido em foco em todas as versões	Todas as versões de tamanho tem suas principais funções em foco	Apenas em alguns dispositivos as funções principais ficam em foco	As funções principais não são facilmente vistas em todos os dispositivos
Foram aplicadas métodos da engenharia de software em sua criação	Foram aplicados todos os métodos necessários da engenharia de software web	Foram aplicados alguns dos métodos necessários da engenharia de software web	Não foram aplicados métodos necessários da engenharia de software web

Fonte: Autoria Própria.

4 RESULTADOS E DISCUSSÃO

Utilizando o material e métodos previamente apresentados, neste capítulo são apresentados e discutidos suas criações e resultados.

4.1 ESTUDO DE CASO DESENVOLVIDO

No estudo de caso experimental desenvolvido para este trabalho, o usuário pode selecionar o restaurante que deseja entre os que estão presentes na lista, escolher sua refeição, logo após, realizar o pagamento, e o restaurante ao receber e aceitar o seu pedido, informar a previsão de horário para a chegada ao local.

O cliente deve poder entrar em contato com o estabelecimento e também conferir em tempo real caso o horário sofra alguma mudança. A página inicial é disponibilizada da seguinte forma:

O usuário tem a opção de digitar a localização, caso não esteja com permissão de acessá-la automaticamente para que os estabelecimentos próximos sejam mostrados.

A versão mobile deve dar preferência e foco principalmente ao menu de busca, pois se o cliente já souber exatamente o que pedir, irá aumentar a eficiência e velocidade do pedido. Logo depois o menu de escolher as categorias de comidas para que o cliente que esteja indeciso possa ver as opções e escolher a partir delas. Por fim as redes sociais do site e também um campo onde, caso o cliente deseje colocar seus dados para receber novidades ou possíveis cupons, ele insere o email e recebe detalhes.

4.2 LEVANTAMENTO DE REQUISITOS

Pensando nas funcionalidades que devem estar presentes tanto na página inicial quanto na sequência de ações, foram levantados os seguintes requisitos:

- Informar a localização/permitir o acesso automático pelo dispositivo;
- Pesquisar o estabelecimento/pedido desejado ou visualizar os tipos de refeições disponíveis;
- Fazer o pedido;
- Escolher método de pagamento;
- Ver horário de previsão/confirmar pedido;
- Falar com restaurante.

4.3 DIAGRAMAS UML

Nem todo projeto é necessário possuir todos os diagramas ou protótipos de telas, pode variar de acordo com a clareza e necessidades específicas do mesmo (PRESSMAN; LOWE, 2009). Nesse caso foram feitos para demonstração os diagramas de classes e atividade, seguindo como exemplo os utilizados no livro Engenharia Web de Pressman e Lowe (2009) para facilitar a visualização do sistema.

Os diagramas apresentados neste trabalho foram pensados para serem simples e mostrar algumas ações que podem acontecer no sistema. O diagrama de classes possui o nome da classe no topo, logo abaixo suas características (conhecidas como atributos) e por fim seus métodos (que representam suas ações ou comportamentos).

É possível ver no diagrama da Figura 8 que o usuário do sistema que é cliente, ele pode alterar sua localização para ver quais restaurantes estão disponíveis, pode criar um novo usuário para fazer reservas caso ainda não seja cadastrado, mudar sua senha caso seja e após logado ele pode fazer sua reserva e também falar com o estabelecimento.

Já o estabelecimento após receber o pedido do cliente, pode confirmar a reserva e caso necessário abrir um chat com o cliente para tirar alguma dúvida ou informar alguma orientação.

A reserva apenas controla o horário e permite que ambos o cliente e o estabelecimento

consigam ver os dados da reserva, como horário e o que foi pedido.

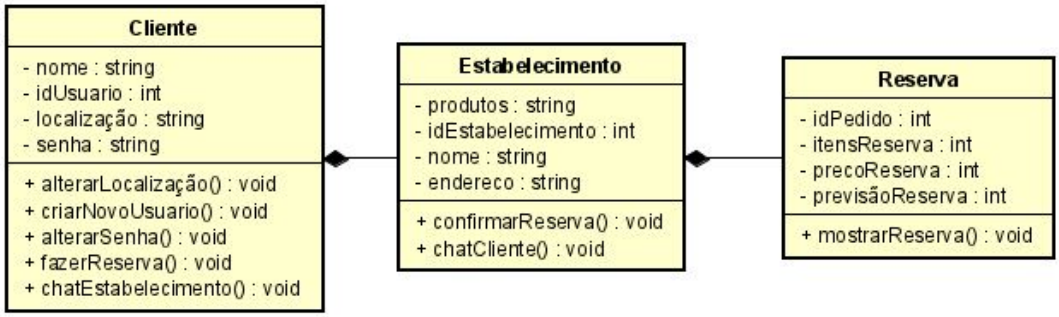

Figura 8 – Diagrama de Classes UML com as classes que representam o estudo de caso

Fonte: Autoria Própria.

O diagrama de atividade é utilizado para mostrar o fluxo das atividades e como ocorre o processo de uma atividade para outra.

Neste caso da Figura 9 mostra o fluxo desde que o usuário abre o sistema até confirmar a reserva, como pode ser observado a seguir:

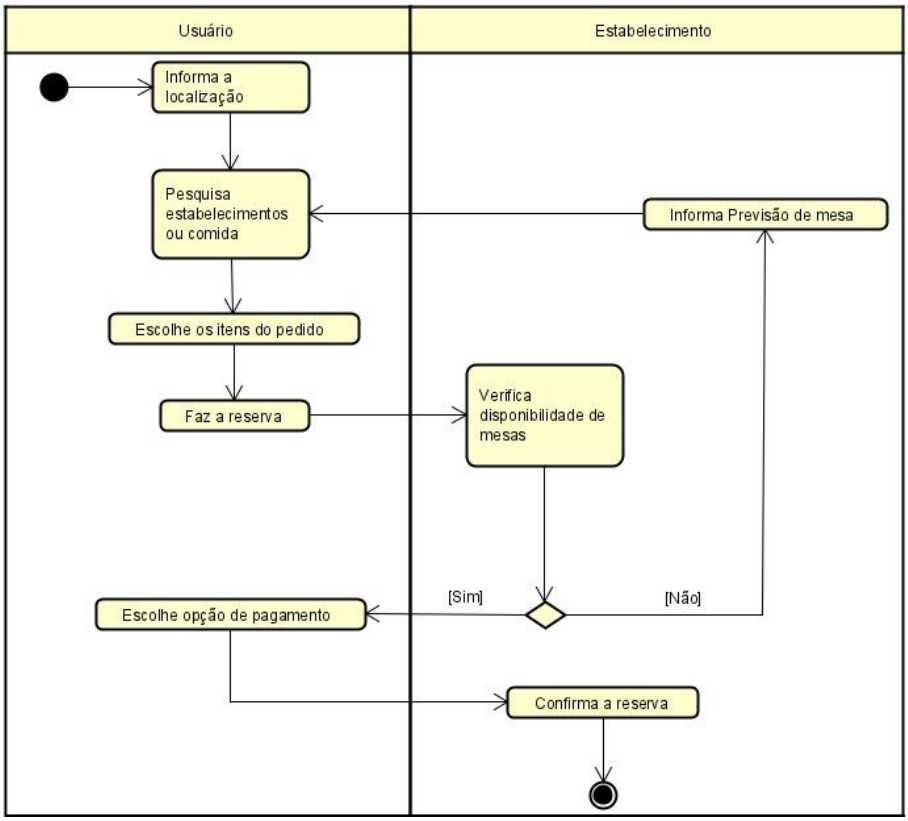

Figura 9 – Diagrama de Atividade para um usuário confirmando sua reserva

Fonte: Autoria Própria.

1. Inicialmente o usuário informa a localização;
2. Na sequência ele faz a pesquisa dos estabelecimentos ou comida que deseja, de acordo com a localização;
3. Ele escolhe os itens de sua reserva;
4. Faz o pedido de reserva;
5. É verificado se o local tem disponibilidade de mesas para a reserva feita;
6. Caso não tenha, o local informa uma previsão e o usuário decide se deseja escolher outro estabelecimento ou aguardar;
7. Caso tenha, o usuário escolhe o método de pagamento;
8. A reserva é confirmada.

4.4 CRIAÇÃO DA TELA INICIAL UTILIZANDO MÉTODOS RESPONSIVOS

Como apresentado na Seção 2.3, existem diversas técnicas que são utilizadas na criação de um sistema responsivo. Nesta seção serão demonstradas a utilização dessas técnicas na criação da tela inicial do estudo de caso.

É possível ver na Figura 10 (a) como pode ser feita a utilização da *meta tag viewport*, na linha 5 do código, logo no começo do documento HTML, de forma simples, apenas inserindo que usará a largura do dispositivo e que a escala se iniciará em 1.

Na Figura 10 (b) pode-se observar a aplicação do método *mobile first*, a partir da linha 64 até a linha 69, onde a configuração se inicia pelos dispositivos menores, onde serão utilizadas as características inseridas ali. No primeiro exemplo, é utilizado a configuração de pequenos dispositivos até smartphones, já na linha 70, é demonstrado de smartphones até tablets, com as mudanças da configuração anterior.

Também é possível observar o layout fluído e mídias flexíveis, pela utilização da forma de medida *"em"* para alterar o tamanho da fonte nas linhas de código 68, 75 e 76, e a forma de medida "%", além de controlar o tamanho das imagens, para que dependendo do tamanho do dispositivo, elas ocupem uma porcentagem da largura, para que possam caber duas imagens por exemplo em uma mesma linha, além da forma de medida "pixels".

Dentro dessa configuração para pequenos dispositivos pode-se ver a utilização das *media queries*, onde ela ajuda a escolher a configuração de layout que melhor atenda o dispositivo em que a página é aberta.

```html
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>Food In Time</title>
        <meta name="description" content="Trabalho de design responsivo para sistemas Web.">
        <meta name="keywords" content="Design Responsivo, Web, Sites">
        <meta name="author" content="Eduardo Pietrantonio">
```

(a) *Código com a meta tag viewport*

```css
/* Método Mobile First */
/* Pequenos dispositivos - Smartphones */
@media screen and (min-width: 480px) {
.logo {width: 214px; background: url(../img/logo1.png) left center/70px no-repeat;}
.btn {font-size: 2em;}
}
/* Smartphones - Tablets */
@media screen and (min-width: 768px) {
.logo {width: 450px; height: 100px;background: url(../img/logo2.png) left center/280px no-repeat;
.servico {width: 49%; float: left; margin-right: 1%;}
.servico:nth-child(2){margin-right: 0;}
.newsletter h2 {font-size: 2em;}
.newsletter h3 {font-size: 1.5em;}
.newsletter input {width: 70%; padding: 2%; float: left; margin-right: 1%;}
.newsletter button {width: 29%; padding: 2%; float: right; margin-top: 0;}
}
```

(b) *Código utilizando Mobile first*/Layout fluído/*Media queries/Mídias flexíveis*

Figura 10 – Código Responsivo

Fonte: Autoria Própria.

Na Figura 11 é possível observar o comportamento do design em uma tela pequena de tamanho 375x667, onde uma característica que pode ser observada é a logo, que é uma versão sem o nome do site, apenas a imagem. Também há a utilização do menu Hambúrguer, que é comumente utilizado para dispositivos móveis. Também há foco nas principais funções, como deixar o menu de Busca e Localização na parte superior, para serem vistos prontamente por quem acessa ao site, dando segundo foco aos tipos de comidas disponíveis.

Na Figura 12, algumas mudanças já podem ser notadas em relação a Figura 11, as principais são, a logo, onde nesse caso, é utilizado uma logo maior pois há mais espaço para preencher, também com um texto correspondente ao nome do sistema.

Outra mudança é o menu, onde todas opções são visualizadas na página de navegação no topo da tela, também pode ser visto que o local de divulgação e os botões de busca de pedido e localização ficam na mesma linha.

Por fim, o número de opções de comidas conseguem aparecer lado a lado na linha, não apenas aumentando o tamanho da imagem e que cada uma delas ocupe uma linha na página.

Isso tudo é possível através dos métodos *mobile first*, *media queries*, layout fluído, imagens flexíveis, controlando o comportamento da página através desses métodos.

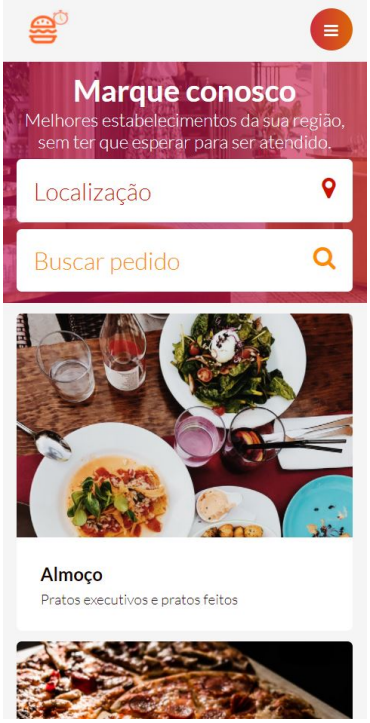

Figura 11 – Tela responsiva de 375x667

Fonte: Autoria Própria.

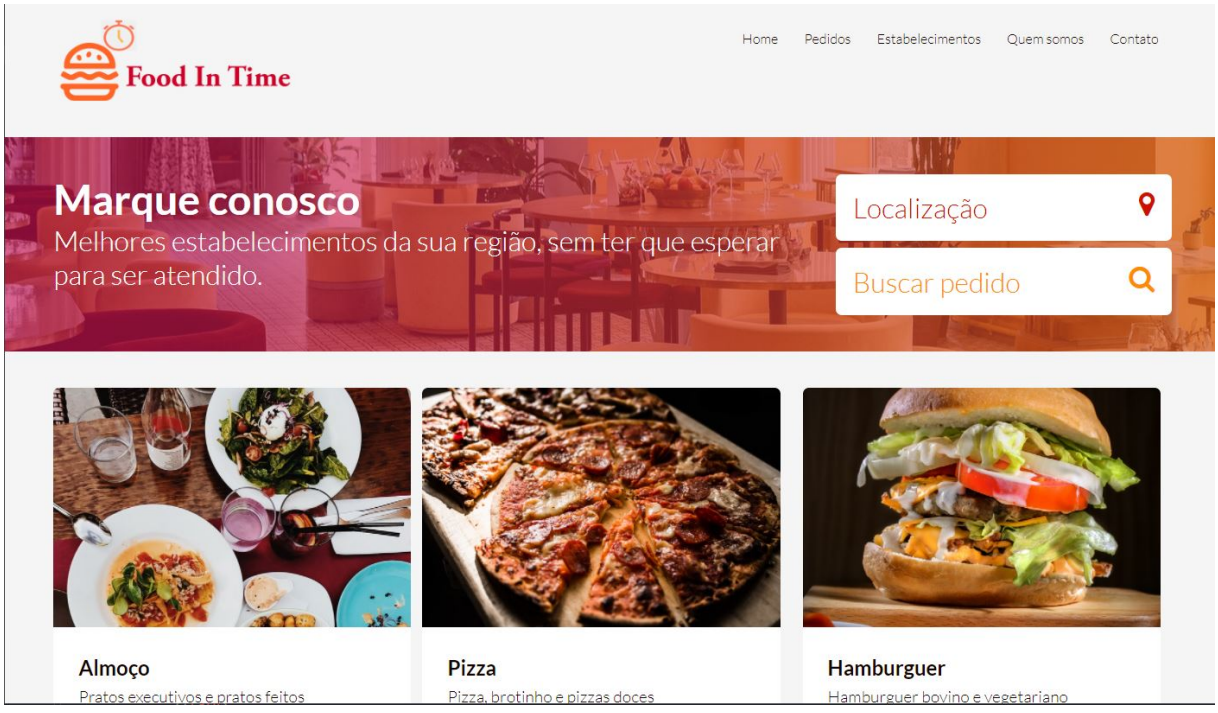

Figura 12 – Tela responsiva de 1440x900

Fonte: Autoria Própria.

4.5 DESENVOLVIMENTO DE PROTÓTIPOS DE TELAS

O desenvolvimento dos protótipos foi feito utilizando-se o software Justinmind. Como apresentado na seção 3.1, por meio deste aplicativo, é possível escolher o tipo de protótipo a ser utilizado. Para o exemplo a ser utilizado neste estudo de caso, apresentado nesta seção, foi escolhido o tipo ”Web”, sendo que os tamanhos de tela escolhidos foram de 375x667 e 1440x900, como apresentado anteriormente na seção 4.4.

Na Figura 13 é possível observar que a tela criada é de um estabelecimento selecionado, nomeado como Lanchonete EGP, aparecem alguns filtros abaixo, para organizar os pedidos e logo após, aparece o cardápio da lanchonete, com as opções de comidas.

As opções são listadas com uma imagem, que varia de tamanho de acordo com o tamanho do dispositivo, como é possível visualizar na Figura 14, seguido pelas outras informações da opção, como nome, detalhes, tempo de espera para que fique pronto e também seu preço.

Figura 13 – Protótipo Justinmind tela de 375x667

Fonte: Autoria Própria.

Na Figura 14, inicialmente já é possível observar algumas diferenças em relação ao design da Figura 13, como por exemplo a logo e o menu como já foi mostrado anteriormente nas Figuras 11 e 12.

Novas mudanças que podem ser notadas ficam na parte dos itens do menu, mantendo alguns detalhes iguais, como o nome da lanchonete e os filtros, apesar de estarem centralizados, mas também mais destaque para as imagens dos pedidos, com uma tamanho maior, tendo ainda mais destaque para os itens em promoção.

Isso faz com que o foco principal do sistema que é a reserva de estabelecimentos, continue em evidência, incentivando o cliente a fazer os pedidos, vendo as fotos em destaque e as promoções dos estabelecimentos.

Logo abaixo é possível observar os itens normais do menu, que não estão em promoção, mas que ainda assim possuem um foco grande através de suas imagens maiores e a barra de busca de pedidos no cardápio.

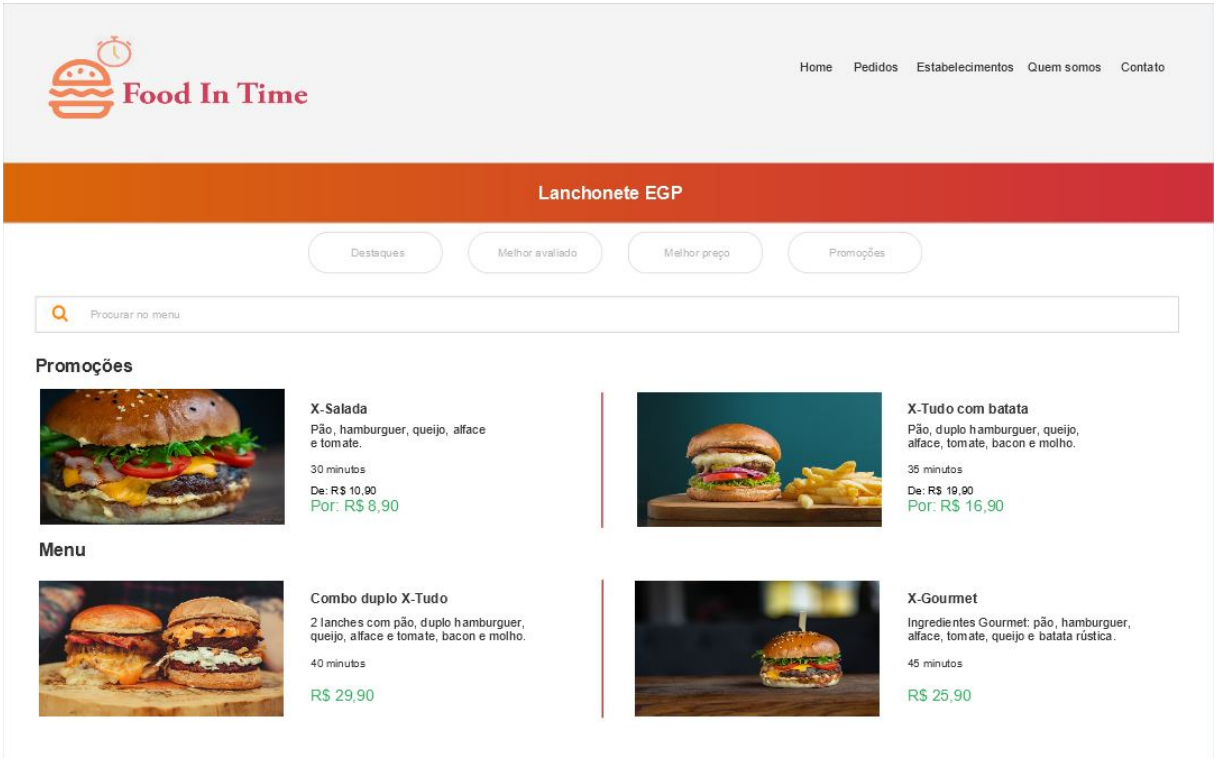

Figura 14 – Protótipo Justinmind tela de 1440x900

Fonte: Autoria Própria.

As Figuras 11, 12, 13 e 14 servirão como base para serem avaliadas pela rubrica e seus critérios, sendo observados seus pontos positivos e negativos e o sistema receberá uma pontuação de acordo com essa avaliação.

4.6 UTILIZAÇÃO DA RUBRICA

Na Figura 7 é possível visualizar a pontuação para cada um dos critérios definidos, de acordo com o que foi feito no sistema.

Tabela 7 – Rubrica de avaliação do sistema

Categoria	Pontuação
Utilização da meta tag viewport	1 ponto
Utilização do método mobile first	1 ponto
Utilização da unidade de medida flexível "Em" (layout fluído)	1 ponto
Utilização da unidade de medida flexível "%" (layout fluido)	0,5 ponto
Utilização das media queries	1 ponto
Breakpoints descobertos	1 ponto
Utilização de imagens e mídias flexíveis	1 ponto
Utilização de imagens em diferentes tamanhos	0,5 ponto
O objetivo principal do sistema é mantido em foco em todas as versões	1 ponto
Foram aplicadas métodos da engenharia de software em sua criação	1 ponto
Pontuação Total:	9 pontos

Fonte: Autoria Própria.

As explicações para cada pontuação são apresentadas a seguir:

- Utilização da *meta tag viewport*: como observado na Figura 10 (a) que mostra o código com a utilização da *meta tag viewport*, o sistema utiliza essa técnica, portanto recebe 1 ponto;
- Utilização do método *mobile first*: como observado na Figura 10 (b) se inicia utilizando o método *mobile first* e segue para telas maiores gradativamente, portanto recebe 1 ponto;
- Utilização da unidade de medida flexível "Em" (layout fluído): nas configurações das fontes, é utilizada a unidade de medida Em, já que seu principal uso são para fontes, portanto recebe 1 ponto;

- Utilização da unidade de medida flexível "%" (layout fluido): apesar de ser utilizado em grande parte do código, em alguns locais ainda são utilizadas as unidades de medidas em pixels, portanto recebe 0,5 ponto;
- Utilização das *media queries*: são criadas e utilizadas diferentes media queries, para diversos tamanhos de dispositivos, portanto cobrindo uma ampla parcela, desde dispositivos menores, até grandes telas de desktop, portanto recebe 1 ponto;
- *Breakpoints* descobertos: para serem criados as media queries, foram descobertos diferentes *breakpoints*, para cobrir o maior número possível de quebras no design e que adaptassem a essas quebras, portanto recebe 1 ponto;
- Utilização de imagens e mídias flexíveis: as imagens alteram o tamanho de acordo com o dispositivo, sendo imagens maiores para dispositivos maiores e menores em dispositivos menores, portanto recebe 1 ponto;
- Utilização de imagens em diferentes tamanhos: não foram utilizadas diferentes versões das imagens dos pedidos, pois nesse caso não se tornou necessário, como as imagens tinham uma boa qualidade, seu aumento não prejudicava a visualização no site, sendo utilizados imagens diferentes apenas para a logo para dispositivos menores e a logo para dispositivos maiores;
- O objetivo principal do sistema é mantido em foco em todas as versões: no caso desse sistema, o foco é no pedido, ou seja, tem que ser fácil para o cliente conseguir buscar 1 pedido e ver as opções de comidas e estabelecimentos, portanto tanto na versão para dispositivos menores, como para os maiores, sempre ficava em primeiro plano, essas funcionalidades, facilitando assim a visualização do cliente para finalizar seu pedido, portanto recebe 1 ponto;
- Foram aplicadas métodos da engenharia de software em sua criação: apesar de nem sempre ser necessário utilizar diversos diagramas, dependendo do sistema e sua criação, sempre facilita futuras mudanças ou criação de um sistema, foram escolhidas algumas técnicas para serem utilizadas, como o levantamento de requisitos, diagramas de classes, sequência e atividade, entre outras coisas para auxiliar nessa criação, portanto recebe 1 ponto.

A pontuação final da avaliação feita utilizando a rubrica, foi de 9 pontos, de 10 possíveis, sendo assim uma boa avaliação, apesar de haver detalhes a melhorar, onde todo sistema é possível ter melhorias, apesar de qualquer método utilizado na criação. O foco das rubricas é facilitar a visualização dessas falhas, para que possam ser corrigidas mais facilmente.

5 CONSIDERAÇÕES FINAIS

Neste capítulo são apresentadas as considerações finais deste trabalho. É discutido se os objetivos foram cumpridos, possíveis aplicações para este trabalho e também recomendações para trabalhos futuros a partir deste.

5.1 CONCLUSÕES

O trabalho visou utilizar rubricas para avaliar e fazer a validação de um sistema, verificando se o mesmo trabalha de forma responsiva e utilizando na sua criação, métodos da engenharia de software web.

Ao criar o estudo de caso experimental, a engenharia de software web, se mostrou muito útil, pois ao levantar os requisitos e criar os diagramas, facilitou a visualização do funcionamento do sistema, fazendo assim com que a criação de suas telas fosse facilitada, com isso é possível concluir que a engenharia web pode auxiliar bastante na criação do sistema responsivo.

A partir disso as técnicas responsivas foram aplicadas de maneira mais fácil, já que havia o conhecimento das funcionalidades do sistema, ou seja, a criação do design iniciado das telas menores, até as telas maiores foi aplicado de maneira otimizada.

Os protótipos de telas também auxiliaram nesse processo, pois ao ter mais telas para visualização, ficou mais fácil definir os critérios da rubrica, provando trabalhar em comum acordo com a utilização das técnicas responsivas apresentadas.

A definição dos critérios se tornou mais fácil graças aos processos feitos, como o levantamento de requisitos, os diagramas, e as telas apresentadas tanto na versão menor, como também na versão maior, pois com as mudanças vistas nas telas, foi possível ver as técnicas responsivas em ação.

Com isso foi possível concluir que as rubricas podem ser muito úteis para avaliação

do design responsivo, tanto a partir de suas telas, quanto a partir do seu código, pois com elas, você consegue verificar o que está de acordo com o design responsivo e também o que não está, não apenas falando o que há de errado, mas também auxiliando na visualização do que pode ser alterado, facilitando assim a correção dos problemas.

5.2 TRABALHOS FUTUROS

No estudo de caso utilizado neste documento foi feito a avaliação a partir de um sistema para pedidos de reserva em restaurantes que é um sistema experimental criado para este trabalho, então a rubrica além de avaliar as técnicas responsivas cujo é o foco principal, avalia também esse tema em específico. A reutilização pode ser feita também para outros sistemas, até mesmo para sites de empresas já existentes, como sites de notícias, uma loja de roupas, redes sociais, onde a avaliação das técnicas responsivas nesses sistemas são mantidas, mas podem ser adicionados novos critérios de acordo com suas necessidades de avaliação.

REFERÊNCIAS

ALVES, C. F.; MATOS, M. E. de. Sequência didática para conteúdo de engenharia de software. **Revista Brasileira de Ensino de Ciência e Tecnologia**, v. 10, n. 3, p. 266–279, 2017. ISSN 1982-873X.

APRIL, A.; KAJKO-MATTSSON, M. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

BANDYOPADHAYAY, A.; WILLSHIRE, M. J. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

BERTOLINO, A.; MARCHETTI, E. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

BIAGIOTTI, L. C. M. C. M. Conhecendo e aplicando rubricas em avaliações. **Congresso Brasileiro de Educação à Distância**, p. 9, 2005. Disponível em: <http://www.abed.org.br/congresso2005/por/pdf/007tcf5.pdf>.

BLAINE, J. D.; BISWAS, D. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

BOURQUE, P.; FAIRLEY, R. E. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE Computer Society, 2014. Disponível em: <www.swebok.org.>.

BROD, C. A. d. A.; KÄFER, J. Engenharia de Software para Software Livre. 2009. Disponível em: <http://www.softwarepublico.gov.br/file/16734767/Engenharia_de_Software_>.

CHAKI, N. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

CHAMPAGNE, R.; APRIL, A. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

DEDIS, L. S. A. Padrões de design responsivo. 2017.

EBERT, C. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

FERRAZ, J.; DUARTE, D.; ESMERALDO, G. Uma ferramenta para suporte à avaliação de participantes de fábricas de software em disciplinas de engenharia de software. **Revista Acta Kariri**, v. 1, n. 1, p. 68–75, 2016.

FRANÇA, S. D. S. Web Design Responsivo: Caminhos Para Um Site Adaptável. **Interfaces Científicas - Exatas e Tecnológicas**, v. 1, n. 2, p. 75, 2015. ISSN 2359-4934.

GIURGIU, L.; GLIGOREA, I. Responsive Web Design Techniques. **International conference KNOWLEDGE-BASED ORGANIZATION**, v. 23, n. 3, p. 37–42, 2017. ISSN 1843-6722.

GSM Association. Mobile economy. **Gsma**, p. 2–62, 2020. Disponível em: <https://www.gsma.com/>.

KOTONYA, G.; SAWYER, P. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

LI, N.; ZHANG, B. The design and implementation of responsive web page based on HTML5 and CSS3. **Proceedings - 2019 International Conference on Machine Learning, Big Data and Business Intelligence, MLBDBI 2019**, IEEE, p. 373–376, 2019.

LOPES, S. **A Web Mobile: Programe para um mundo de muitos dispositivos**. São Paulo: Casa do Código, 2013. 223 p.

MARCOTTE, E. Responsive Web Design. A Book Apart, 2010.

MCDONALD, J. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

MEIRELLES, F. S. Pesquisa. 2019.

MOYEEN, M. A. et al. An automatic layout faults detection technique in responsive web pages considering JavaScript defined dynamic layouts. **2016 3rd International Conference on Electrical Engineering and Information and Communication Technology, iCEEiCT 2016**, IEEE, p. 1–5, 2017.

PENG, X. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

PRESSMAN, R. S. **Engenharia de Software: A Practitioner's Approach**. New York: McGraw Hill, 2005.

PRESSMAN, R. S.; LOWE, D. **Engenharia Web**. Rio de Janeiro: LTC, 2009.

PROSTT, M. E. Interface Web Utilizando Design Responsivo : Um Estudo De Caso Aplicado a Smartphones , Tablets , Computadores E Televisores Interface Web Utilizando Design Responsivo : Um Estudo De Caso Aplicado a Smartphones , Tablets ,. 2013.

REILLY, A.; FAIRLEY, R. E. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

RINE, D. **Web Engineering Advancements and Trends : Building New Dimensions**. [S.l.: s.n.], 2010. ISBN 9781605667195.

SHEFFIELD, A.; ZOU, H. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

SILVA, A. d. A. P. D. Design Responsivo: Técnicas, Frameworks E Ferramentas Arthur. 2014.

SIOK, M. F. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

SUN, Y. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.

TOMAZINI, M. et al. Web design responsivo - Bootstrap Introdução. 2018.

WARD, C. **Responsive Web Design Modern Responsive Solutions**. Austrália: SitePoint, 2017. ISBN 978-0-9953827-2-5.

WROBLEWSKI, L. MOBILE. n. October, 2011.

ZEMEL, T. **Web Design Responsivo: Páginas Adaptáveis para todos os dispositivos**. São Paulo: Casa do Código, 2012. 143 p.

ZOU, H. **Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)**. New Jersey: IEEE, 2014.