**FAI – CENTRO DE ENSINO SUPERIOR EM GESTÃO, TECNOLOGIA E EDUCAÇÃO**

**CURSO DE SISTEMAS DE INFORMAÇÃO**

**ALEXANDRE DA SILVA RIBEIRO**

**GABRIEL RIBEIRO GARCIA**

**LUAN REIS DE CARVALHO**

**EASYJOB – DESENVOLVIMENTO DE UM SISTEMA DE SOFTWARE PARA OFERTA E PROCURA DE TRABALHOS INTERMITENTES**

**SANTA RITA DO SAPUCAÍ - MG**

**2020**

**FAI – CENTRO DE ENSINO SUPERIOR EM GESTÃO, TECNOLOGIA E EDUCAÇÃO**

**CURSO DE SISTEMAS DE INFORMAÇÃO**

**ALEXANDRE DA SILVA RIBEIRO**

**GABRIEL RIBEIRO GARCIA**

**LUAN REIS DE CARVALHO**

**EASYJOB – DESENVOLVIMENTO DE UM SISTEMA DE SOFTWARE PARA OFERTA E PROCURA DE TRABALHOS INTERMITENTES**

**Projeto de final de curso apresentado à FAI – Centro de Ensino Superior em Gestão, Tecnologia e Educação, como requisito parcial para obtenção do Título de Bacharel em Sistemas de Informação, sob a orientação da professora Ma. Eunice Gomes de Siqueira.**

**SANTA RITA DO SAPUCAÍ - MG**

**2020**

**FOLHA DE APROVAÇÃO**

**HISTÓRICO DE REVISÕES**

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versão | Autor | Descrição |
| 19/11/2020 | 2.14 | Alexandre da Silva Ribeiro | Correções da Fase3, criação do diagrama de componentes. |
| 14/11/2020 | 2.13 | Luan Reis De Carvalho | Correções dos capítulos 6,7,8 |
| 19/10/2020 | 2.12 | Eunice G. Siqueira | Apontamentos da Fase 3 (ainda incompleta). |
| 10/10/2020 | 2.11 | Luan Reis De Carvalho | Correções dos Capítulos 5,6,7,8 e 9. |
| 10/09/2020 | 2.10 | Eunice G. Siqueira | Apontamentos da Fase 3. |
| 04/09/2020 | 2.9 | Luan Reis de Carvalho | Correção do diagrama de atividades, diagrama de classes e diagrama de pacotes. |
| 04/09/2020 | 2.8 | Alexandre da Silva Ribeiro | Elaboração do diagrama de sistema distribuído. |
| 04/09/2020 | 2.7 | Alexandre da Silva Ribeiro | Elaboração dos Capítulos 7 e 8. |
| 03/09/2020 | 2.6 | Alexandre da Silva Ribeiro  Luan Reis De Carvalho | Elaboração do Capítulo 6. |
| 28/06/2020 | 2.5 | Luan Reis De Carvalho  Gabriel Ribeiro Garcia  Alexandre da Silva Ribeiro | Correção dos capítulos 3,4,5 e 6 |
| 28/06/2020 | 2.4 | Luan Reis De Carvalho  Gabriel Ribeiro Garcia  Alexandre da Silva Ribeiro | Revisão da segunda entrega. |
| 06/06/2020 | 2.3 | Eunice G. Siqueira | Indicação das correções da banca de qualificação. |
| 30/05/2020 | 2.2 | Luan Reis De Carvalho  Gabriel Ribeiro Garcia | Correção dos capítulos 4,5, 6 e 7. |
| 25/05/2020 | 2.1 | Eunice G. Siqueira | Indicação de correções da Fase 2. |
| 23/05/2020 | 2.0 | Alexandre da Silva Ribeiro  Luan Reis De Carvalho | Criação do diagrama de dados lógico, do dicionário do modelo lógico, diagrama de atividades, diagrama de classes, diagrama de pacotes e preenchimento da estimativa por casos de uso. |
| 22/05/2020 | 1.9 | Gabriel Ribeiro Garcia  Luan Reis de Carvalho | Correção dos Capítulos 3, 4 e 5. |
| 22/05/2020 | 1.8 | Luan Reis de Carvalho | Elaboração do Capítulo 6. |
| 18/05/2020 | 1.7 | Alexandre da Silva Ribeiro | Correção do modelo entidade relacionamento, Correção dos casos de usos e criação do diagrama de classes. |
| 03/05/2020 | 1.6 | Eunice G. Siqueira | Correção parcial da Fase 1. |
| 24/04/2020 | 1.5 | Luan Reis de Carvalho | Correção do Capítulo 5. |
| 20/04/2020 | 1.4 | Gabriel Ribeiro Garcia | Correção dos Capítulos 3 e 4. |
| 11/04/2020 | 1.3 | Alexandre da Silva Ribeiro | Correção dos Capítulos 1 e 2, da lista de figuras e quadros automáticos. |
| 31/03/2020 | 1.2 | Eunice G. Siqueira | Correção parcial da Fase 1. |
| 28/03/2020 | 1.1 | Alexandre da Silva Ribeiro Gabriel Ribeiro Garcia Luan Reis de Carvalho | Revisão da primeira entrega. |
| 28/03/2020 | 1.0 | Alexandre da Silva Ribeiro Gabriel Ribeiro Garcia Luan Reis de Carvalho | Primeira entrega. |
| 27/03/2020 | 0.3 | Alexandre da Silva Ribeiro Gabriel Ribeiro Garcia | Formatação do documento com base nas Diretrizes de trabalhos acadêmicos da FAI. |
| 24/03/2020 | 0.2 | Alexandre da Silva Ribeiro | Elaboração da documentação: Capítulos de 1 e 2. |
| 24/03/2020 | 0.3 | Gabriel Ribeiro Garcia Luan Reis de Carvalho | Elaboração da documentação: Capítulos de 3 ao 5. |
| 17/03/2020 | 0.1 | Alexandre da Silva Ribeiro | Criação do documento base. |

**AGRADECIMENTOS**

**RESUMO**

O trabalho intermitente, que é aquele realizado de forma não contínua, tornou-se para algumas pessoas uma oportunidade para obterem uma renda extra e, para outras, um meio de sobrevivência. Com o aumento do uso das tecnologias de informação, surgiram várias ferramentas que facilitam a oferta e procura de trabalhos intermitentes e o *EasyJob*, projeto apresentado neste documento, destina-se ao desenvolvimento de um sistema de software voltado para esse setor. Com foco no trabalho intermitente realizado em eventos sociais, o *EasyJob* é um sistema que possibilitará aos estabelecimentos ofertarem vagas para trabalho temporário, assim como permitirá que às pessoas interessadas se candidatem a elas. A metodologia adotada é neste projeto é a pesquisa bibliográfica, gerência com referência ao Guia de Conhecimentos do *Project Management Institute,* especificação e modelagem baseada na *Unified Modeling Language* e construção de software orientado a objetos. Como resultado do projeto, obteve-se um sistema de software disponível para uso da *Web.*

**Palavras-chave**: Trabalho intermitente. *Freelancer*. Organização de eventos. Projeto Final de Curso. Sistemas de Informação.

**LISTA DE FIGURAS**

[FIGURA 1 - Divisão do Mercado de Trabalho no 4° trimestre de 2019. 18](#_Toc56704038)

[FIGURA 2 - Taxa de Desocupação no Brasil e nas grandes regiões no 4° trimestre de 2019. 19](#_Toc56704039)

[FIGURA 3- Taxa de desocupação no Brasil na pandemia. 20](#_Toc56704040)

[FIGURA 4- Crescimento do número de vagas informais no Brasil entre 2017 e 2018. 21](#_Toc56704041)

[FIGURA 5- Telas do aplicativo GetNinjas 24](#_Toc56704042)

[FIGURA 6- Telas do aplicativo Mystaff. 25](#_Toc56704043)

[FIGURA 7- Modelo de ciclo de vida incremental. 31](#_Toc56704044)

[FIGURA 8 - Repositório de versionamento de código Github. 37](#_Toc56704045)

[FIGURA 9 - Tela inicial do *EasyJob* 54](#_Toc56704046)

[FIGURA 10 - Tela de inscrição do *EasyJob*. 54](#_Toc56704047)

**LISTA DE QUADROS**

[QUADRO 1 - Cronograma de entregas do projeto 32](#_Toc56704064)

[QUADRO 2 - Recursos Humanos 33](#_Toc56704065)

[QUADRO 3 - Ferramentas de software. 33](#_Toc56704066)

[QUADRO 4 - Estimativas de esforço 35](#_Toc56704067)

[QUADRO 5 - Documento relevante para testes 63](#_Toc56704068)

[QUADRO 6 - Equipamentos para a realização dos testes. 64](#_Toc56704069)

[QUADRO 7 - Softwares para a realização dos testes 64](#_Toc56704070)

[QUADRO 8 - Identificação dos itens a serem testados 65](#_Toc56704071)

**LISTA DE ABREVIATURAS E SIGLAS**

ANVISA –Agência Nacional de Vigilância Sanitária.

API – *Application Programming Interface.*

[CETESB](http://www.cetesb.sp.gov.br/) – Companhia Ambiental do Estado de São Paulo.

CLT *–* Consolidação das Leis Trabalhistas.

CPF  *–* Cadastro de Pessoa Física.

CSS *– Cascading Style Sheets.*

DAO *– Data Access Object*

FGTS *–* Fundo de Garantia do Tempo de Serviço.

GB *– Gigabytes.*

HTML *– HyperText Markup Language.*

HTTPS *– HyperText Transfer Protocol Secure.*

IBGE *–* Instituto Brasileiro de Geografia e Estatística.

IOS *–* Sistema operacional móvel da *Apple Inc*.

MIT  *–*  *Massachusetts Institute of Technology.*

MVC  *– Model-View-Controller.*

ORM *– Object-relational mapping*

PMI *–*   *Project Management Institute*

RAM *– Random Access Memory.*

REST *– Representation State Transfer.*

RF  *–* Requisitos Funcionais.

RNF *–* Requisitos Não Funcionais.

SEVISA  *–* Sistema Estadual de Vigilância Sanitária.

SGDB *–* Sistema de Gerenciamento de Banco de Dados.

SMTP  *– Simple Mail Transfer Protocol.*

SQL *– Structured Query Language.*

UML – *Unified Modeling Language.*

**SUMÁRIO**

[1 INTRODUÇÃO 16](#_Toc56702405)

[2 REVISÃO BIBLIOGRÁFICA 18](#_Toc56702406)

[2.1 DESEMPREGO E TRABALHO INFORMAL 18](#_Toc56702407)

[2.2 TRABALHO INTERMITENTE 22](#_Toc56702408)

[2.3 TRABALHOS RELACIONADOS 23](#_Toc56702409)

[2.3.1 *GetNinjas* 23](#_Toc56702410)

[2.3.2 *MyStaff* 24](#_Toc56702411)

[3 OBJETIVO DO PROJETO 26](#_Toc56702412)

[3.1 FORMULAÇÃO DO PROBLEMA 26](#_Toc56702413)

[3.2 OBJETIVOS 26](#_Toc56702414)

[3.3 JUSTIFICATIVA 27](#_Toc56702415)

[3.3.1 Importância da regulamentação 27](#_Toc56702416)

[3.3.2 Exigências legais para *buffets* funcionarem 27](#_Toc56702417)

[3.4 PÚBLICO ALVO 28](#_Toc56702418)

[3.5 NÍVEIS DE DECISÃO E GRUPOS FUNCIONAIS ATENDIDOS 28](#_Toc56702419)

[3.6 ADERÊNCIA AOS OBJETIVOS DE DESENVOLVIMENTOS SUSTENTÁVEL 28](#_Toc56702420)

[4 GERÊNCIA DO PROJETO 30](#_Toc56702421)

[4.1 PLANO DE ELABORAÇÃO E GERENCIAMENTO DO PROJETO 30](#_Toc56702422)

[4.1.1 Partes Interessadas 30](#_Toc56702423)

[4.2 MODELO DE CICLO DE VIDA 31](#_Toc56702424)

[4.3 RECURSOS NECESSÁRIOS 32](#_Toc56702425)

[**4.3.1 Recursos Humanos** 32](#_Toc56702426)

[**4.3.2 Recursos de Hardware** 33](#_Toc56702427)

[**4.3.3 Recursos de Software** 33](#_Toc56702428)

[4.4 RELATÓRIO DE DESEMPENHO 34](#_Toc56702429)

[5 ESPECIFICAÇÃO E ANÁLISE DOS REQUISITOS DO SISTEMA 38](#_Toc56702430)

[5.1 DESCRIÇÃO DE REQUISITOS 38](#_Toc56702431)

[5.1.1 Requisitos Funcionais 38](#_Toc56702432)

[5.1.2 Requisitos Não Funcionais 45](#_Toc56702433)

[5.1.2.1 Requisitos de Produto 46](#_Toc56702434)

[5.1.2.2 Requisitos Organizacionais 46](#_Toc56702435)

[5.1.2.3 Requisitos Externos 47](#_Toc56702436)

[5.2 ANÁLISE DOS REQUISITOS 49](#_Toc56702437)

[5.2.1 Diagrama de Casos de Uso 49](#_Toc56702438)

[5.2.2 Descrição dos Casos de Uso 49](#_Toc56702439)

[5.2.3 Modelo Conceitual dos Dados 49](#_Toc56702440)

[**5.2.4 Modelo Inicial da Interface de Usuário** 49](#_Toc56702441)

[6 ARQUITETURA E PROJETO DO SISTEMA DE SOFTWARE 51](#_Toc56702442)

[6.1 VISÃO ESTRUTURAL 51](#_Toc56702443)

[6.1.1 Diagrama de Pacotes 51](#_Toc56702444)

[6.1.2 Diagramas de Classes 51](#_Toc56702445)

[6.2 VISÃO COMPORTAMENTAL 51](#_Toc56702446)

[6.2.1 Diagrama de Sequência 52](#_Toc56702447)

[6.2.2 Diagrama de Atividades 52](#_Toc56702448)

[6.3 VISÃO DE DADOS 52](#_Toc56702449)

[6.3.1 Modelo Lógico 52](#_Toc56702450)

[6.3.2 Dicionário de Dados do Modelo Lógico 52](#_Toc56702451)

[6.4 PROJETO DA INTERAÇÃO HUMANO-COMPUTADOR 52](#_Toc56702452)

[6.4.1 Perfil do Usuário 53](#_Toc56702453)

[6.4.2 Projeto da Interface de Usuário 54](#_Toc56702454)

[6.4.4 Acessibilidade 55](#_Toc56702455)

[6.5.1 Procedimentos para Tratamento dos Desafios 55](#_Toc56702456)

[6.5.1.1 Heterogeneidade 56](#_Toc56702457)

[6.5.1.2 Escalabilidade 56](#_Toc56702458)

[6.5.1.3 Abertura 56](#_Toc56702459)

[6.5.1.4 Segurança 57](#_Toc56702460)

[6.6.1.5 Manuseio de falhas 57](#_Toc56702461)

[6.5.1.6 Concorrência 57](#_Toc56702462)

[6.5.1.7 Transparência 58](#_Toc56702463)

[6.5.2 Tecnologias e Arquiteturas de Distribuição 58](#_Toc56702464)

[7 IMPLEMENTAÇÃO DO SISTEMA DE SOFTWARE 59](#_Toc56702465)

[7.1 COMPONENTES DO SISTEMA DE SOFTWARE 59](#_Toc56702466)

[7.2 TECNOLOGIAS DE IMPLANTAÇÃO 59](#_Toc56702467)

[7.2.1 Linguagens de Programação e *Frameworks* Adotados 59](#_Toc56702468)

[7.2.2 *Design Patterns* Aplicados 60](#_Toc56702469)

[7.2.3 Convenções e Guias para Codificação 61](#_Toc56702470)

[7.2.4 Estrutura Física do Banco de Dados 62](#_Toc56702471)

[8 PLANO DE TESTES 63](#_Toc56702472)

[8.1 FINALIDADE 63](#_Toc56702473)

[8.2 ESCOPO 63](#_Toc56702474)

[8.3 ESPECIFICAÇÃO DOS CASOS DE TESTES 64](#_Toc56702475)

[8.3.1 Item a Testar 64](#_Toc56702476)

[8.3.2 Rastreabilidade entre Requisitos e Casos de Teste 65](#_Toc56702477)

[8.3.3 Descrição dos Casos de Teste 65](#_Toc56702478)

[8.4 RESULTADOS DOS TESTES 65](#_Toc56702479)

[8.4.1 Histórico de Realização 65](#_Toc56702480)

[9 IMPLANTAÇÃO 65](#_Toc56702481)

[10 CONCLUSÃO 66](#_Toc56702482)

[REFERÊNCIAS 67](#_Toc56702483)

[APÊNDICE A - PLANO DE ELABORAÇÃO E GERENCIAMENTO DO PROJETO 69](#_Toc56702484)

[APÊNDICE B – DIAGRAMAS DE CASOS DE USO 70](#_Toc56702485)

[APÊNDICE C – RELATÓRIO DE DESEMPENHO 71](#_Toc56702486)

[APÊNDICE D – DIAGRAMA DE ENTIDADE E RELACIONAMENTO 72](#_Toc56702487)

[APÊNDICE E – DESCRIÇÃO DOS CASOS DE USO 73](#_Toc56702488)

[APÊNDICE F - MODELO DE INTERFACE COM O USUÁRIO 74](#_Toc56702489)

[APÊNDICE G - DIAGRAMA DE PACOTES 75](#_Toc56702490)

[APÊNDICE H - DIAGRAMA DE ATIVIDADES 76](#_Toc56702491)

[APÊNDICE I - ENTREVISTAS 77](#_Toc56702492)

[APÊNDICE J - RESULTADOS DO FORMULÁRIO 78](#_Toc56702493)

[APÊNDICE K - MODELO LÓGICO 79](#_Toc56702494)

[APÊNDICE L - DICIONÁRIO DO MODELO LÓGICO 80](#_Toc56702495)

[APÊNDICE M - DIAGRAMA DE CLASSES 81](#_Toc56702496)

[APÊNDICE N - ESTIMATIVA POR CASO DE USO 82](#_Toc56702497)

[APÊNDICE O – DIAGRAMA DE SISTEMA DISTRIBUIDO 83](#_Toc56702498)

[APÊNDICE P – RASTREABILIDADE ENTRE REQUISITOS E CASOS DE TESTES 84](#_Toc56702499)

[APÊNDICE Q – HISTÓRICO DE REALIZAÇÃO DE TESTES 85](#_Toc56702500)

[APÊNDICE R – DIAGRAMA DE COMPONENTES 86](#_Toc56702501)

# 1 INTRODUÇÃO

O número de pessoas que buscam por emprego, vaga temporária ou uma segunda jornada de trabalho vem aumentando a cada ano, segundo Instituto Brasileiro de Geografia e Estatística (IBGE) (2020).

Em julho de 2017, foi promulgada a Lei no. 13.467, que teve a finalidade de adequar a legislação trabalhista às novas relações de trabalho e combater a crise de desemprego no Brasil. Essa lei permite o contrato de trabalho intermitente, que é aquele no qual a prestação de serviços não é contínua e ocorre com alternância entre períodos de trabalho e de inatividade (BRASIL, 2017).

Já a falta de oferta de trabalho formal no Brasil contribui para o fenômeno conhecido como “uberização” - que é uma forma de transformar o trabalhador em um empreendedor de si mesmo, com diversas ocupações, qualificações variadas e rendimentos. Em 2019, empresas de aplicativos e sites de serviços que oferecem postos de trabalho sem vínculo empregatício formal, tornaram-se os maiores empregadores do país, empregando um contingente aproximado de 14 milhões de pessoas (BRASIL DE FATO, 2019).

Diante disso, este projeto, denominado *EasyJob,* objetiva desenvolver um sistema de software composto por componentes *Web*, móvel e de serviço *Representational State Transfer* (REST) que proporcione um meio intermediário entre quem quer trabalhar e quem quer contratar mão de obra intermitente para trabalho em um estabelecimento comercial ou evento social. Esse sistema deve ser um meio alternativo para busca de vagas, mesmo avulsas, para quem esteja desempregado, vive de *freelance* ou quer complementar sua renda financeira.

A metodologia adotada é a pesquisa bibliográfica, gerência com referência ao Guia de Conhecimentos do *Project Management Institute* (PMI)*,* especificação e modelagem baseada na *Unified Modeling Language* (UML)e construção de software orientado a objetos.

Este documento está estruturado em outros 7 capítulos cujos conteúdos são apresentados a seguir.

O Capítulo 2 traz uma revisão bibliográfica e o embasamento necessário para o entendimento deste projeto.

O Capítulo 3 demonstra o problema, o objetivo do projeto e a justificativa para a realização.

O Capítulo 4 aborda os métodos gerenciais adotados para este projeto.

O Capítulo 5 aborda os requisitos funcionais e não funcionais que constituem o sistema *EasyJob.*

O Capítulo 6 mostra a arquitetura e projeto do sistema. A arquitetura está representada pela visão estrutural, por meio dos diagramas de pacotes e pelos diagramas de classes, pela visão comportamental, por meio dos diagramas de sequência e diagramas de visão geral de interação; e pela visão de dados, por meio do modelo lógico e dicionário de dados.

No Capítulo 7, encontra-se as características técnicas de implementação do sistema de software.

No Capítulo 8, o leitor encontra o plano de teste elaborado para o *EasyJob*. Neste capítulo são especificados o tipo de teste realizado, os documentos relevantes para a elaboração do plano, o ambiente para a realização dos testes e os itens que serão testados. Além disso, o capítulo também é composto pelo relato dos resultados dos testes realizados.

No Capítulo 9, encontra-se o planejamento da implantação do sistema de software em seu ambiente de operacional.

Por fim, no Capítulo 10, encontra-se a conclusão do projeto.

Este documento contém, ainda, as referências, obras consultadas e apêndices.

# 2 REVISÃO BIBLIOGRÁFICA

Neste capítulo encontram-se conceitos e fundamentos que são necessários para o entendimento deste projeto, bem como cita trabalhos relacionados que apresentam características semelhantes ao projeto desenvolvido.

## 2.1 DESEMPREGO E TRABALHO INFORMAL

O desemprego se refere às pessoas que têm idade para trabalhar (acima de 14 anos), mas estão disponíveis e tentam encontrar trabalho e renda, portanto, são aquelas que estão desocupadas contra a vontade.

O gráfico da Figura 1 mostra a distribuição da população brasileira, de acordo com as divisões do mercado de trabalho, no 4º trimestre de 2019.

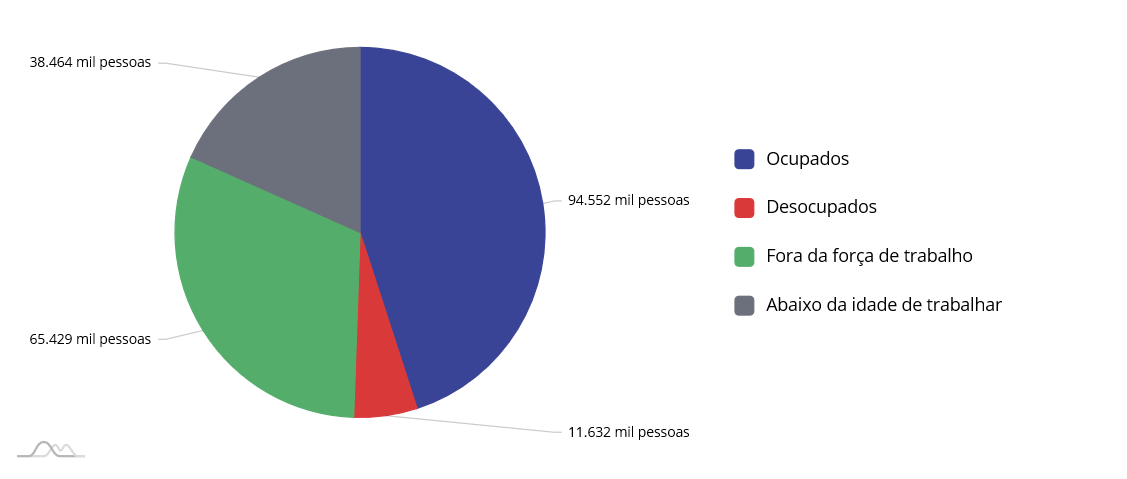


FIGURA 1 - Divisão do Mercado de Trabalho no 4° trimestre de 2019.

FONTE: IBGE (2019).

O gráfico da Figura 2 mostra a taxa de desocupação no Brasil e nas grandes regiões, no 4º trimestre de 2019.

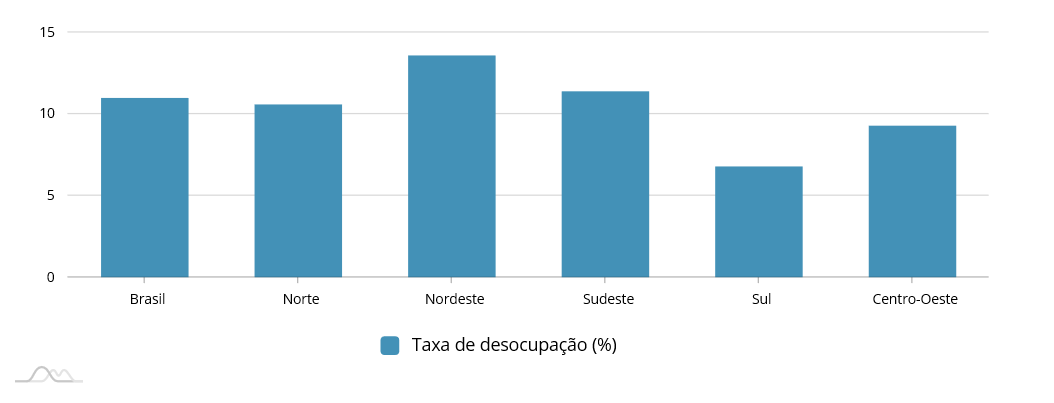


FIGURA 2 - Taxa de Desocupação no Brasil e nas grandes regiões no 4° trimestre de 2019.

FONTE: IBGE (2019).

Para lidar com a desocupação, parte dos brasileiros precisa de empregos informais, que são aqueles em que não existe o recolhimento de impostos trabalhistas e nem benefícios legais. Com a alta taxa de desemprego formal que atinge o Brasil, estima-se que cerca de 39,5 milhões de pessoas estão na informalidade (BRASIL DE FATO, 2019).

Devido a pandemia o número de brasileiros desempregados bateu recorde no número de desemprego desde o início da pandemia.

“Embora as informações sobre a desocupação tenham ficado estáveis na comparação semanal, elas sugerem que mais pessoas estejam pressionando o mercado em busca de trabalho, em meio à flexibilização das medidas de distanciamento social e à retomada das atividades econômicas”, disse em nota do IBGE(2020).

O gráfico da Figura 3 mostra a taxa de desocupação no Brasil de maio até a última semana se setembro de 2020.

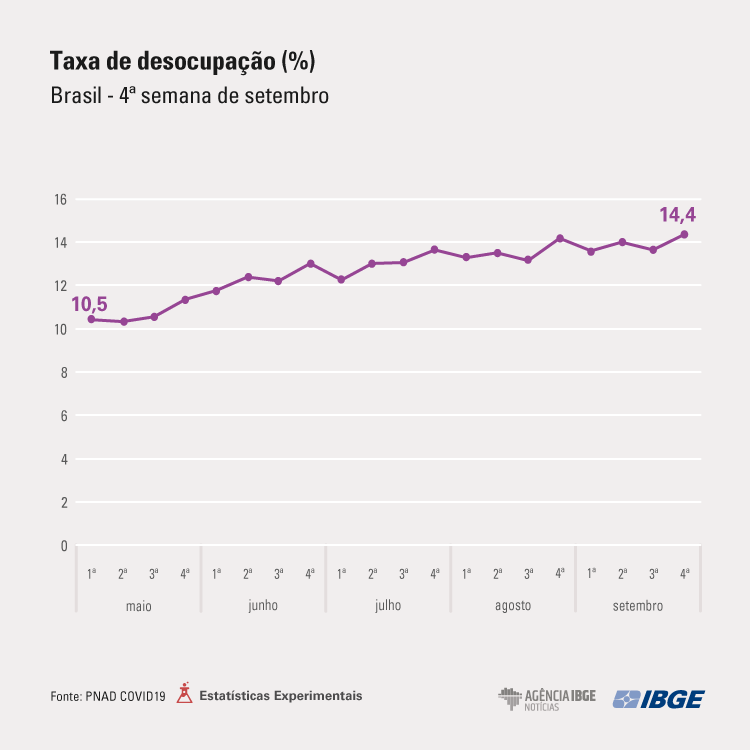


FIGURA 3- Taxa de desocupação no Brasil na pandemia.

FONTE : PNAD COVID19 2020

O emprego informal pode ser definido, segundo Alonso (2020), como sendo aquele no qual uma pessoa trabalha sem condições regulamentadas pelo governo, não há vínculo empregatício e o trabalhador não possui registro em carteira de trabalho ou de autônomo e não usufrui dos benefícios que lhes são de direito, como Fundo de Garantia do Tempo de Serviço (FGTS), direito à licença maternidade, auxílio-doença e seguro desemprego.

O mercado de trabalho informal no Brasil é constituído, na maioria, por profissionais que possuem dificuldades de voltar ao mercado de trabalho formal, principalmente, após a crise econômica que se iniciou em 2014. No mercado informal estão pessoas em situações diferentes, como empregados sem registro em carteira de trabalho, trabalhadores por conta própria, sem remuneração e, também, pequenos empregadores.

Conforme mostra o gráfico da Figura 3, o trabalho informal é uma forma de lidar com o desemprego e ele cresce na medida em que os empregos com carteira assinada diminuem. Assim, o crescimento da informalidade é acompanhado pelo declínio do número de trabalhadores com carteira de trabalho assinada e pelo crescimento dos que trabalham sem registro ou por conta própria.

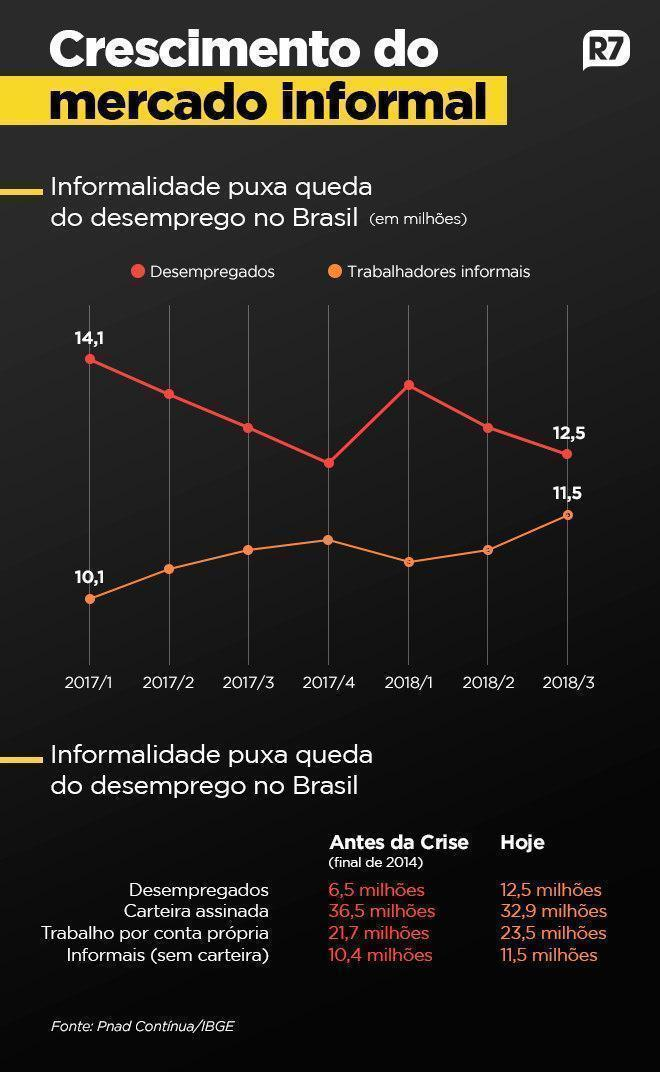


FIGURA 4- Crescimento do número de vagas informais no Brasil entre 2017 e 2018.

FONTE: Brasil de Fato (2019 apud IBGE, 2018).

## 

A “uberização” evidencia a tendência de transformação do trabalhador em microempreendedor e em trabalhador amador produtivo. Esse efeito atinge as mais diversas ocupações e segue uma alta tendência de trabalhos guiados por diversos aplicativos e com rendimentos variados.

A 'uberização' é uma nova forma de organização, de controle, de gestão do trabalho que conta com a transformação do trabalhador em um empreendedor de si próprio. Melhor seria a gente falar que ele é um auto-gerente de si (ABÍLIO*,* 2017, p. 1).

De acordo com Abílio (2017), as empresas promotoras da “uberização” - conhecidas como empresas-aplicativo (por exemplo, Uber, iFood e Loggi) - desenvolvem mecanismos de transferência de riscos e custos para uma multidão de trabalhadores autônomos. Tal transferência é gerenciada por softwares e plataformas online de propriedade dessas empresas, os quais conectam usuários trabalhadores a usuários consumidores e ditam e administram as regras (incluídos aí custos e ganhos) dessa conexão.

Estima-se que no ano de 2019, empresas-aplicativos tornaram-se os maiores empregadores do país, empregando um contingente aproximado de 14 milhões de pessoas (BRASIL DE FATO, 2019).

## 2.2 TRABALHO INTERMITENTE

È importante ressaltar a diferença entre trabalho informal e intermitente (ou *freelancing),* que são atividades com finalidades diferentes. Segundo Ribeiro (2018), o trabalho informal é aquele que é exercido por pessoas que não possuem nenhum tipo de vínculo empregatício com o empregador, já  [*freelancing* é uma modalidade de trabalho na qual o profissional, o *freelancer,* atua de forma autônoma](https://comunidade.rockcontent.com/o-que-e-freelancer/) e presta serviços para pessoas ou empresas por um determinado tempo. Ele é favorecido por questões tributárias e pela facilidade de conquistar serviços no mercado online.

A Lei no. 13.467, promulgada em Julho de 2017, teve a finalidade de adequar a legislação trabalhista às novas relações de trabalho e combater a crise de desemprego no Brasil. Essa lei permite o contrato de trabalho intermitente, conforme segue:

Art. 443. O contrato individual de trabalho poderá ser acordado tácita ou expressamente, verbalmente ou por escrito, por prazo determinado ou indeterminado, ou para prestação de trabalho intermitente [...].

§ 3o  Considera-se como intermitente o contrato de trabalho no qual a prestação de serviços, com subordinação, não é contínua, ocorrendo com alternância de períodos de prestação de serviços e de inatividade, determinados em horas, dias ou meses, independentemente do tipo de atividade do empregado e do empregador, exceto para os aeronautas, regidos por legislação própria.

De acordo com Ribeiro (2018), esse tipo de relação trabalhista foi criado com o intuito de beneficiar formas de trabalho mais flexíveis e a legislação recomenda que essa modalidade profissional deva ser combinada entre empregadores e trabalhadores.

## 2.3 TRABALHOS RELACIONADOS

Nesta seção são apresentados os aplicativos que se assemelham às funcionalidades propostas pelo sistema de software *EasyJob.* São citadas 2 soluções que oferecem serviços de captação de mão de obra intermitente: *GetNinjas* e *MyStaff*.

### 

### 2.3.1 *GetNinjas*

A plataforma *GetNinjas* foi lançada no Brasil, em 2011, pelo empreendedor Eduardo Orlando L’Hotellier. Trata-se da maior plataforma de contratação de serviços do Brasil, pois conecta as pessoas que precisam de um serviço com aquelas que o solicitam, de forma online, pela Internet (GETNINJAS, 2020).

As principais características da *GetNinjas* são:

1. pagamento via aplicativo;
2. recebimento de pedidos de acordo com a localidade;
3. comunicação entre prestador do serviço e contratante;
4. validação fim - fim para cadastros;
5. recebimento de notificações compatíveis com o perfil do interessado;
6. diversas categorias de trabalho;
7. disponibilidade para *Android* e *iOS*;
8. sistema de recomendação entre usuários.

A Figura 4 ilustra a página principale um cadastro de busca por serviço.

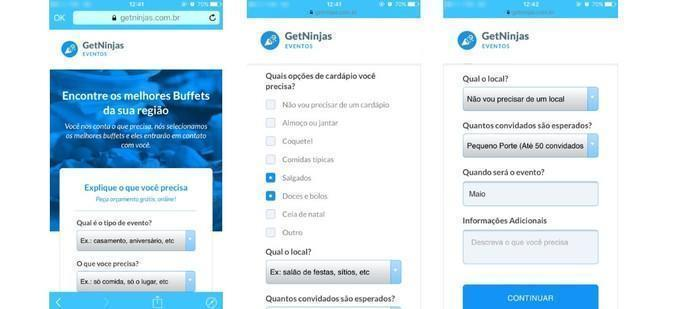


FIGURA 5- Telas do aplicativo GetNinjas

FONTE: *GetNinjas* (2020).

### 2.3.2 *MyStaff*

O aplicativo *MyStaff* foi lançado no Brasil, em 2018, pelo empresário Fernando Gaudio. Esse aplicativo permite contratar profissionais que atuam em bares, restaurantes e eventos noturnos, como *freelancers* ou trabalhos por períodos (MYSTAFF, 2020).

As principais características do aplicativo *MyStaff* são:

1. existem dois planos de monetização, plano avulso que é recolhido 18% do valor e mensalista, com pagamento mensal e de até R$ 3000,00 por mês;
2. possui um sistema de recomendações que o contratante faz para o contratado;
3. o pagamento do serviço é feito pelo aplicativo ou em dinheiro;
4. os usuários recebem as vagas de acordo com a localidade;
5. disponível para os sistemas operacionais *Android* e *IOS.*

A Figura 5 ilustra a página principal e uma tela de seleção de busca por categoria.



FIGURA 6- Telas do aplicativo Mystaff.

FONTE: *MyStaff* (2020)

Diante da fundamentação teórica apresentada, no próximo capítulo apresentam-se os objetivos do Projeto *EasyJob*.

# 3 OBJETIVO DO PROJETO

Este capítulo apresenta os problemas que emergem da necessidade de oferta de trabalho e renda e que demandam uma solução que justifica a elaboração do projeto. Além disto, são apresentados os objetivos, a justificativa, o público a quem se destina o sistema de software e os grupos funcionais com os respectivos níveis de decisão envolvidos.

## 3.1 FORMULAÇÃO DO PROBLEMA

Mesmo considerando a alta taxa de desemprego no Brasil, a falta de profissionais qualificados no mercado de trabalho é uma grande dificuldade que os empregadores enfrentam quando precisam contratar alguém, temporariamente, para trabalhar em seu estabelecimento comercial ou na organização de algum evento social, como *buffets*, festas, shows, dentro outros.

Observa-se que muitas empresas contratam pessoas com o perfil, a partir de indicações de conhecidos. Esse procedimento traz certa insegurança tanto para o empregador quanto para o empregado, pois o empregador pode não ter referências de contratantes anteriores e o empregado pode não ter conhecimento sobre a idoneidade e capacidade de seu empregador.

Ainda que os problemas apresentados possam ser preocupantes, é possível que eles sejam minimizados com a participação mútua dos envolvidos, por meio de uma plataforma *online* de oferta e procura de trabalho intermitente. Essa plataforma deve ter foco no ramo de organização de festas e shows, na qual os usuários possam procurar por vagas e pessoas especializadas e analisarem tanto o perfil do empregado, quanto do empregador, propiciando mais informações dentro dos termos oferecidos por ambos.

## 3.2 OBJETIVOS

O objetivo do projeto *EasyJob* é desenvolver um sistema de *software* que seja utilizado para divulgar ofertas e permitir buscas de vagas de trabalhos intermitentes, sendo um canal intermediário para contatos entre empregadores e empregados.

Como objetivos específicos deste projeto têm-se:

1. fornecer um canal de comunicação eficiente entre empregadores de vagas temporárias e empregados interessados;
2. oferecer um meio de localização e encontro entre ambos;
3. assegurar um meio adequado para uma pré-análise do empregado a partir de informações dadas por empregadores anteriores;
4. assegurar um meio adequado para uma pré-análise do empregador a partir de informações dadas por empregados anteriores;

## 3.3 JUSTIFICATIVA

A razão de ser criado um sistema de software para o ramo de estabelecimentos comerciais e organização de eventos sociais (serviços de *buffet* e festas em geral) se dá devido às pesquisas empíricas em relação ao trabalho nessa área, pois há uma dificuldade de procurar gente especializada na prestação de serviços. Muitas pessoas seguem esse ramo de trabalho como serviço principal e precisa qualificar e demonstrar seu serviço. Para trabalhar legalmente e ter seus benefícios nesse ramo é necessário seguir algumas exigências apresentadas a seguir.

### 3.3.1 Importância da regulamentação

De acordo com Sistema Buffet (2017), é obrigatório que os serviços de *buffet* busquem a regularização para exibir credibilidade aos clientes e evitar problemas posteriores com a legislação, problemas que podem, inclusive, afetar o crescimento do negócio.

Além dos aspectos jurídicos, seguir as normas higiênicas e de organização determinadas pelos órgãos competentes também facilita a escolha de opções dos próprios gestores e [funcionários](http://sistemabuffet.web7095.uni5.net/gestao-de-buffets/contratacao-de-autonomos-em-buffet-questoes-legais-e-juridicas-importantes/). Com uma forma simples de se direcionar, seus serviços se tornam mais seguros.

### 3.3.2 Exigências legais para *buffets* funcionarem

A Agência Nacional de Vigilância Sanitária ([ANVISA](http://portal.anvisa.gov.br/)) é a agência responsável por fiscalizar qualquer empresa do setor alimentício, incluindo os *buffet*s.

A legislação ambiental também precisa ser respeitada. Neste caso, a empresa só precisa atender às normas se os alimentos utilizados no *buffet* forem de fabricação própria — caso seja terceirizado, a preocupação deve ser majoritariamente com a ANVISA.

Basicamente, todo *buffet* precisa de instalações adequadas para conservação de alimentos, sejam eles produzidos por conta própria ou terceirizados, atender a todos os processos higiênicos (conservação e manipulação de alimentos), uso de água potável, higienização de funcionários e, tão importante quanto, ter licença de funcionamento.

Esse alvará de funcionamento deve ser conseguido no Cadastro Municipal da Vigilância em Saúde (CMVS) e Sistema Estadual de Vigilância Sanitária (SEVISA), respectivamente os órgãos municipais e estaduais de vigilância sanitária, cada qual com suas documentações e burocracias.

Por fim, após a aquisição do alvará, é preciso adquirir a caderneta de controle sanitário na Secretaria Municipal de Abastecimento, para que as vistorias estejam sempre em dia.

Com base nessas informações e como no mercado da tecnologia não possui esse tipo de software focado para esse ramo, o *EasyJob* fará todas as transações seguindo as exigências legais tanto do empregador quanto do empregado para que todo o trabalho seja dentro da lei.

## 3.4 PÚBLICO ALVO

O sistema de software é destinado às pessoas que procuram por vagas temporárias e empresas ou pessoas físicas que precisam contratar profissionais especializados na prestação de serviços em eventos sociais públicos ou privados.

## 3.5 NÍVEIS DE DECISÃO E GRUPOS FUNCIONAIS ATENDIDOS

O sistema de software *EasyJob* possuirá dois grupos funcionais, o primeiro são os empregados, prestadores de serviços. O segundo são os empregadores que serão os responsáveis por avaliar e decidir qual empregado contratar referente a sua necessidade. Em ambos os casos, as informações prestadas pelo sistema serão apenas de nível de decisão operacional.

## 3.6 ADERÊNCIA AOS OBJETIVOS DE DESENVOLVIMENTOS SUSTENTÁVEL

Espera-se que *EasyJob* contribua com as metas dos ODS propostas pela Organização das Nações Unidas (ONU, 2015). Dentre os ODS, está “promover o crescimento econômico inclusivo e sustentável, emprego pleno e produtivo e trabalho decente para todas e todos”. Visa-se o desenvolvimento sustentável que apoiam as atividades produtivas, geração de emprego decente, empreendedorismo, criatividade e inovação, e incentivar a formalização e o crescimento das micro, pequenas e médias empresas.

# 4 GERÊNCIA DO PROJETO

Neste capítulo é retratado o plano de projeto, o qual possui as partes interessadas, o modelo de ciclo de vida, os recursos necessários e as estimativas de esforço de desenvolvimento.

“O gerenciamento de projetos de software é uma parte essencial da engenharia de software. Um bom gerenciamento não pode garantir o sucesso de um projeto. No entanto, um mau gerenciamento geralmente resulta em falha do projeto” (SOMMERVILLE, 2007, p. 61).

## 4.1 PLANO DE ELABORAÇÃO E GERENCIAMENTO DO PROJETO

Nesta seção são apresentadas as partes interessados do projeto, utilizando para isto uma matriz de Poder x Interesse, o modelo de processo adotado, estimativas de esforço e também recursos humanos, *hardware* e *software* necessários para o desenvolvimento do projeto. O plano de elaboração e gerenciamento do projeto está no Apêndice A.

### 4.1.1 Partes Interessadas

De acordo com Pressman (1997 apud SOMMERVILLE, 2011) é definido por parte interessada, aquela que de forma direta ou indireta, beneficiar-se-á do projeto que está sendo desenvolvido.

Para o desenvolvimento deste projeto, foram identificados as seguintes partes interessadas:

1. estabelecimentos interessadas em contratar pessoas qualificadas e direcionadas para a vaga específica;
2. *freelancers:* aqueles que procuram vagas dentro da sua qualificação e conhecimento dentro do ramo;
3. equipe de desenvolvimento: Alexandre da Silva Ribeiro, Gabriel Ribeiro Garcia e Luan Reis de Carvalho. São os responsáveis pelo desenvolvimento das atividades compostas durante todas as fases deste projeto, com o objetivo de manter a qualidade do mesmo;
4. orientadora: profa. Eunice Gomes de Siqueira, responsável pela orientação da equipe de desenvolvimento.

### 4.2 MODELO DE CICLO DE VIDA

Segundo Sommerville (2007, p. 43) “um modelo de processo de software é uma representação abstrata de um processo de software”. Os processos de software, de forma genérica, possuem as seguintes atividades: comunicação; planejamento; modelagem; construção; e implantação.

O Modelo de Ciclo de Vida Incremental é a combinação de elementos dos fluxos de processos lineares e paralelos, onde é aplicada uma metodologia linear similar ao modelo cascata, no entanto de forma escalonada, gerando incrementos do software (PRESSMAN, 2011). A Figura 6 mostra possíveis incrementos ao longo do tempo.

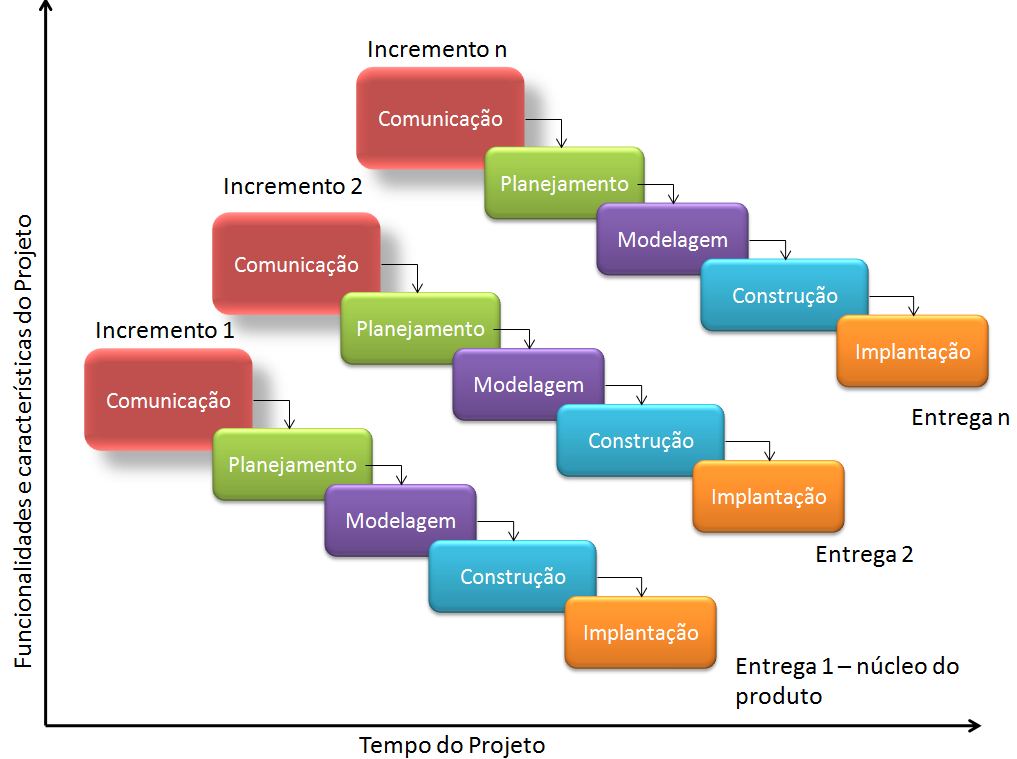


FIGURA 7- Modelo de ciclo de vida incremental.

Fonte: Pressman (2011).

O modelo de ciclo de vida que norteia o desenvolvimento do *EasyJob* é o modelo Incremental, sendo que a cada entrega é retomada a fase de planejamento da próxima entrega passando pelas fases posteriores.

Este modelo mostra-se adequado às necessidades do projeto, já que existem alguns pontos de incerteza que serão esclarecidos à sucessão do tempo e a aparição de aprimoramentos consequentes da utilização de todo o sistema. Este projeto possui 5 incrementos entregues de acordo com o Quadro 1.

|  |  |  |
| --- | --- | --- |
| **Incremento** | **Data de entrega** | **Descrição** |
| I | 28/03/2020 | Entrega dos tópicos do projeto: revisão bibliográfica, objetivos do projeto e suas seções, gerência do projeto e suas seções, especificação e análise dos requisitos com suas seções e as referências. |
| II | 23/05/2020 | Entrega dos tópicos dos projetos: Revisão modelo de ciclo de vida, justificativa e recursos necessários. Início do preenchimento dos tópicos estimativas de tamanho e esforço e gerenciamento de configuração. |
| III | 25/06/2020 | Revisão nos tópicos do modelo: Gerenciamento de configuração. |
| IV | 05/09/2020 | Atualizações das Fases 1e 2, Complementar o plano de elaboração e gerenciamento do projeto, adicionado tópico tecnologias e implementação, e plano de testes. |
| V | 31/10/2020 | Atualizações e correções das Fases 1, 2 e 3, revisão dos capítulos e correções apontadas. |
| VI | 14/11/2020 | Entrega do projeto, vídeo e dos slides para a banca examinadora final. |

QUADRO 1 - Cronograma de entregas do projeto

### 4.3 RECURSOS NECESSÁRIOS

Nesta seção são apresentados os recursos humanos, de *hardware* e de *software* necessários ao desenvolvimento do projeto.

#### **4.3.1 Recursos Humanos**

A equipe deste projeto é constituída por 3 membros, graduandos em Sistemas de Informação. O Quadro 2 apresenta a relação entre os integrantes e seus papéis no projeto.

|  |  |
| --- | --- |
| **Integrante** | **Responsabilidade** |
| Alexandre da Silva Ribeiro | Codificação, documentação, pesquisa e testes. |
| Gabriel Ribeiro Garcia | Gerência do projeto, codificação, documentação e pesquisa. |
| Luan Reis de Carvalho | Codificação, documentação, pesquisa e testes. |

QUADRO 2 - Recursos Humanos

#### **4.3.2 Recursos de Hardware**

Como recursos de *hardware* são necessários para o desenvolvimento três computadores cuja especificação mínima de processador é Intel Core i3 ou equivalente, disco rígido de 500 Gigabytes (GB), 8 GB de *random access memory* (RAM) e acesso à Internet.

#### **4.3.3 Recursos de Software**

Para o desenvolvimento do projeto, faz-se necessário a utilização de alguns *softwares* tanto gerenciais como auxiliares na execução das tarefas. O Quadro 3 descreve as ferramentas de *softwares* utilizadas no decorrer de todo o projeto.

|  |  |
| --- | --- |
| **Ferramenta** | **Versão** |
| Google Maps API | 2020 |
| Dia Modeler | 0.97 |
| Ideas Modeler | 12.46 |
| Git | 2.26.0 |
| Visual Studio Code | 1.43.2 |
| Microsoft Project | 2016 |
| Pacote *Microsoft Office* | 2016 |
| PostgreSQL | 11.7 |
| Google Chrome | 80.0.3987.149 |
| Sistema Operacional Windows 10 | 2020 |
| Node.js | 12.04 LTS |
| AngularJs | 1.7.7 |

QUADRO 3 - Ferramentas de software.

### 4.4 RELATÓRIO DE DESEMPENHO

Nesta seção é apresentado o relatório de desempenho do projeto. A descrição dos casos de uso encontra-se no Apêndice C deste documento.

Não foi entregue o Relatório de desempenho da Fase 3 (estava no FAI online).

4.5 ESTIMATIVAS DE TAMANHO E ESFORÇO

Para possibilitar estimar o esforço necessário para o desenvolvimento ainda na fase de levantamento de Casos de Uso e tendo como base a complexidade compreendida, Karner (1993)[[1]](#footnote-1) criou a métrica de PCU.

A estimativa de PCU, a partir da Fase 2, são observadas no Quadro 4.

Para a métrica de PCU foram realizadas as estimativas segundo os métodos de Schneider e Winters e de Karner. Todas as estimativas foram realizadas a partir da segunda fase do projeto, e deve-se considerar que mudanças podem ocorrer entre as fases devido ao melhor entendimento do projeto pela equipe.

|  |  |  |  |
| --- | --- | --- | --- |
| **Técnica** | **Fase 2** | **Fase 3 (reestimativa)** | **Fase 4 (reestimativa)** |
| PCU – Schneider e Winters | 1999 HH | 1735HH | 1103HH |
| PCU – Karner | 1428 HH | 2520 HH | 1103HH |

QUADRO 4 - Estimativas de esforço

Na Fase 2, seguindo a métrica de PCU de Schneider e Winters, que utilizou uma estimativa de 20 Homem/Hora (HH), e a métrica de PCU de Karner, que utilizou uma estimativa de 20 H/H, foi possível se obter estimativas do esforço necessário para a execução do projeto.

Durante o planejamento da Fase 3 foi estimado um esforço de 28Homem/Hora, por conta do desenvolvimento do EasyJob *Web Service(Api) e do Client Web*, além do tempo utilizado com testes.

No planejamento da Fase 4 foi estimado um esforço de 20 Homem/Hora que a documentação a se fazer nessa fase é menor do que nas fases anteriores, mas o desenvolvimento é maior do que nas fases anteriores.

4.6 GERENCIAMENTO DE CONFIGURAÇÃO

A gestão de configuração é essencial em todas as fases do projeto, pois, com ela, é possível melhor organizar e controlar todos os artefatos produzidos.

Neste projeto, cada integrante da equipe é designado a executar uma ou mais atividades. Dias antes da entrega do incremento previsto é realizada uma reunião para a integração das partes desenvolvidas por cada membro.

Um repositório foi criado no *Google Drive* para o controle e *backup* das versões do projeto e a parte da documentação. Esse repositório é compartilhado com todos os membros da equipe, sendo que todos possuem permissões para fazer as alterações que forem necessárias, porém, somente após comunicar os membros da equipe, porém não existe um controle interno dos documentos.

A ferramenta *Google Docs* é utilizada na execução de atividades de maior importância, onde todos os membros da equipe participam. Essa ferramenta permite que todos editem em paralelo o mesmo documento.

Para o controle de versão do código é utilizado serviço do Github, que é um repositório que implementa o padrão de versionamento Git, o membro Alexandre Ribeiro é o responsável por administrar o sistema de versão.

Na figura 8 mostra a estrutura de versionamento do projeto, tanto backend como frontend.

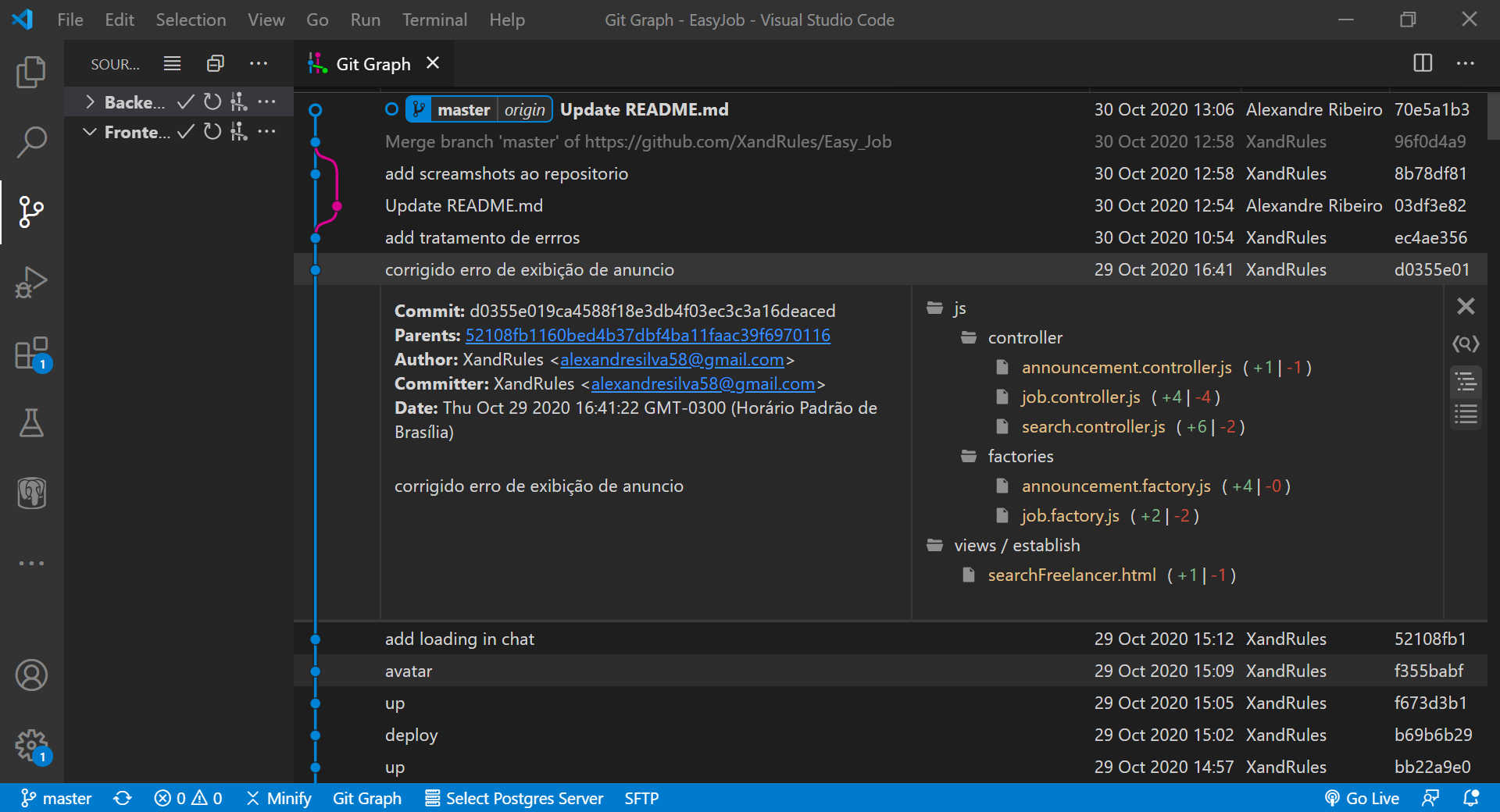


FIGURA 8 - Repositório de versionamento de código Github.

# 5 ESPECIFICAÇÃO E ANÁLISE DOS REQUISITOS DO SISTEMA

Este capítulo descreve os requisitos e apresenta os modelos de casos de uso, o modelo conceitual dos dados e modelo inicial da interface de usuário do sistema *EasyJob.*

## 5.1 DESCRIÇÃO DE REQUISITOS

Os requisitos são parte importante do desenvolvimento, pois eles descrevem como serão as funcionalidades e características que o sistema deverá apresentar ao longo dos incrementos e entregas.

### 5.1.1 Requisitos Funcionais

São as declarações de serviços que o sistema deve fornecer, como o sistema deve reagir a entradas específicas e como o sistema deve se comportar em determinadas situações (SOMMERVILLE, 2007).

Os requisitos serão classificados de acordo com sua importância para o projeto, eles poderão ser essenciais, importantes ou desejáveis. Sendo essenciais os de maior prioridade para o desenvolvimento do sistema, os importantes com uma prioridade intermediária e os desejáveis são os de menor prioridade.

Nesta seção são definidos os requisitos funcionais (RF) para os dois componentes do sistema *EasyJob*: Aplicativo *Web* (para uso em navegadores *Web*) e Aplicativo *Mobile* (para uso em dispositivos móveis com sistema operacional *Android*), ambos possuem os mesmos requisitos*.*

Módulo de Acesso

**RF01 – Realizar autenticação**

Este requisito refere-se à necessidade de o usuário realizar uma autenticação da conta já cadastrada de um *freelancer* (RF02) ou de um tipo de estabelecimento (RF03). Para realizar o acesso ao sistema, o usuário deverá acessar a opção de “Acessar” e fornecer os seguintes dados corretamente:

1. email: campo obrigatório único que já foi fornecido pelo usuário no cadastro seja ele *freelancer* (RF02) ou estabelecimento (RF03);
2. senha: campo obrigatório, com no mínimo 8 caracteres, que já foi fornecido pelo usuário no cadastro seja ele *freelancer* (RF02) ou estabelecimento (RF03);

Após ter o acesso autorizado ao sistema, o usuário poderá fazer uso do sistema*.*

Prioridade: essencial.

**RF02 – Realizar cadastro de *freelancer***

Este requisito refere-se à possibilidade de o usuário *freelancer* fazer cadastro no sistema para posterior acesso e uso. Para isso, ele deverá acessar, na página principal do aplicativo *Web* opção de “Cadastrar *freelancer*” e fornecer os seguintes dados:

1. nome: campo obrigatório que se refere ao nome completo do usuário;
2. *e-mail*: campo obrigatório de valor único, que será utilizado para acesso ao sistema;
3. CPF: campo obrigatório de valor único;
4. número do telefone celular: campo opcional de valor único;
5. senha: campo obrigatório, com no mínimo 8 caracteres, que será utilizado para acesso ao sistema;
6. sexo: campo obrigatório;
7. data de nascimento: campo obrigatório;
8. especialidade: campo obrigatório, que poderá ser mais de uma de acordo com as especialidades cadastradas;
9. endereço: campo obrigatório, que possui os dados da localização do usuário, como o nome do logradouro, bairro, município, UF, número, CEP e complemento;
10. disponibilidade: campo que sinaliza que o usuário *freelancer* está disponível ou indisponível para prestar serviços;
11. Concordância com os termos de uso do sistema: campo obrigatório, nele o usuário irá concordar que aceita todos os termos de uso do sistema.

Depois de fornecer esses dados, o usuário *freelancer* concluirá o seu cadastro e poderá fazer uso do sistema.

Prioridade: essencial.

**RF03 – Realizar cadastro de estabelecimento comercial**

Este requisito refere-se à possibilidade de o usuário (responsável por um estabelecimento comercial) fazer cadastro no sistema para posterior acesso e uso. Para isso, ele deverá acessar, na página principal do aplicativo *Web,* a opção de “Cadastrar estabelecimento” e fornecer os seguintes dados:

1. nome: campo obrigatório que se refere ao nome do estabelecimento;
2. razão social: campo obrigatório;
3. *e-mail*: campo obrigatório de valor único, que será utilizado para acesso ao sistema;
4. CNPJ: campo obrigatório de valor único;
5. número do telefone celular ou fixo: campo obrigatório de valor único;
6. senha: campo obrigatório, com no mínimo 8 caracteres, que será utilizado para acesso ao sistema;
7. endereço: campo obrigatório, que possui os dados da localização do usuário, como o nome do logradouro, bairro, município, UF, número, CEP e complemento;
8. concordância com os termos de uso do sistema: campo obrigatório, nele o usuário irá concordar que aceita todos os termos de uso do sistema.

Depois de fornecer esses dados, o usuário concluirá o cadastro de seu estabelecimento e poderá fazer uso do sistema.

Prioridade: essencial.

**RF04 – Alterar senha de acesso do usuário**

Este requisito refere-se à possibilidade de alteração da senha de acesso ao sistema por um usuário já cadastrado. Em caso de perda ou esquecimento da senha, o usuário poderá utilizar a opção de alteração de senha, e fornecer os seguintes dados:

1. e-mail: campo obrigatório único, com no mínimo 7 caracteres, que já foi fornecido pelo usuário no cadastro (RF02 e RF03);

Após fornecer o e-mail, o usuário receberá um *link* para modificar sua senha, então ele poderá digitar sua nova senha e concluir a alteração. Esta senha será utilizada nas próximas vezes que o usuário realizar a autenticação (RF01).

Prioridade: essencial.

**RF05 - Remover conta de acesso do usuário**

Este requisito refere-se à possibilidade de um usuário, previamente cadastrado remover a sua conta do acesso ao sistema, sendo excluídos todos os dados a ele associados. Para isso, ele deverá entrar no aplicativo *Web* (RF01) e selecionar a opção “Excluir conta”, e confirmar.

Prioridade: essencial.

**RF06 – Enviar mensagem ao administrador do sistema**

Este requisito diz respeito à opção dada ao usuário para enviar uma mensagem ao administrador do sistema *EasyJob*. Para acessar essa funcionalidade, o usuário deverá estar previamente cadastrado e autenticado na aplicação Web(RF01), acessar a opção “Contatar administrador” e preencher os seguintes campos:

1. mensagem: campo obrigatório, deverá conter no mínimo 10 caracteres, com a mensagem que o usuário deseja enviar ao administrador, podendo ser um pedido de ajuda, dúvida, reclamação e etc;
2. imagem: é um campo opcional, caso o usuário queira, ele poderá adicionar uma imagem para melhor explicar sua mensagem;

Após digitar uma mensagem, o usuário já poderá enviar ao administrador, que receberá no e-mail e após ler, poderá enviar outro e-mail para o usuário com a resposta.

Prioridade: desejável.

**RF07 – Pesquisar serviços fornecidos**

Esse requisito refere-se à possibilidade de um usuário, pesquisar por um serviço que deseja contratar, mesmo não estando previamente cadastrado (RF02 e RF03) ou autenticado (RF01). Utilizando a pesquisa, ele deverá digitar um dos seguintes dados:

1. serviço procurado: o usuário deverá informar o título do serviço que precisa;
2. localidade de oferta do serviço.

Após digitar, a pesquisa fornecerá como resultado os serviços que mais se relacionam ao que foi buscado pelo usuário. Também será considerada a localização do usuário, sendo que os serviços mostrados serão de estabelecimentos mais próximos a ele.

Prioridade: essencial.

Módulo específico para os usuários do tipo *freelancer*

**RF08 – Criar anúncio de serviço a ser oferecido**

Este requisito refere-se à criação de um anúncio de serviço a ser oferecido por um *freelancer.* O usuário *freelancer* para utilizar essa funcionalidade deverá já estar cadastrado (RF02) e ter se autenticado no sistema (RF01). Depois disso, ele deverá escolher a opção “criar anúncio de serviço” e fornecer os seguintes dados:

1. Título do serviço oferecido: campo obrigatório, com no mínimo 6 e no máximo 30 caracteres, onde o usuário deverá dar um título ao seu serviço;
2. Preço por hora: campo opcional, onde o usuário poderá informar o preço que cobra por hora de seu serviço. Caso esse campo não seja preenchido, o valor padrão será “a combinar”;
3. Imagem: campo opcional a ser utilizado para *upload* de uma imagem que representará o serviço fornecido;
4. Turno: campo opcional, a ser utilizado para informar o horário que normalmente o usuário estará disponível para realizar o serviço. Poderá ser escolhido um ou mais valores entre as opções que são: manhã, tarde e noite;
5. Dia(s) da semana: campo opcional a ser utilizado para informar o(s) dia(s) da semana que normalmente o usuário estará disponível para realizar o serviço. O usuário deverá selecionar um ou mais dias entre as opções: domingo, segunda, terça, quarta, quinta, sexta e sábado;
6. Tipo: campo obrigatório, onde o usuário deverá informar o tipo em que se enquadra o seu serviço, podendo ser: limpeza, música, bebidas, cozinha, fotografia e filmagem, dentre outros;
7. Descrição: campo obrigatório, com no mínimo 6 e no máximo 60 caracteres, onde o usuário deverá descrever o serviço fornecido.

Após a entrada dos dados corretamente, o anúncio de serviço do usuário será cadastrado no sistema.

Prioridade: essencial.

**RF09 – Aceitar a realização de um serviço**

Esse requisito refere-se à possibilidade de o usuário *freelancer* previamente cadastrado (RF02) aceitar a realização de um serviço que tenha sido solicitado (RF13) por um estabelecimento também já cadastrado (RF03). Após o usuário confirmar que realizará o serviço, ambos receberão mensagens de e-mails informando os dados preenchidos na solicitação do serviço (RF13).

Prioridade: essencial.

**RF10 - Sinalizar indisponibilidade**

Esse requisito refere-se à possibilidade de um usuário *freelancer* escolher a opção que sinaliza que ele está indisponível para realizar serviços. Quando o *freelancer* escolher essa opção, o sistema não mostrará aos usuários os anúncios criados por ele.

Prioridade: importante.

**RF11 - Editar os dados de um anúncio de serviço**

Esse requisito se refere à possibilidade de um usuário *freelancer* atualizar ou apagar, se quiser, qualquer um dos dados de seus anúncios de serviços já cadastrados (RF08).

Prioridade: essencial.

**RF12 - Excluir um anúncio de serviço**

Esse requisito se refere à possibilidade de o usuário excluir seus anúncios de serviços já cadastrados (RF08). Poderá ser excluído qualquer anúncio de serviço, que não esteja contratado no momento da exclusão.

Prioridade: essencial.

Módulo específico para os usuários do tipo estabelecimento

**RF13 – Contratar serviço**

Este requisito refere-se à funcionalidade que permite ao usuário dono de estabelecimento contratar um serviço. Para isso ele deverá estar previamente cadastrado (RF03) e com acesso (RF01). O usuário deverá selecionar o serviço que deseja contratar e a opção “contratar”, após isso deverá fornecer os seguintes dados para o contrato:

1. Horário: é um campo obrigatório, que deverá possuir o horário de realização do serviço;
2. Data: é um campo obrigatório, que deverá possuir a data de realização do serviço;
3. Localização: é um campo obrigatório, que deverá receber a localização, a partir do endereço fornecido pelo estabelecimento;
4. Concordância com os termos de contrato: campo obrigatório, em que o usuário deverá concordar com os termos para a contratação de um serviço.

Prioridade: essencial.

**RF14 - Sinalizar que o serviço foi realizado**

Esse requisito refere-se à possibilidade de um usuário (responsável pelo estabelecimento), previamente cadastrado, escolher a opção que sinaliza que o serviço contratado (RF13) por ele foi realizado, após isso ele poderá ser avaliado (RF15).

Prioridade: essencial.

**RF15 - Avaliar serviço**

Este requisito refere-se à funcionalidade que permite ao estabelecimento avaliar um serviço já contratado (RF13) e realizado, para isso ele deverá estar previamente cadastrado (RF04) e com acesso ao sistema (RF03), além de já ter recebido o serviço que pretende avaliar. O usuário deverá selecionar o serviço que deseja avaliar e preencher os seguintes campos:

1. Nota: é um campo obrigatório, que deve possuir a nota do serviço de 1 a 5;
2. Comentário: é um campo opcional, que poderá possuir um comentário e a avaliação do usuário sobre o serviço contratado.

Prioridade: essencial.

Módulo de Conversação (*chat*)

**RF16 – Enviar convite para nova conversa**

Este requisito refere-se à possibilidade de um estabelecimento, previamente cadastrado (RF03), enviar um convite para uma conversa com um *freelancer,* também já cadastrado (RF02). Esse convite deverá ser feito em anúncio de serviço criado por um *freelancer* (RF08), após selecionar qual serviço deseja (RF07), o estabelecimento contratante poderá escolher a opção “iniciar conversa”, e enviar uma mensagem para o fornecedor do serviço selecionado, o *freelancer* deverá aceitar a conversa (RF16) para que os dois possam se comunicar. A conversa será aberta em modo do tipo *chat*, podendo o *freelancer* e o estabelecimento conversar novamente quando quiserem.

Será permitida ao usuário, a exclusão de toda conversa salva no seu *chat* quando quiser, sendo que a mesma permanecerá armazenada no *chat* do outro usuário que participava da conversa.

Prioridade: importante.

**RF17 – Aceitar conversa**

Este requisito refere-se à possibilidade do *freelancer* previamente cadastrado (RF02) poder aceitar uma conversa com um estabelecimento, também já cadastrado (RF03), interessado em seus serviços. O *freelancer* após aceitar a conversa, poderá visualizar as mensagens e responder se assim desejar. Todas as mensagens já recebidas por um usuário ficarão armazenadas, podendo os dois usuários participantes da conversa iniciarem nova conversa quando quiserem.

Será permitida ao usuário, a exclusão de qualquer conversa salva na sessão do *chat* quando quiser, sendo que a sessão permanecerá armazenada no *chat* do outro usuário que participar da conversa.

Prioridade: importante.

### 5.1.2 Requisitos Não Funcionais

Os requisitos não funcionais são restrições nos serviços e nas funções oferecidas pelo sistema e, por exemplo, podem incluir restrições de tempo, restrições no processo de desenvolvimento e outras impostas por normas legais ou organizacionais (SOMMERVILLE, 2007).

A seguir são descritos os requisitos não funcionais (RNF) do sistema *EasyJob*.

#### 5.1.2.1 Requisitos de Produto

**RNF01 – Disponibilidade do sistema**

A taxa de disponibilidade do servidor deve ser de 98%, aproximadamente, 358 dias por ano, para atender as necessidades dos usuários.

**RNF02 – *Layout* responsivo**

O aplicativo *Web* deverá se ajustar a diversos formatos de tela de diferentes dispositivos.

**RNF03 – Facilidade de acesso**

O sistema possibilitará que o acesso seja feito pela *Web* ou por dispositivos móveis, facilitando o acesso por diferentes dispositivos.

#### 5.1.2.2 Requisitos Organizacionais

Os requisitos organizacionais se referem às políticas, metodologias e padrões da organização que utilizará este sistema especificado.

1. Requisitos de entrega

**RNF04 – Datas de entregas dos incrementos**

As datas marco de entrega dos artefatos no primeiro semestre de 2020 estão definidas em quatro datas: 28 de março de 2020, 25 de abril de 2020, 23 de maio de 2020 e 05 de junho de 2020. No segundo semestre de 2020 as entregas serão nas datas: 05 de setembro, 10 de outubro, 31 de outubro e 14 de novembro.

**RNF05 - Ambiente de desenvolvimento**

Deverá ser utilizado o Microsoft Visual Code, versão 1.45.

**RNF06 – Linguagem de programação e serviços REST**

Para se criar o aplicativo *mobile* será utilizada a linguagem de programação JavaScript e a framework AngularJs, versão 1.7.7.

Para o banco de dados será utilizado a *Structured Query Language* (SQL).

O aplicativo *Web* utilizará o *Hypertext Markup Language* (HTML), *Cascading Style Sheets* (CSS) e também *Javascript.*

Os serviços acessados via API REST serão desenvolvidos por meio do Node.js, versão 4.0.0.

1. Requisitos de padrões

**RNF07 – Padrão de documentação**

A documentação do projeto seguirá as diretrizes técnicas para elaboração de trabalhos da FAI – Centro de Ensino Superior em Gestão, Tecnologia e Educação.

**RNF08– Processo de desenvolvimento**

O desenvolvimento do sistema tem como processo aplicado o Modelo Incremental.

#### 5.1.2.3 Requisitos Externos

Os requisitos externos são de fatores externos ao sistema e seu processo.

1. Requisitos de interoperabilidade

**RNF09– Compatibilidade com navegadores**

O aplicativo *Web* deverá assegurar a compatibilidade com o navegador Google Chrome, a partir da versão 81.0.4044.92.

**RNF10 - Compatibilidade com sistemas operacionais**

O aplicativo *mobile* deve assegurar a compatibilidade com o sistema operacional *Android* a partir da versão 6. O aplicativo *Web* poderá ser utilizado por qualquer sistema operacional, contanto que seja acessado por um navegador como o *Google Chrome*.

**RNF11 – Sistema Gerenciador de Banco de dados (SGBD)**

Deverá ser utilizado o *PostgreSQL* versão 12.0 para persistência dos dados.

**RNF12 – Servidor de E-mail (SMTP)**

Deverá ser utilizado um servidor de e-mail SMTP, sendo ele o *Nodemailer* (versão 6.4.2), pois é um módulo para Node.js já consolidado entre a comunidade node, apresenta boa documentação e é *open-source* sob licença MIT.

1. Requisitos éticos

**RNF13 – Confidencialidade das informações**

Os dados particulares dos usuários do sistema não poderão ser expostos sem a permissão do mesmo e os dados sensíveis devem ser criptografados no banco de dados.

1. Requisitos de privacidade e segurança

**RNF14– Políticas de senhas**

As senhas de acesso devem ser formadas por caracteres numéricos e letras e possuir no mínimo 8 caracteres. As senhas armazenadas pelo sistema deverão ser criptografadas antes de serem armazenadas.

**RNF15– Privacidade dos dados cadastrais**

Os dados cadastrados devem ser armazenados de forma segura para evitar acessos indevidos e serão manipulados conforme os termos de uso apresentados aos usuários.

**RNF16 - Protocolo de Comunicação Segura**

As comunicações entre clientes e servidor do sistema devem acontecer por meio do protocolo *Hypertext Transfer Protocol Secure* (HTTPS).

**RNF17– Termos de Uso**

Para que o acesso ao sistema seja autorizado, o usuário deverá aceitar o termo de uso apresentado. Caso ocorra alguma modificação nesses termos, o usuário deverá novamente concordar com as alterações. Se isso não acontecer, sua conta deverá ser removida do sistema.

## 

## 5.2 ANÁLISE DOS REQUISITOS

Neste projeto, a visão funcional é obtida por meio do Modelo de Casos de Uso, que é composto pelo Diagrama de Casos de Uso e pela descrição dos fluxos de eventos.

### 5.2.1 Diagrama de Casos de Uso

Segundo Pressman (2011), o primeiro passo para construir um caso de uso é definir o conjunto de atores envolvidos no sistema. Um ator é qualquer elemento que se comunica com o sistema e está externo à fronteira dele e que possui uma ou mais metas ao utilizar o sistema.

O diagrama de casos de uso modelado para o sistema *EasyJob* encontra-se no Apêndice B deste documento.

### 5.2.2 Descrição dos Casos de Uso

Nesta seção são apresentadas as descrições dos casos de uso, construídos a partir dos requisitos funcionais. Por meio da descrição dos casos de uso, é possível visualizar todos os cenários de cada caso de uso, os atores envolvidos, suas pré-condições e pós-condições.

A descrição dos casos de uso encontra-se no Apêndice E deste documento.

### 5.2.3 Modelo Conceitual dos Dados

O projeto lógico construído para o sistema utiliza o conceito do modelo de entidade-relacionamento para especificar os dados permanentes que serão manipulados pelo sistema.

Desta forma, o Diagrama Entidade-Relacionamento do sistema *Easy Job* encontra-se no Apêndice D deste documento.

## **5.2.4 Modelo Inicial da Interface de Usuário**

O modelo inicial de interface com usuário é uma forma de melhor entender as necessidades do projeto e encontra-se no Apêndice F deste documento.

# 6 ARQUITETURA E PROJETO DO SISTEMA DE SOFTWARE

Neste capítulo é apresentada a arquitetura e o projeto do sistema de software com a modelagem da visão estrutural, comportamental e dos dados do sistema *EasyJob*. Utiliza-se a Linguagem de Modelagem Unificada (*Unified Modeling Language* - UML) para a modelagem das visões.

# 6.1 VISÃO ESTRUTURAL

Nesta seção são apresentados os diagramas de pacotes e de classes do sistema, que permitem apresentar a estrutura do sistemae seus componentes.

# 6.1.1 Diagrama de Pacotes

No diagrama de pacotes é mostrado como estão agrupadas as classes de objetos do sistema. Para isso, utiliza-se o padrão arquitetural MVC, sendo que existe uma camada de interação com o usuário *(view*), uma camada de manipulação de dados *(model)* e uma camada para realizar controle (*controller)*. Também estará no diagrama de pacotes a API REST que será usada na aplicação, além disso o sistema conterá dois tipos de pacotes de interface com usuário, um para o aplicativo *Web* e outro para o aplicativo *mobile*. O diagrama de pacotes se encontra no Apêndice G.

# 6.1.2 Diagramas de Classes

No diagrama de classes são mostradas as classes vão ser utilizadas no desenvolvimento do sistema. O diagrama de classes apresentado se refere ao pacote Entidade que se encontra no Apêndice M.

# 6.2 VISÃO COMPORTAMENTAL

Nesta seção são apresentados os diagramas de sequência e de atividades do sistema *EasyJob.*

# 6.2.1 Diagrama de Sequência

Feito a partir do diagrama de casos de uso, o diagrama de sequência mostra a interação entre objetos do sistema. Ele se encontra no Apêndice O. => Dentro do apêndice está o Diagrama de Distribuição do SD.

# 6.2.2 Diagrama de Atividades

O diagrama de atividades mostra as ações necessárias para se realizar cada caso de uso. Os diagramas de atividade de 3 casos de uso do sistema *EasyJob* se encontram no Apêndice H.

# 6.3 VISÃO DE DADOS

Nesta seção serão apresentados o modelo lógico e o dicionário de dados do modelo lógico.

# 6.3.1 Modelo Lógico

No modelo lógico são apresentadas as relações, os atributos, as chaves primárias, chaves únicas e chaves estrangeiras do banco de dados. Ele se encontra no Apêndice K.

# 6.3.2 Dicionário de Dados do Modelo Lógico

O dicionário de dados consiste numa lista organizada de todos os elementos de dados que são pertinentes para o sistema. Ele consiste num ponto de referência de todos os elementos envolvidos na medida em que permite associar um significado a cada termo utilizado. O dicionário de dados encontra-se no Apêndice L.

# 6.4 PROJETO DA INTERAÇÃO HUMANO-COMPUTADOR

Esta seção mostra o perfil de possíveis usuários do sistema *EasyJob* e como utilizarão o sistema. Nesta seção será apresentado o perfil do usuário, o projeto da interface com o usuário, as heurísticas de usabilidade e acessibilidade.

# 6.4.1 Perfil do Usuário

A fim de conhecer as características de possíveis usuários do sistema *EasyJob*, entrevistas foram realizadas para encontrar os perfis deles e de como utilizarão o sistema. Foram entrevistadas 3 pessoas como possíveis usuárias do sistema *EasyJob*. As entrevistas se encontram no Apêndice I.

Também com o mesmo intuito, criou-se um questionário semiestruturado (formulário no *Google Forms*) que foi divulgado para várias pessoas expressarem sua opinião sobre o trabalho intermitente e o sistema que proposto neste projeto. O questionário contém as seguintes perguntas:

a) Você em caso de estar desempregado ou precisando de uma renda extra, aceitaria trabalhar como *freelancer*?

b) Qual(is) das seguintes opções você aceitaria um emprego? São fornecidas as opções: garçom, segurança, caixa, *motoboy, bartender,* recepcionista, cozinheiro, banda, nenhuma, esperaria uma opção CLT da minha área e outros.

c) Você usaria um aplicativo para se candidatar a uma vaga de *freelancer* citada acima?

d) O que seria importante ter nesse aplicativo ou site para te encorajar a usar o serviço?

e) Cite uma função que você gostaria de encontrar no *website* ou aplicativo.

f) Qual categoria você acha que faltou na lista?

As respostas obtidas pela aplicação do questionário mostram que a maioria dos respondentes aceita trabalhos intermitentes, caso estejam desempregados ou precisando de uma renda extra. Sobre qual dos empregos intermitentes aceitaria, a maioria respondeu outros, mostrando que aceitaria outros trabalhos intermitentes que não estavam entre as opções do questionário. A maioria respondeu também que usaria um sistema para se candidatar a uma vaga, e que seria importante ter muitas ofertas de trabalho, garantia de segurança ao usar o sistema e que informações detalhadas sobre os serviços a ser realizados e as vagas possam ser fornecidas. As pessoas que responderam ao questionário acham que a função de buscas por serviços é importante para o sistema e que este poderia ter a possibilidade de mostrar serviços na área de tecnologia da informação. Os resultados do questionário se encontram no Apêndice J.

# 6.4.2 Projeto da Interface de Usuário

A interface utiliza um plano de fundo branco, com uma barra de navegação cinza e outros ícones com a cor azul. O texto é apresentado com a cor preta ou branca, com fonte Arial com tamanhos variados.

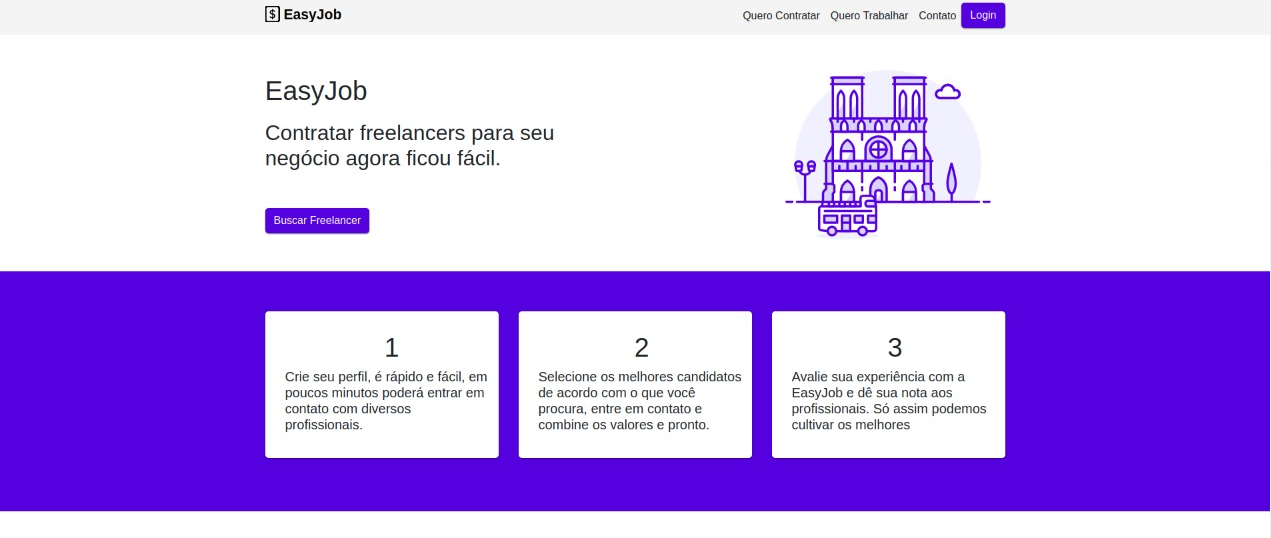


FIGURA 9 - Tela inicial do *EasyJob*



FIGURA 10 - Tela de inscrição do *EasyJob*.

**6.4.3 Heurísticas de Usabilidade**

A interface de usuário possui diversos símbolos que facilitam o entendimento das funções do sistema *EasyJob.* Na interface, o usuário terá total liberdade para navegar entre as páginas do sistema, cancelar operações que realizou e excluir sua conta se assim desejar. Todas as informações de como realizar as operações no sistema estarão também disponíveis para o usuário, caso ocorra algum erro será informado ao usuário incluindo como resolvê-lo. A interface conterá poucas figuras e cores facilitando o entendimento de como realizar as tarefas.

É preciso citar uma heurística de usabilidade por vez e apresentar telas do sistema que mostram a aplicação de cada uma delas.

# 6.4.4 Acessibilidade

Todas as imagens possuem os atributos *alt* das *tags* <img> da HTML, preenchidos com alguma frase que dê significado para elas, possibilitando que pessoas com alguma dificuldade visual possam utilizar leitores de tela. O sistema utiliza também poucas cores e faz um contraste significativo entre elas para facilitar a visualização.

6.5 PROJETO DO SISTEMA DISTRIBUÍDO

Um sistema distribuído é um conjunto de computadores independentes que se apresenta a seus usuários como um sistema único e coerente. Não há premissa ou uma configuração necessária com relação aos computadores do sistema distribuído, de forma que essas configurações, bem com sua distribuição de processamento, devem ser transparentes aos usuários, que devem ter a percepção de estar utilizando um único sistema (TANENBAUM, 1999).

# 6.5.1 Procedimentos para Tratamento dos Desafios

Nesta seção, apresentam-se obstáculos que, eventualmente, podem vir a acontecer no desenvolvimento e implementação de um sistema distribuído, de forma que os mesmos sejam superados a fim de atribuir uma qualidade satisfatória ao sistema.

# 6.5.1.1 Heterogeneidade

O sistema distribuído é composto de diversos *hosts* interconectados, onde cada um pode possuir um sistema operacional diferente, que por sua vez pode executar um cliente que foi escrito em uma linguagem de programação diferente da qual foi utilizada no desenvolvimento de determinado serviço oferecido por um servidor ao qual o cliente se conecta. Justamente pela existência de diferentes nós computacionais dentro do mesmo sistema distribuído, este se torna um obstáculo para o mesmo, sendo necessário neste projeto, o desenvolvimento de uma API padronizada para consultas e operações de escrita dentro da arquitetura.

O projeto utilizará a API para fazer a comunicação entre os diferentes componentes do sistema *EasyJob* (aplicativos *Web* e *mobile*) com o banco de dados, sendo que cada um dos componentes será feito com tecnologias diferentes.

# 6.5.1.2 Escalabilidade

A escalabilidade vem como um componente importante da implementação de um sistema distribuído, o qual estando em rede, pode ter um aumento ou diminuição de possíveis *hosts* clientes dentro da arquitetura. Esta possibilidade pode acarretar perda de poder de processamento do sistema. Desta forma, a arquitetura do sistema distribuído deve ser dinâmica a ponto de suportar a conexão e desconexão de clientes, de acordo com a demanda.

Será utilizada a escalabilidade horizontal, que significa que quando houver necessidade será adquirido outros equipamentos para atender a demanda de recursos.

# 6.5.1.3 Abertura

A abertura no âmbito de um sistema distribuído se faz importante na manutenção ou realização de novas implementações. Para isso, os métodos devem ser padronizados e novos métodos adicionados devem ter a capacidade de se comunicarem com os já existentes sem causar-lhes danos.

No desenvolvimento do sistema foi adotado diversos padrões arquiteturais para que seja garantido que não ocorram graves erros. Dentre esses padrões podemos citar o MVC, que já foi citado anteriormente.

# 6.5.1.4 Segurança

Os aspectos de segurança devem ser considerados na problemática quando se projeta um sistema distribuído. Requisitos como confidencialidade, autenticidade, integridade e disponibilidade devem ser tratados de forma especial pelo sistema. Quanto ao obstáculo da confidencialidade, o *EasyJob* pode fazer uso de métodos criptográficos em suas transações, bem como implementar o controle de acesso por usuário. Na autenticidade, deve utilizar certificados digitais tanto no cliente *Web*, quanto na aplicação *mobile*. Em consideração à integridade, os dados devem ser checados antes de qualquer transação ser realizada, evitando dados fantasmas e perda de conteúdo dentro do sistema. Quanto à disponibilidade, o *host* servidor deve possuir um controle de *proxy*, o qual deve estar constantemente atualizado bloqueando assim ataques, como o de negação de serviço.

# 6.6.1.5 Manuseio de falhas

O controle das falhas dentro de um sistema distribuído é certamente um dos maiores desafios em seu desenvolvimento. Uma vez que existem muitos componentes computacionais dentro da mesma arquitetura, a observação e o tratamento quanto às falhas devem ser constantes e devem ser informados aos administradores, para que os mesmos possam vir a tomar decisões e entrarem com medidas corretivas. Dentro do *EasyJob*, podem ser implementados mecanismos de *log* com falhas de processamento, de forma que este arquivo possa ser consultado para aplicar ações por parte dos administradores.

# 6.5.1.6 Concorrência

Com o aumento do número de clientes dentro do sistema distribuído, é possível que dois venham a tentar acessar o mesmo recurso, e a arquitetura deve ser projetada de forma que todas as requisições sejam respondidas, positiva ou negativamente. O *EasyJob* deve possuir um controle de requisições efetuadas, principalmente em relação a seu banco de dados, o qual deve permitir que apenas uma requisição manipule o recurso por vez, a fim de evitar consultas e transações inconsistentes no banco de dados e nas aplicações.

# 6.5.1.7 Transparência

Como em um sistema distribuído, os recursos computacionais estão localizados em diferentes componentes, o acesso aos mesmos deve ser facilmente realizado, sendo transparente ao cliente (usuário, programador ou processo) o fato de o mesmo estar acessando recursos locais ou remotos. O *EasyJob* deve permitir que o cliente acesse os recursos de forma fácil e ágil, não fazendo distinção sobre a localidade do componente no qual o recurso está localizado.

# 6.5.2 Tecnologias e Arquiteturas de Distribuição

Para a programação dos componentes do sistema de *software* EasyJob, foi utilizado *Javascript* com AngularJSe *NodeJs.*

Com esta tecnologia, é possível estabelecer uma arquitetura cliente-servidor. Neste modelo, os processos do sistema distribuído são divididos em dois grupos: clientes e servidores. Um servidor é um processo que implementa um serviço específico, já um cliente, é um processo que requisita um serviço de um servidor enviando-lhe uma requisição e aguardando uma resposta do servidor (TANENBAUM, 1999).

No Apêndice O está o diagrama de distribuição projetado.

# 7 IMPLEMENTAÇÃO DO SISTEMA DE SOFTWARE

Este capítulo apresenta a implementação do sistema *EasyJob*. As informações expostas neste capítulo devem auxiliar para um melhor conhecimento do sistema quanto à implementação do código fonte, *frameworks* e bibliotecas adotadas e esquemas de bancos de dados.

## 7.1 COMPONENTES DO SISTEMA DE SOFTWARE

O sistema EasyJob possui os componentes *Web*, API.

O diagrama de componentes é a representação dos componentes, suas interfaces e suas dependências, os objetivos principais esperados em um diagrama de componentes é definir e mostrar a configuração do software por meio dos relacionamentos de dependência;

O diagrama de componentes *EasyJob* encontra-se no Apêndice S.

## 7.2 TECNOLOGIAS DE IMPLANTAÇÃO

Nesta seção apresentam-se as linguagens, *frameworks* e bibliotecas, os padrões de projetos aplicados, as convenções e regras de codificação e a estrutura física do banco de dados.

# 7.2.1 Linguagens de Programação e *Frameworks* Adotados

a) Linguagens

Para a programação dos componentes do sistema de *software* EasyJob utiliza-se a linguagem *Javascript.* Esta é voltada nas duas camadas do projeto, sendo a primeira o *back-end* com a API em *Node.js* e a segunda o *front-end* utilizando a framework AngularJs.

Também se utilizam HMTL 5 juntamente com CSS 3.

b) *Frameworks*

Um *framework* descreve a arquitetura de um sistema genérico, os tipos de objetos e as interações entre os mesmos. Ele pode ser visto como o esqueleto de um sistema que pode ser customizado conforme a necessidade em questão. Para Johnson (1997, p.39–42), um *framework* pode facilitar a construção de novos componentes e fornecer uma interface padrão para os mesmos trocarem dados, manipularem erros e operações.

No desenvolvimento do *EasyJob* empregam-se os seguintes *frameworks:*

*Express:* é um *framework* para aplicações web para Node.js, lançado como software livre e de código aberto sob a Licença MIT. É feito para otimizar a construção de aplicações web e APIs. É um dos mais populares *frameworks*para servidores em Node.js.

*Sequelize*: é um  *Object-Relational Mapper* (ORM) para Node.js, que tem suporte aos bancos de dados PostgreSQL, MariaDB, MySQL, SQLite e MSSQL, como ORM ele faz o mapeamento de dados relacionais (tabelas, colunas e linhas) para objetos *JavaScript*.

AngularJs: é um framework que auxilia no desenvolvimento da interface com usuário.

c) Servidores e módulos

*Node.Js:* um software open-source, cross-platform, e de um runtime de JavaScript que executa código de JavaScript a nível backend e frontend.

*Nodemailer*: Nodemailer é um módulo para aplicativos Node.js para permitir um envio de e-mail facilmente.

# 7.2.2 *Design Patterns* Aplicados

Um *desing* *pattern* é basicamente uma forma anteriormente validada que fornece, de maneira genérica, uma solução para problemas específicos.

1. DAO: é um padrão utilizado que representa o pacote que é composto pelas classes de persistência com o banco de dados. Essas classes são compostas de operações de inserção, leitura, alteração e remoção de elementos nas tabelas do banco de dados.
2. MVC: é um padrão arquitetural que possui como premissa básica fornecer uma divisão trilateral: *model*, *view* e *controller*. O *controller* é responsável por fazer a conexão entre o usuário e as regras de negócio contidas no *model*, assim, é por meio dele que as solicitações e requisições da *view* são transmitidas ao *model* e retornadas à *view*. O *model* é responsável por conter todas as regras de negócio da aplicação e, além disto, as classes de persistência com o sistema gerenciador de banco de dados. A *view* é a camada na qual o usuário interage diretamente com o sistema;
3. *Singleton*: especifica que apenas uma instância da classe pode existir, e esta será utilizada por toda a aplicação. Dessa forma temos apenas um ponto de acesso central a esta instância da classe.

# 7.2.3 Convenções e Guias para Codificação

A convenção utilizada para codificação do sistema de software *EasyJob*é a estabelecida pela *Ecma International*. Trata-se de boas práticas de codificação de software da linguagem *Javascript,* utilizada para facilitar o entendimento do código e aumentar a legibilidade do software.   A seguir algumas convenções adotadas:

Classes e interfaces: devem ser nomeadas por meio do método *Camel Case*, isto é, os substantivos devem estar juntos e a primeira letra de cada palavra deve ser maiúscula. Exemplo: “FreelancerService”;

Atributos e variáveis: para nomeá-los, é necessário que a primeira palavra comece com letra minúscula e as demais com letras maiúsculas.

Funções: devem ser nomeados por verbos, com a primeira letra da palavra inicial com letra minúscula e as demais com letras maiúsculas.

Identação: o alinhamento do código deve ser feito com quatro espaços do teclado (espaço correspondente à tecla tab).

Além dessas convenções também é utilizada a ferramenta *ESLint*, que é uma ferramenta plugável para *Javascript,* que permite configurar o editor para atender às regras de organização de código.

Basicamente, o ESLint, irá olhar o código e acusar os erros baseados na regra que foidefinida, ou que vem em uma regra de código popular, como a da AirBnb (pode ser encontrada no link - <https://github.com/airbnb/javascript>).

# 7.2.4 Estrutura Física do Banco de Dados

A estrutura física do banco de dados do projeto *EasyJob* se encontra no Apêndice R. No Apêndice R estão os *scripts* do *Sequelize* usados para a criação de todas as tabelas do banco de dados PostgreSQL.

# 8 PLANO DE TESTES

Neste capítulo são mostradas algumas das estratégias de testes adotadas neste projeto, qual sua finalidade, os artefatos testados, como os testes são executados e quais os resultados obtidos.

Segundo Bartié (2002), um plano de testes é um meio de formalizar uma das estratégias adotadas e assim contem o escopo de testes, a configuração de ambiente, os recursos necessários e o cronograma de execução.

Os erros e falhas no sistema de *software* podem influenciar, de forma negativa. Assim, com o objetivo de evitar que erros e falhas aconteçam, a equipe de desenvolvimento e testes devem adotar estratégias de testes.

## 8.1 FINALIDADE

O plano de testes aqui apresentado tem a finalidade testar algumas das funcionalidades do *EasyJob*.

## 8.2 ESCOPO

Esta seção apresenta todos os documentos e *softwares* necessários para a realização dos testes planejados.

**8.2.1 Referências aos Documentos Relevantes**

|  |  |
| --- | --- |
| **Tipo do material** | **Referência** |
| EasyJob-PFC.docx | Capitulo 5 - Especificação e análise dos requisitos. |

QUADRO 5 - Documento relevante para testes

**8.2.2 Ambiente para a Realização dos Testes**

O Quadro 6 mostra o equipamento necessário para a realização dos testes planejados.

|  |  |  |
| --- | --- | --- |
| **Equipamento** | **Marca/Modelo/Configuração** | **Finalidade** |
| Notebook | Processador Dell Core i5-7200U; memória RAM de 8 GB; Sistema Operacional Windows 10 Home 64 bits. | Executar os testes de integração e os testes caixa preta no componente Web. |

QUADRO 6 - Equipamentos para a realização dos testes.

O Quadro 7 mostra os *softwares* necessários para a realização dos testes planejados.

|  |  |  |
| --- | --- | --- |
| **Software/Versão** | **Fabricante** | **Finalidade** |
| Visual Studio Code 1.43.6 | Microsoft | Execução de *scripts* para realização dos casos de teste. |
| Mocha [8.1.3](https://github.com/mochajs/mocha/releases/tag/v8.1.3) | Open Source sob licença MIT | Execução de teste assíncronos em *Javascript* com [*Node.js*](https://nodejs.org/) e no navegador. |
| PostgreSQL 9.5 | PostgreSQL Global Development Group | Sistema gerenciador de banco de dados que fornece a base de dados para os componentes. |
| Google Chrome 85.0.4109.16 | Google LLC | Aplicação cliente que será utilizada para entrada e saída de dados. |

QUADRO 7 - Softwares para a realização dos testes

## 8.3 ESPECIFICAÇÃO DOS CASOS DE TESTES

Esta seção define o conjunto de casos de teste a serem realizados, os itens a serem testados, especificação das entradas e saídas esperadas, rastreabilidade entre requisitos e as descrições dos casos de testes.

### 8.3.1 Item a Testar

O Quadro 8 mostra os itens a serem testados e uma descrição de cada item.

|  |  |
| --- | --- |
| **Identificação do Item** | **Descrição** |
| 1 | RF01 – Realizar cadastro de *freelancer* |
| 2 | RF02 – Efetuar autenticação do *freelancer* |
| 3 | RF03 – Cadastrar estabelecimento comercial |
| 4 | RF04 – Alterar senha de acesso do usuário |
| 5 | RF05 – Remover conta de acesso do usuário |
| 6 | RF07 – Pesquisar serviços fornecidos |
| 7 | RF08 – Criar Anúncio de Serviço |

QUADRO 8 - Identificação dos itens a serem testados

### 8.3.2 Rastreabilidade entre Requisitos e Casos de Teste

Para cada item a ser testado, deve-se elaborar um conjunto de casos de teste que abordam todas, ou quase todas, as possibilidades dentro de um cenário. No Apêndice P encontra-se a tabela de mapeamento dos requisitos e seus respectivos casos de testes.

### 8.3.3 Descrição dos Casos de Teste

Um item a ser testado pode conter um ou mais casos de teste. Sendo que cada caso de teste aborda um possível cenário diferente de execução.

## 8.4 RESULTADOS DOS TESTES

Nesta seção são apresentados os resultados obtidos com a execução dos casos de teste previstos neste capítulo.

### 8.4.1 Histórico de Realização

No histórico de produção dos casos de teste encontra-se o nome do responsável pelo teste, a data de prática do teste e o resultado obtido. O Apêndice Q contém o histórico de realização de uma bateria dos testes aqui previstos.

# 9 IMPLANTAÇÃO

O front-end do sistema Easyjob foi implantado na Vercel e o back-end na Heroku,

# 10 CONCLUSÃO

Com a evolução das TICs, surgem aplicações de software que visam facilitar a vida das pessoas. Neste sentido, o *EasyJob* se propõe a ser uma delas. Inspirado em aplicativos semelhantes já existentes, foi constatada uma demanda específica para oferta e procura de vagas no setor de organização de eventos sociais, como *buffets* e shows.

Espera-se que o sistema *EasyJob* seja uma contribuição para os trabalhadores intermitentes e organizadores de eventos, pois será um intermediário entre o contratante e o fornecedor de serviços. Considerando que muitos organizadores de eventos não encontram bons profissionais para trabalhar em seus eventos, o aplicativo irá fornecer essa facilidade para eles, já os trabalhadores autônomos, que muitas vezes não encontram onde exercer sua profissão, poderão encontrar possíveis contratantes rapidamente.

Com todas as informações levantadas no desenvolvimento deste projeto, a equipe conseguiu entender melhor o funcionamento do mercado de eventos e de aplicativos de fornecimento de serviços. As entrevistas e pesquisas feitas mostraram como os possíveis usuários do sistema o utilizariam e quais as funcionalidades são mais relevantes para eles. A princípio a equipe conseguiu desenvolver 30% dos casos de uso essenciais, usando as tecnologias já apresentadas por este documento. Espera-se dar continuidade ao presente trabalho.

# REFERÊNCIAS

BRASIL. Lei nº. 13.467, de 13 de Julho de 2017. **Diário Oficial [da] União**, Brasília, DF, 14 Jul 2017.

ABÍLIO, Ludmila Costek. **Uberização do trabalho:** subsunção real da viração. Passa Palavra, 2017. Disponível em: <<http://passapalavra.info/2017/02/110685/>>. Acesso em: 22 mar. 2020.

ALONSO, Suelen. **Emprego informal**. Brasil Escola, 2020. Disponível em: <https://brasilescola.uol.com.br/geografia/empregos-informais.htm>. Acesso em: 25 mar. 2020.

BARTIÉ, A. **Garantia da Qualidade de Software**. 5. ed.  Elsevier, 2002.

BRASIL DE FATO. **De ambulante a motorista de aplicativo:** Brasil tem 39 milhões no trabalho informal. **Disponível em:** **<**<https://www.brasildefato.com.br/2019/05/03/de-ambulante-a-motorista-de-aplicativo-brasil-tem-39-milhoes-no-trabalho-informal> **>.** Acesso em: 25 mar. 2020.

ENDLER, D. **Dicas para Escrita de Casos de Teste.** Medium, 2018. Disponível em: <https://medium.com/cwi-software/dicas-para-escrita-de-casos-de-teste-ccea14a7fdd9>. Acesso em: 29 ago. 2020.

GET NINJAS. **GetNinjas**: o jeito mais esperto de se contratar um serviço. São Paulo: 2020. Disponível em: <<https://www.getninjas.com.br/>>. Acesso em 11 abril 2020.

GLOBO.COM. **Uso da internet no Brasil cresce, e 70% da população está conectada**. 2020. Disponível em: <<https://g1.globo.com/economia/tecnologia/noticia/2019/08/28/uso-da-internet-no-brasil-cresce-e-70percent-da-populacao-esta-conectada.ghtml>>. Acesso em: 23 mar. 2020.

AGENCIA BRASIL. **IBGE: desemprego na pandemia atinge maior patamar.** 2020. Disponível em < https://agenciabrasil.ebc.com.br/economia/noticia/2020-09/ibge-desemprego-na-pandemia-atinge-maior-patamar-em-agosto >. Acesso em 19/11/2020.

JOHNSON, R. E. Frameworks = (Components + Patterns). **Communications of the ACM**, v. 40, n. 10, outubro, 1997.

KARNER,G. **Resource Estimation for Objectory Projects.** Sweden: Objetive System, 1993.

ORGANIZAÇÃO DAS NAÇÕES UNIDAS (ONU). **Transformando Nosso Mundo**: A Agenda 2030 para o Desenvolvimento Sustentável. Nova York: [S. n.], 2015.

PRESSMAN, R. S. **Engenharia de Software**. 7. ed. São Paulo: McGraw Hill/Nacional, 2011.

IBGE. **Desemprego cai para 11,9% na média de 2019; informalidade é a maior em 4 anos**. Rio de Janeiro: 2020a. Disponível em: < <https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/26741-desemprego-cai-para-11-9-na-media-de-2019-informalidade-e-a-maior-em-4-anos>>. Acesso em: 11 abr. 2020.

\_\_\_\_**.Desemprego.** Rio de Janeiro, 2019. Disponível em: <<https://www.ibge.gov.br/explica/desemprego.php>>. Acesso em: 11 abr. 2020.

MYSTAFF. **Sobre o My Staff.** São Paulo: 2020, Disponível em: <<https://mystaffapp.com.br/>>. Acesso em 11 abr. 2020.

RIBEIRO, Renato. **Por que não podemos confundir trabalho informal com freelancing?**. Rockcontent, 2018. Disponível em: <<https://comunidade.rockcontent.com/trabalho-informal/>>. Acesso em: 25 mar. 2020.

SISTEMA BUFFET. Conheça algumas exigências legais para Buffet. 2017. Disponível em:

<https://www.sistemabuffet.com.br/gestao-de-buffets/conheca-algumas-exigencias-legais-para-buffets/> . Acesso em: 25 abr. 2020.

SOMMERVILLE, I. **Engenharia de Software**. 9. ed. São Paulo: Pearson, 2011.

SOMMERVILLE, I. **Engenharia de Software**. 7. ed. São Paulo: Pearson, 2007.

## 

# APÊNDICE A - PLANO DE ELABORAÇÃO E GERENCIAMENTO DO PROJETO

O plano de elaboração e gerenciamento do projeto está disponível na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE B – DIAGRAMAS DE CASOS DE USO

Os diagramas de casos de uso estão disponíveis na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE C – RELATÓRIO DE DESEMPENHO

O relatório de desempenho encontra-se disponível na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE D – DIAGRAMA DE ENTIDADE E RELACIONAMENTO

O diagrama de entidade-relacionamento está disponível na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE E – DESCRIÇÃO DOS CASOS DE USO

A descrição dos casos de uso está disponível na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE F - MODELO DE INTERFACE COM O USUÁRIO

A descrição do modelo de interface com o usuário está disponível na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE G - DIAGRAMA DE PACOTES

O diagrama de pacotes está disponível na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE H - DIAGRAMA DE ATIVIDADES

O diagrama de atividades está disponível na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE I - ENTREVISTAS

As entrevistas feitas estão disponíveis na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE J - RESULTADOS DO FORMULÁRIO

Os resultados do formulário estão disponíveis na pasta APÊNDICES do arquivo anexado a este documento.

# APÊNDICE K - MODELO LÓGICO

O modelo lógico encontra-se na pasta APÊNDICE K.

# APÊNDICE L - DICIONÁRIO DO MODELO LÓGICO

O dicionário de dados encontra-se na pasta APÊNDICE L.

# APÊNDICE M - DIAGRAMA DE CLASSES

O diagrama de classes encontra-se em anexo na pasta APÊNDICE M.

# APÊNDICE N - ESTIMATIVA POR CASO DE USO

O relatório de estimativa por caso de uso encontra-se em anexo na pasta APÊNDICE N.

# APÊNDICE O – DIAGRAMA DE SISTEMA DISTRIBUIDO

O diagrama de sistema distribuído encontra-se em anexo na pasta APÊNDICE O.

# APÊNDICE P – RASTREABILIDADE ENTRE REQUISITOS E CASOS DE TESTES

O relatório de casos de testes encontra-se em anexo na pasta APÊNDICE P.

# APÊNDICE Q – HISTÓRICO DE REALIZAÇÃO DE TESTES

O relatório de histórico de testes encontra-se em anexo na pasta APÊNDICE Q.

# APÊNDICE R – ESTRUTURA FÍSICA DO BANCO DE DADOS

A estrutura física do banco de dados encontra-se em anexo na pasta APÊNDICE R.

# APÊNDICE S – DIAGRAMA DE COMPONENTES

O diagrama de componentes encontra-se em anexo na pasta APÊNDICE S.

1. KARNER,G. **Resource Estimation for Objectory Projects.** Sweden: Objetive System, 1993. [↑](#footnote-ref-1)