INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO NORTE

JULIANA MARIA SILVA DE LYRA
PEDRO CASTRO DE FREITAS JÚNIOR
RANNY LAÍS DE LIMA OLIVEIRA

MAPA CARIDOSO

CEARÁ-MIRIM-RN
2017

JULIANA MARIA SILVA DE LYRA
PEDRO CASTRO DE FREITAS JÚNIOR
RANNY LAÍS DE LIMA OLIVEIRA

MAPA CARIDOSO

Relatório de Prática Profissional apresentado ao Curso Técnico Integrado em Informática do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, em cumprimento às exigências legais como requisito parcial para a obtenção do título de Técnico em Informática.

Orientador: Prof. Pedro Baesse A. Pereira

Ceará-Mirim-RN
2017

JULIANA MARIA SILVA DE LYRA
PEDRO CASTRO DE FREITAS JÚNIOR
RANNY LAÍS DE LIMA OLIVEIRA

MAPA CARIDOSO

Relatório de Prática Profissional apresentado ao Curso Técnico Integrado em Informática do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, em cumprimento às exigências legais como requisito parcial para a obtenção do título de Técnico em Informática.

Aprovado em:_/_/_____

Nota Final: _____

Prof. Pedro Baesse Alves Pereira - Presidente
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

Prof(a). Maíra de Faria Barros Medeiros, Membro da banca - Examinadora
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

Prof. Carlos Albuquerque, Membro da banca - Examinador
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

De forma a auxiliar instituições filantrópicas em suas ações após perceber que contemporaneamente estas empresas estão perdendo prestígio socialmente, e~~E~~ste trabalho tem por objetivo a criação de um sistema online, de cadastro, capaz de catalogar informações ~~(tais como endereço, tipo, nome)~~ referentes a ~~instituições filantrópicas~~empresas que desejem utilizar o serviço. O presente projeto visa auxiliar essas entidades que apresentem necessidades básicas, a fim de tornar o intermédio entre as mesmas e a informação a futuros auxiliadores mais simples~~curto~~. A partir disso, foi criado um sistema online de cadastro de instituições e doadores, em conjunto ~~a um~~com a API do Google Maps, mostrando as instituições e ~~seus~~suas ~~endereços~~localizações em um mapa, como também suas necessidades. Tem a finalidade de auxiliar essas instituições a suprir suas necessidades, informando-as ~~em~~no ~~nosso~~ sistema, com o propósito de os doadores cadastrados ~~em nosso site~~, como também visitantes, possam visualizá-las, diminuindo drasticamente o tempo que esse processo levaria ordinariamente. O projeto apresenta uma boa adequação à realidade empresarial, como também ao cotidiano, além de oferecer benefícios, como: praticidade, simplicidade, segurança e economia de tempo.

Palavras-Chave: Instituições. Doadores. Cadastro.

1 INTRODUÇÃO

Com e devido o advento da Revolução técnico-científica no mundo pós-guerra, de acordo com ~~a obra~~ “Infância Brasileira e contextos de desenvolvimento” (2002 EDUFBA), de Eulina da Rocha Lordelo, Ana Maria Almeida Carvalho e Sílvia Helena Kolleras, instituições filantrópicas ganharam força para reparar todo o estrago trazido por esse evento em um contexto mundial. Com o fim da II Guerra Mundial, após a queda do nazismo, o mundo trazia consigo marcas profundas provindas desse ~~evento~~momento. Muitos países distribuídos por todo o planeta estavam passando por dificuldades extremas de sobrevivência. Devido a isso, haviam ~~uma única~~ soluções restritas para intermediar toda a situação, sendo uma delas a caridade.~~o que intermediava toda a situação: a caridade.~~

O mundo pós-guerra teve que se reconstruir aos poucos, e a caridade pode ser considerada como o alicerce que firmou o reerguimento de vários países, como Japão, recordando as bombas atiradas pelos estadunidenses ~~a potência americana~~ nas cidades de Hiroshima e Nagasaki como forma de provar a superioridade norte-americana. Nessa época, as instituições de caridade e auxílio aos mais necessitados ganharam um grande foco e potência em busca de reparar os danos da guerra. Em vez cultivar o individualismo humano, relatado no livro ~~“~~Modernidade Líquida~~”~~ ~~, do (sociólogo Zygmunt~~ BAUMAN, Zygmunt~~auman)~~, onde relata o escritor que as relações sociais no mundo estão se tornando cada vez mais líquidas, fáceis de se desfazer, essas instituições fizeram com que a relação social coletiva, o uno, tornasse-se mais intensa, a fim de que todos auxiliassem a todos, formando uma corrente de solidariedade. Isso levou um processo de reerguimento drástico, e em pouco tempo um vilarejo, uma cidade, uma região, um país, estava erguido ~~(a)~~ novamente.

Com o passar do tempo, essas instituições sociais começaram a ganhar cada vez mais credibilidade, tornando-se uma ferramenta forte contra o sofrimento social pelas necessidades básicas, principalmente nas décadas de 80 e 90, com o crescimento do movimento Hippie. Entretanto, contemporaneamente observamos que essas entidades filantrópicas, que antes eram sinônimos de crescimento social, estão perdendo prestígio, devido ao crescimento do conceito de desenvolvimento enraizado desde a Guerra Fria que atualmente se destaca: o capitalismo. Em pleno

Século~~século~~ XXI percebemos a grande massa capitalista que está se instalando, vítima do inexorável desejo de consumo, gerando na sociedade e no mundo uma grande onda relativista, sendo seu principal fruto o individualismo. Vale ressaltar que o individualismo não é ruim, visto que é um conceito político, moral e social que exprime a afirmação e a liberdade do indivíduo frente a um grupo, à sociedade ou ao Estado. O individualismo se torna preocupante quando gera o egoísmo, acarretando geralmente em graves problemas.

A cultura nazista pode se adequar nesse contexto, uma vez que o conceito de raça gera/gerou divisão, preconceitos, elevação de minorias arianas e humilhação de massas judias. O egoísmo torna o outro inferior, insignificante, dispensável, desnecessário, como Kant com sua ~~frelata o filósofo Immanuel Kant com sua f~~ilosofia moral relata, afetando assim toda a massa populacional que hodiernamente passa por problemas psicológicos, como ~~—~~ idosos que vivem isolados em asilos sem nenhuma proteção ou bem-estar ~~—~~, problemas físicos, onde ~~—~~ ONG's diariamente recebem pacientes com as mais diversas fraturas, doenças, ~~—,~~ entre outros.

Em virtude disso, este projeto pretende ajudar qualquer instituição de caridade que esteja necessitando de algo para realizar suas atividades, através da informação do cadastro de seu endereço e de sua ~~urgência~~necessidade, a fim de que os doadores cadastrados e visitantes ~~do sistema~~ supram qualquer emergência que possa ser apresentada, proporcionando assim uma maior praticidade e segurança para o bom funcionamento dessas entidades filantrópicas.

1.1 JUSTIFICATIVA

Instituições filantrópicas, na atualidade, devem ser uma grande preocupação social em vista de seu ramo variado de atuação, seja com animais, com pessoas, entre outros. Em busca de um maior investimento em informação, a fim de que essas instituições possam abranger um maior raio, podendo suprir qualquer necessidade aparente, como a recuperação de um animal ferido, ~~,~~ O~~o~~ projeto une a área da informática, devido ser a mais atuante no convívio ordinário e empresarial, com o alicerce da caridade do mundo pós-guerra.

Nesse âmbito, o sistema foi planejado para tornar prática e rápida a propagação das informações sobre as necessidades de instituições de caridade, incluindo as diversas áreas de atuação, promovendo um grande conglomerado online de necessidades institucionais em um único sistema, com uma interface

amigável e clara para a instituição e o usuário que deseje utilizá-lo, com a finalidade de facilitar o intermédio entre doador e instituição.

1.2 OBJETIVOS

Objetivo Geral

Este projeto visa contribuir com entidades de caridade que estão necessitadas de doações, por meio de um sistema de cadastro de instituições e doadores, disponibilizando um mapa mostrando ~~todas~~suas as localidades e as respectivas carências das entidades, facilitando assim o intermédio entre instituição-doador.

Objetivos Específicos

- ~~Ter bom conhecimento da~~Estudar a tecnologia Flask de desenvolvimento web, acompanhado de JavaScript, HTML e CSS;
- Análise de requisitos do sistema;
- ~~Disponibilizar o sistema teste para uso local;~~
- ~~Tornar o sistema bem visível no município de Ceará-Mirim;~~
- ~~Ter uma quantidade significativa de instituições cadastradas no sistema;~~
- ~~Intensificar as informações das necessidades de cada entidade cadastrada;~~
- Implementar o API do Google Maps no sistema online;
- Disponibilizar o sistema para uso mundial;
- Receber sugestões para melhoria do sistema.

2 DADOS GERAIS DO PROJETO

TÍTULO DO PROJETO: Mapa Caridoso
PERÍODO DE REALIZAÇÃO: De Maio de 2017 a Dezembro de 2017

TOTAL DE HORAS: 340 horas.

2.1 SÍNTESE DE CARGA HORÁRIA E ATIVIDADES

Quadro 1 – Síntese de Carga horária e Atividades.

CARGA HORÁRIA	ATIVIDADES DESENVOLVIDAS
30 horas	Estudo do micro framework Flask, junto a linguagem Python.~~, com o auxílio de vídeo aulas disponibilizadas por utilizadores da rede social de vídeos Youtube com o assunto de Flask e Python.~~
60 horas	Construção de um sistema teste (intitulado 'microblog'),~~ disponibilizado por utilizadores da rede social de vídeos Youtube,~~ com os conhecimentos em Python e micro framework Flask com o auxílio do editor de texto gedit (Ubuntu).
30 horas	Estudo do software livre GIT.~~ por meio de vídeo aulas disponibilizadas na rede social de vídeos YouTube.~~
30 horas	Construção dos diagramas relacional, conceitual, caso de uso e de classe do projeto Mapa Caridoso e análise de requisitos;
30 horas	Início da produção do sistema Mapa Caridoso com a implementação das telas de login, telas de cadastros, junto à conexão com o Banco de Dados ~~SQLServer~~ pela construção das ~~Tables~~tabelas.
30~~40~~ horas	Continuação da produção do sistema Mapa Caridoso com a implementação do Mapa do Google Maps, ~~pelo API disponibilizado pela empresa~~por meio de

	API, retornando as informações inseridas nos cadastros das instituições de caridade no Banco de Dados. ~~SQLServer~~
40~~50~~ horas	Continuação da produção do sistema Mapa Caridoso com a implementação das telas e funcionalidades seguintes ao Login do Usuário, seja doador, instituição ou administrador definidas na análise de requisitos prévia.
30 horas	Continuação da produção do sistema 'Mapa Caridoso', com os conhecimentos em Python e micro framework Flask, de tela direcionada à administração do sistema pelo login do administrador de acordo com as funcionalidades definidas para cada personagem na avaliação de requisitos prévia.
10 horas	Estudo da plataforma online PythonAnyWhere.
25 horas	Revisão final das funcionalidades e telas do sistema pela análise de requisitos final e finalização das últimas alterações das telas e no banco de dados.
5 horas	Disponibilização do sistema na plataforma online PythonAnyWhere.
20 horas	Escrita do Relatório de prática profissional (RPP) referente a todo o processo descrito acima.

	Lembrete: Todas as finalizações das ações referidas nesse tópico foram atualizadas constantemente na plataforma GitHub.

3 METODOLOGIA

O sistema de cadastro de instituições de caridade e doadores utiliza ~~rá~~ conceitos de programação para internet, através das linguagens de programação Python e JavaScript e do micro framework Flask (para a parte do 'back-end'), da linguagem de marcação HTML (HyperText Markup Language) e da linguagem de folhas de estilo CSS (Cascading Style Sheets), com o auxílio do micro framework (para a parte do 'front-end'). Assim, de início, ~~ser~~foram ~~ão~~ modelados os diagramas conceitual, de caso de uso e de classes, que ~~farão~~ fazem o papel de abstrair todo o projeto do software. Ao finalizá-los, a implementação ~~será~~ foi iniciada, com a construção da tela inicial, junto às telas de cadastros, de login, e as telas de confirmações, através das validações feitas destas por meio dos formulários do micro framework Flask. Com isso, e com a validação confirmada, esses formulários retornam~~rão~~ os valores desses campos às funções do Flask encarregadas de direcionar essas informações ao banco de dados SQLAlchemy ~~'SQLServer'~~. Esses valores s~~ão~~o~~erão~~ salvos no banco de dados e utilizados no sistema.

Posteriormente, foram~~serão~~ criadas as telas advindas do login da Instituição, do Doador e do Administrador, sendo direcionadas às funções de cadastrar, visualizar e administrar os pedidos de doações cadastrados, os doadores e as instituições, respectivamente. Cada tela te~~rmá~~ funções específicas definidas na análise de requisitos prévia do sistema. Em seguida, foi implementado~~será implementado~~ a~~o~~ API do Google Maps no sistema, utilizando conceitos de JavaScript e o direcionamento disponível pelo próprio Google Maps.

Ademais, éserá utilizado os cadastros das instituições, mais especificamente os dados relacionados aos seus endereços, para junto ao Mapa do Google Mapsde um mapa, disponível pelo API do Google Maps, implementado no sistema, serem serem visualizadas cada instituição e seus pedidos de doações na interface gráfica. O doador e a instituição cadastrados, usuário visitante e administrador poderão visualizar o Mapa Caridoso, visto que haverá 4 mapas no sistema, na página inicial geral (index), na página inicial do login do Doador, da Instituição e do Administrador. Todas as informações referidas neste e nos parágrafos anteriores serão trabalhadas na plataforma GitHub a fim de que todos os programadores busquem uma melhor comunicação acerca do processo de produção do sistema.

Por fim, o sistema será revisado e modificado em busca da aceitação pela análise de requisitos, a fim de publicá-lo na plataforma online PythonAnyWhere e disponibilizá-lo para a usabilidade de todos que desejarem.

4 FUNDAMENTAÇÃO TEÓRICA

4.1 HTML (HYPERTEXT MARKUP LANGUAGE)

HTML é uma linguagem de marcação empregada na construção de páginas Web, podendo ser interpretada em navegadores usuais, como Internet Explorer, Mozilla, Google Chrome, Safari e Opera.

Qualquer documento HTML deve possuir marcadores para ser considerado dessa linguagem, tendo as ações de comandar a formatação da mesma. Cada marcador se apresenta entre parênteses angulares (‘<’ e ‘>’), chamado de ‘tag’, e traz consigo atributos, valores e outros elementos que facilitam a sua utilização. Cada atributo tem uma permissão para modificar os resultados de uma tag, como a cor e forma.

A representação HTML que é utilizada e vista normalmente entre programadores é representada na figura 1:

Figura 1: Código HTML ~~simples~~ mostrando título e parágrafo.

FONTE: *https://www.w3schools.com/html/img_notepad.png*

~~*Figura 2: Logo HTML informando a versão mais atualizada.*~~

~~FONTE: *http://www.codingdojo.com/blog/wp-content/uploads/html.png*~~

4.2 CSS (CASCADING STYLE SHEETS)

CSS é um mecanismo de formatação de estilo, em um documento HTML, de cores, fontes, espaçamentos, entre outros. Em vez de~~Ao invés de~~ formatar manualmente no documento HTML web o estilo das tags, o CSS cria um link com um arquivo que contém todos os estilos desejados, a fim de tornar essa ação automática no arquivo web. Sendo assim, em uma página web, com apenas um arquivo HTML, a utilização do CSS não causa muita diferença ao não usar. Entretanto, em um grande projeto web, com inúmeras páginas e arquivos HTML, o CSS é um mecanismo que torna o processo de produção mais prático e rápido, modificando o estilo das tags apenas no arquivo CSS em vez de buscar em cada página, uma por uma, de forma manual, os marcadores desejados.

Em suma, o CSS tem uma linguagem sintática simples, utilizando uma série

de palavras em inglês de forma a especificar os nomes de diferentes estilos de valores de um marcador. Encontra-se, na declaração, um seletor e um bloco de declaração, contendo uma propriedade e um valor, separados por dois pontos, e cada declaração é separada por um ponto e vírgula.

As informações anteriores são exemplificadas na figura 23:

Figura 23: Arquivo CSS de formatação de uma página HTML ~~simples~~.

FONTE: *http://csharpcorner.mindcrackerinc.netdna*

A tag 'body', presente no HTML, é formatada no arquivo CSS com os atributos de 'font-size', 'font-family', 'background-color' e 'color' (tamanho da fonte, família da fonte, cor do plano de fundo e cor, respectivamente). Esses atributos ficam entre chaves, tendo seus valores definidos após os dois pontos e separados por ponto e vírgula. Em geral, essa representação se adequa à exemplificação da linguagem de estilo.

~~*Figura 4: Logo CSS informando a versão mais atualizada.*~~

~~FONTE: *https://techchurian.files.wordpress.com/2013/02/css3-logo.png*~~

4.3 JAVASCRIPT

JavaScript é uma linguagem de programação interpretada originalmente implementada nos navegadores web usuais, a fim de que os scripts pudessem ser executados no lado do cliente sem a necessidade de passar pelo servidor. Ela foi concebida para ser uma linguagem script com orientação a objetos, baseada em protótipos, tipagem e funções de primeira classe. Em sua estrutura apresenta suporte à programação convencional, sendo considerada por esse e outros fatores, como recursos de fechamentos e funções de alta ordem comumente, a linguagem de programação mais utilizada no mundo.

Em páginas web HTML, o uso do JavaScript é essencialmente escrever funções que são incluídas nesses arquivos, interagindo com o DOM (Modelo de Objeto de Documentos). Muitas dessas funções servem para abrir uma nova janela com atributos de tamanho e posição, validar valores de um formulário HTML e mudar imagens à medida que o cursor do mouse se movimenta sobre elas, por exemplo.

A forma de construção de uma função JavaScript em um arquivo HTML está representada na figura 35:

Figura 35: Função 'alert()' retornando um 'Hello World' por JavaScript no arquivo HTML.

FONTE: *https://blog.udemy.com/wp-content/uploads/2013/10/JavascriptHelloWorld.png*

Vale ressaltar que a função JavaScript, de forma direta, encontra-se entre a tags '<script>' e '</script>' (tag referente a função Script) dentro das tags '<head>' e '</head>' no arquivo HTML.

Figura 6: Logo JavaScript.

FONTE: *https://www.reddit.com/r/javascript/comments/48eynl/udemy_offering_javascript_course_with_java_logo/*

4.4 PYTHON

Python é uma linguagem de programação de alto nível, dinâmica, forte, interativa, orientada a objetos, funcional, de tipagem, de script e interpretada. Hoje em dia possui um modelo de desenvolvimento para comunidade. Devido às suas características, sua utilização é mais voltada ao processamento de textos, dados científicos e criação de mecanismos para páginas web.

Grandes corporações e empresas utilizam essa linguagem de programação. O Google, o Yahoo! e a NASA são breves exemplos da relevância dessa linguagem. Além de grandes empresas, a linguagem também é utilizada bastante no uso da indústria da segurança da informação. Não resumindo a apenas esses exemplos, programas de edição de imagem, programas de edição tridimensional, sistemas operacionais, entre outros, utilizam a linguagem como um componente padrão, sendo disponível em diversas distribuições Linux.

Figura 7: Logo Python.

FONTE: https://upload.wikimedia.org/wikipedia/commons/thumb/c/c3/Python

4.5 FLASK (MICRO-FRAMEWORK)

Um framework é um conjunto de códigos que tem por objetivo resolver problemas de domínios específicos. Muitos frameworks trazem consigo funcionalidades específicas que ao utilizarmos em processos de sistemas podem facilitar de forma extraordinária toda a programação. O Flask é um micro framework python, ~~(~~um framework mais leve, mais simples, que traz consigo apenas o necessário para o desenvolvimento web, porém é estendível, podendo instalar mais recursos.~~) python~~ e é baseado na biblioteca WSGI Werkzeug.

O Flask tem a flexibilidade da linguagem de programação Python e ainda oferece um modelo simples de desenvolvimento, fazendo com que todo o processo do Mapa Caridoso se tornasse claro, prático e rápido. Como o Flask suporta extensões, as funcionalidades de abstração do banco de dados, validação de formulários, ou qualquer outro componente de biblioteca de terceiros, esse micro framework pode adicionar à aplicação final.

A forma de construção de um código Python através do micro framework Flask JavaScript está representada na figura 4 a seguir.

Figura ~~4~~8: Arquivo Flask, apresentando uma rota, retornando um 'Hello World' no navegado.

~~FONTE: *https://image.slideshareedn.com/apresentacaoflask-100922073255*~~

~~*Figura 9: Logo Flask.*~~

~~FONTE: *http://flask.pocoo.org/static/logo/flask.png*~~

4.6 BACK-END ~~E~~e FRONT-END

Na ciência da computação, as etapas iniciais e finais de um processo são claramente diferenciadas e claramente indistintas. Cada ciclo é separado em generalizações definidas nos termos ‘front-end’ e ‘back-end’. O front-end é o processo responsável por coletar qualquer entrada do usuário do sistema nas mais variadas formas possíveis e processá-la, a fim de adequá-la às especificações necessárias para que o back-end possa utilizá-la.

Normalmente, em projetos de software, onde o Mapa Caridoso se adequa, o modelo de visão controlada e oferecida pelo sistema fornece o front-end ao usuário; ao processar as informações passadas pelo usuário no front-end, o back-end entra em ação, com a forma de um banco de dados, por exemplo, salvando valores de formulários HTML, ou na ação de componentes de processamentos. Em linhas gerais, o lado front-end é qualquer parte manipulada pelo cliente, e o lado back-end reside no servidor. De forma clara, as definições de front-end e back-end podem ser definidas de acordo com a figura 5~~10~~:

Figura 5~~10~~: Simbologia da definição de Back-end e Front-end

FONTE: *https://www.nczonline.net/images/wp*

Nesse exemplo, o lado do navegador é o lado front-end, com todos os recursos possíveis para o bom processamento do usuário, como a utilização de linguagens de marcação e estilo (HTML e CSS respectivamente). O lado do servidor é o lado back-end, com linguagens de programação necessárias para o bom

processamento do sistema junto ao banco de dados, como Python, PHP, JavaScript, micro-framework Flask.

4.7 GIT

O Git é um software livre com ênfase em velocidade, inicialmente projetado para o desenvolvimento do Kernel Linux, porém foi mundialmente adotado em outros projetos. Em suma, sua atuação lida com diretórios, onde cada um é um repositório que apresenta um histórico completo das versões. Sua atuação é muito usual em projetos que lidam com um grande ramo de programadores, favorecendo uma comunicação mais prática e rápida entre os mesmos.

Figura 12: Logo Git

FONTE: *https://git-scm.com/images/logos/downloads*

4.8 GITHUB

O GitHub é uma plataforma que hospeda o código-fonte de um programa, controlando a versão, usando o Git, permitindo que os usuários dessa plataforma possam contribuir em projetos privados ou em projetos abertos (Open Source) em todo o mundo.

O GitHub é exacerbadamente utilizado por programadores para compartilhar seus trabalhos com outros programadores, a fim de que haja uma contribuição para tal, favorecendo também a comunicação devido aos recursos propostos pela plataforma GitHub. Ela trabalha por meio de repositórios, que são lugares onde esses códigos são devidamente remotos.

Naturalmente, a plataforma é utilizada por empresas mundialmente famosas, como a Google, Microsoft e WordPress, não se restringindo apenas a essas, sendo utilizada também por mais de 3 milhões de usuários ativos espalhados pelo mundo.

Figura 12: Logo GitHub

FONTE: *https://www.analyticsvidhya.com*

4.9 SQLAlchemy SERVER

SQLAlchemy é uma biblioteca de mapeamento objeto-relacional SQL em código aberto desenvolvido para a linguagem de programação Python.O Microsoft SQL Server é um sistema gerenciador de Banco de dados de forma relacional, que foi desenvolvido pela Microsoft.

Figura 13: Logo SQL Server

FONTE: *https://obieosobalu.files.wordpress*

4.10 PYTHONANYWHERE

O PythonAnyWhere é uma plataforma online focada para programadores que busca tornar seus WebApps em WebSites. Ele usa linguagem de programação Python e, porém também é capaz de utilizar uma aplicação feita em Flask e Django, com algumas alterações no arquivo WSGI. Sua utilização facilita a alocação na rede, pois permite fazer toda a parte estrutural da aplicação utilizando o menu proveniente da plataforma.

Figura 14: Logo PythonAnyWhere

~~FONTE: *http://ww1.prweb.com/prfiles/2012/05/18/9522833/PythonAnywhere*~~

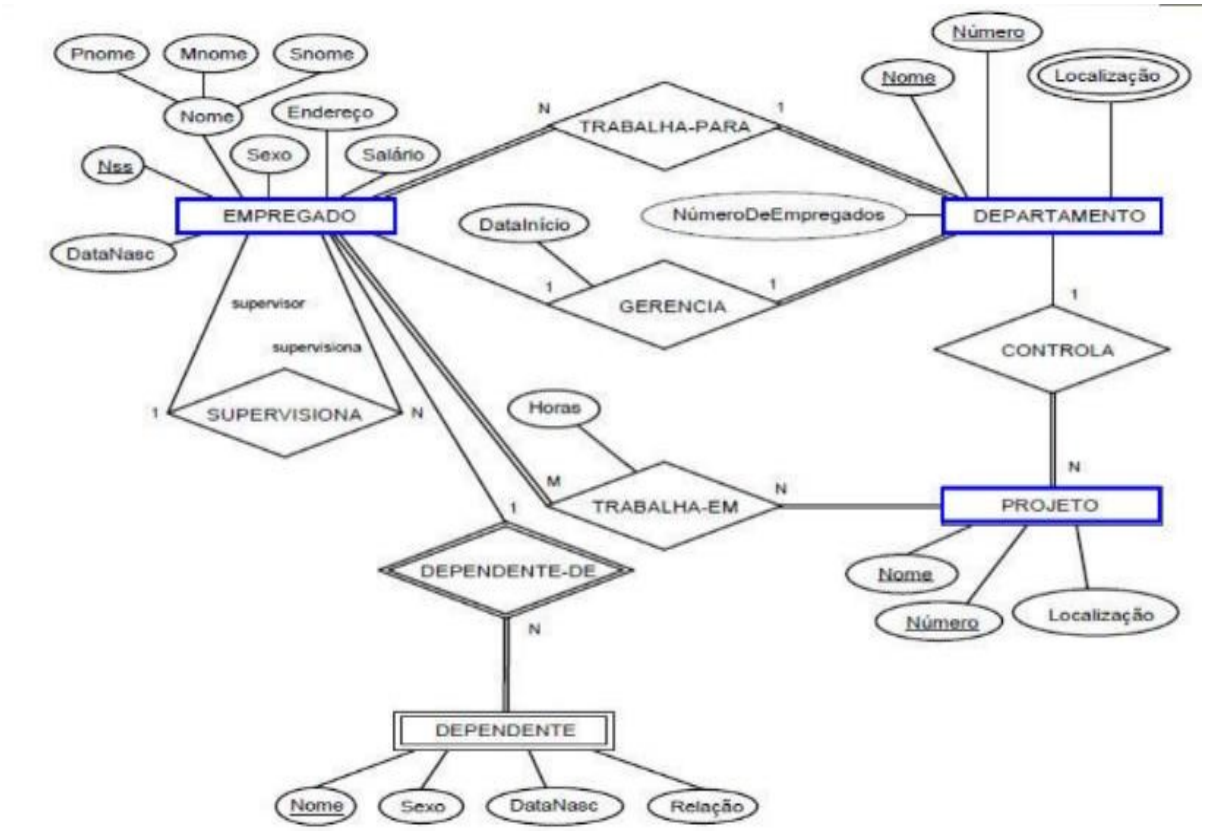
4.11 MODELAGEM DE DADOS

Modelar dados é simplesmente uma representação abstrata, por meio de um modelo prévio, que explicite, resumidamente, todas as características de funcionamento e comportamento de um projeto no qual será criado, facilitando assim o entendimento de todo o seu processo, através das características principais, evitando possíveis erros de funcionamento, projeção e programação. Por isso, pode ser considerada como uma das primeiras ações em um processo de construção de um software, e uma das mais importantes. Tecnicamente, modelos criados para representar abstratamente um projeto são denominados “diagramas”, como o diagrama conceitual, de caso de uso e de classes, que veremos a seguir.

4.11.1 Diagrama conceitual

O diagrama conceitual é um diagrama, formatado em blocos, que exprime todas as relações entre as entidades, seus atributos, suas especializações, autorrelaçõesauto relações, entre outros fatores. Também é considerado como uma descrição do banco de dados de forma independente de qualquer implementação prévia em um sistema de gerenciamento. Por isso, o conceitual trazconceitual trás consigo todos os dados que podem aparecer no banco de dados do projeto, entretanto não registra como será esse processo de armazenamento dos dados no sistema de gerenciamento do banco de dados. A figura 615 a seguir mostra a definição de Modelo Conceitual.

Figura 165: Exemplo simples de um Modelo Conceitual.

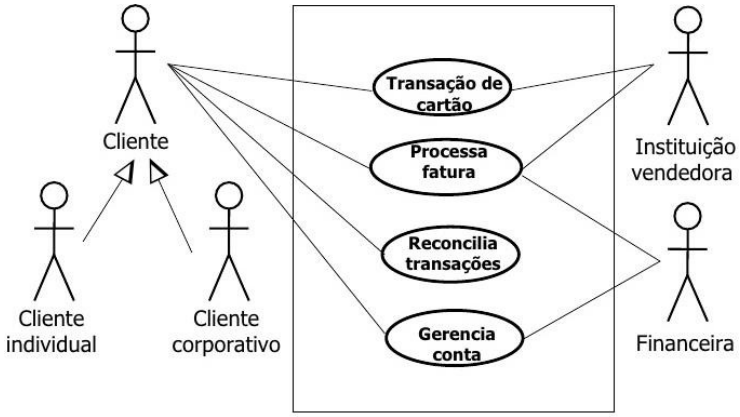


FONTE: *http://www.professor.pbaesse.net/wp-content/uploads/2013/11/Aula-4-Modelo- Conceitual-Entidade-Relacionamento-Parte-1-de-2.pdf*

4.11.2 Diagrama de caso de uso

Cada diagrama terá características únicas e essências para o processamento do projeto de software. O diagrama de caso de uso tem como objetivo auxiliar a comunicação entre os analistas do processo de software e o cliente que solicitou o objeto. Ele identifica os tipos de usuários que interagem com o sistema, os papéis que eles assumem e as funções requisitadas, por meio de um cenário que mostra as funcionalidades do projeto no ponto de vista do usuário. Além disso, é utilizado para identificar os requisitos do sistema, sendo usado também como base para criação de outros diagramas. Por fim, o diagrama de caso de uso deve exprimir as principais funcionalidades do sistema para o cliente. A figura 7~~16~~ exemplifica a definição do Diagrama de Caso de Uso.

Figura 7~~16~~: Exemplo simples de um Modelo de Caso de Uso.



FONTE: *https://pt.slideshare.net/DianaAdamatti/aula3-casos-de-uso*

4.11.3 Diagrama de classes

O Diagrama de classes é outro modelo extremamente importante para o processo de desenvolvimento de software, devido ser uma representação estática utilizada para descrever a estrutura de um sistema, apresentando suas classes, atributos, operações e as relações entre os objetos. Este tipo de representação é útil no desenvolvimento de sistemas e de softwares de computação, pois define todas as classes que o sistema precisa ter. A figura 8 ~~17~~ traz um exemplo da definição do Diagrama de Classe.

Figura ~~8~~17: Exemplo simples de um Diagrama de Classes.

FONTE: *https://www.significados.com.br/diagrama-de-classes/*

5 DESCRIÇÃO DO PROCESSO DO SISTEMA MAPA CARIDOSO

O processo para a realização do sistema Mapa Caridoso utilizou conceitos de programação para internet, através das linguagens de programação Python e JavaScript e do micro framework Flask (para a parte do 'back-end'), da linguagem de marcação HTML (HyperText Markup Language) e da linguagem de folhas de estilo CSS (Cascading Style Sheets), com o auxílio do micro framework (para a parte do 'front-end'). Assim sendo, foram feitos os diagramas conceitual, de caso de uso e de classes, abstraindo todo o processo de desenvolvimento de software. Todos os diagramas estão exibidos e descritos posteriormente.~~se encontram e foram comentados no final desta descrição.~~ Por continuidade, foram implementadas a tela inicial, junto às telas de cadastros, de login, e as telas de validação, feitas através de formulários Flask. Após isso, e com a confirmação da validação, esses formulários retornam os valores do campo às funções do micro framework Flask encarregadas de direcionar essas informações ao banco de dados ~~'SQLServer'~~. Esses valores são salvos no banco de dados e utilizados para o sistema.

Posteriormente, foram criadas as telas advindas do login da Instituição, do Doador e do Administrador, sendo realizadas as funções de cadastrar, visualizar e administrar os pedidos de doações cadastrados, os doadores e as instituições, respectivamente. Cada tela possui funções específicas definidas na análise de requisitos prévia do sistema. Em seguida, foi implementado o API do Google Maps no sistema, utilizando conceitos de JavaScript e o direcionamento disponível pelo próprio Google Maps.

Ademais, foram utilizados os cadastros das instituições, mais especificamente os dados relacionados aos seus endereços, para junto ao Mapa do Google Maps, disponível pelo API do Google Maps, implementado no sistema, serem visualizadas cada instituição e seus pedidos de doações na interface gráfica. O doador e a instituição cadastrados, o usuário visitante e o administrador podem visualizar o Mapa Caridoso, visto que há 4 mapas no sistema, um na página inicial geral (index), na página inicial do login do Doador, da Instituição e do Administrador. Todas as informações referidas neste e nos parágrafos anteriores foram trabalhadas na plataforma GitHub a fim de que todos os programadores buscassem uma melhor

comunicação acerca do processo de produção do sistema, e na plataforma Bootstrap, como base para muitos layouts presentes no mesmo.

Nesse âmbito e adentrando nas funcionalidades de cada usuário definida pela análise de requisitos, o que diferencia entre visitante, instituição, doador e Administrador são as funcionalidades. O visitante pode visualizar as instituições no mapa e fazer o cadastro de seu autor, seja instituição, ou doador ~~referente ao seu tipo~~; a instituição pode informar sua necessidade de doação e visualizar as avaliações feitas relacionadas~~feitas relacionada~~ às suas ações no sistema pelos usuários; já o doador pode visualizar essas doações, avaliar as instituições e denunciá-las, a fim de que o Administrador administre todas essas ações mencionadas neste parágrafo, como também excluir, atualizar e realizar cadastros advindos das mesmas.

Para o desenvolvimento do projeto, foram feitos três diagramas que serviram como auxílio~~auxilio~~ durante todo processo de criação do sistema, sendo eles: diagrama conceitual, diagrama de caso de uso e diagrama de classes. O diagrama conceitual está representado na figura 9.

Figura 9~~18~~: Diagrama Conceitual.

FONTE: *Elaborada pelos autores*

Primeiramente, o diagrama conceitual, mostrado acima, foi produzido retratando as 8 entidades que foram implementadas no projeto durante seu processo (‘Doador’, ‘Tipo_Doador’, ‘Denuncia’, ‘Doação’, ‘Avaliação’, ‘Instituição’, ‘Endereço_Instituição’, ‘Tipo_Instituição’), sendo fiéis~~ficis~~ à análise de requisitos prévia do sistema realizada antes da produção do modelo. Entre as entidades, foram retratados os principais relacionamentos que seriam essências

para o sistema online, de forma que todos esses relacionamentos trazem a maior quantidade possível das funcionalidades que estão presentes no projeto, facilitando assim o desenvolvimento do Software. Junto à entidade, foram incluídos seus atributos, relatando quais seriam salvos no Banco de Dados, entretanto esse modelo não traz consigo as ações que seriam feitas para salvar esses atributos no sistema geral do banco de dados, somente quais seriam. As entidades ‘Tipo_Doador’, ‘Tipo_Instituição’ e ‘Endereço_Instituição’ foram criadas para substituir os atributos multivalorados que estavam junto às entidades ‘Doador’ e ‘Instituição’, respectivamente. Esse Diagrama, que está representado na figura 10, foi essencial para o início da implementação do sistema, visto que o ‘Model’, do microframework Flask, favoreceu drasticamente a criação do banco de dados advindo das tabelas ~~‘Tables’~~ referentes ao diagrama citado.

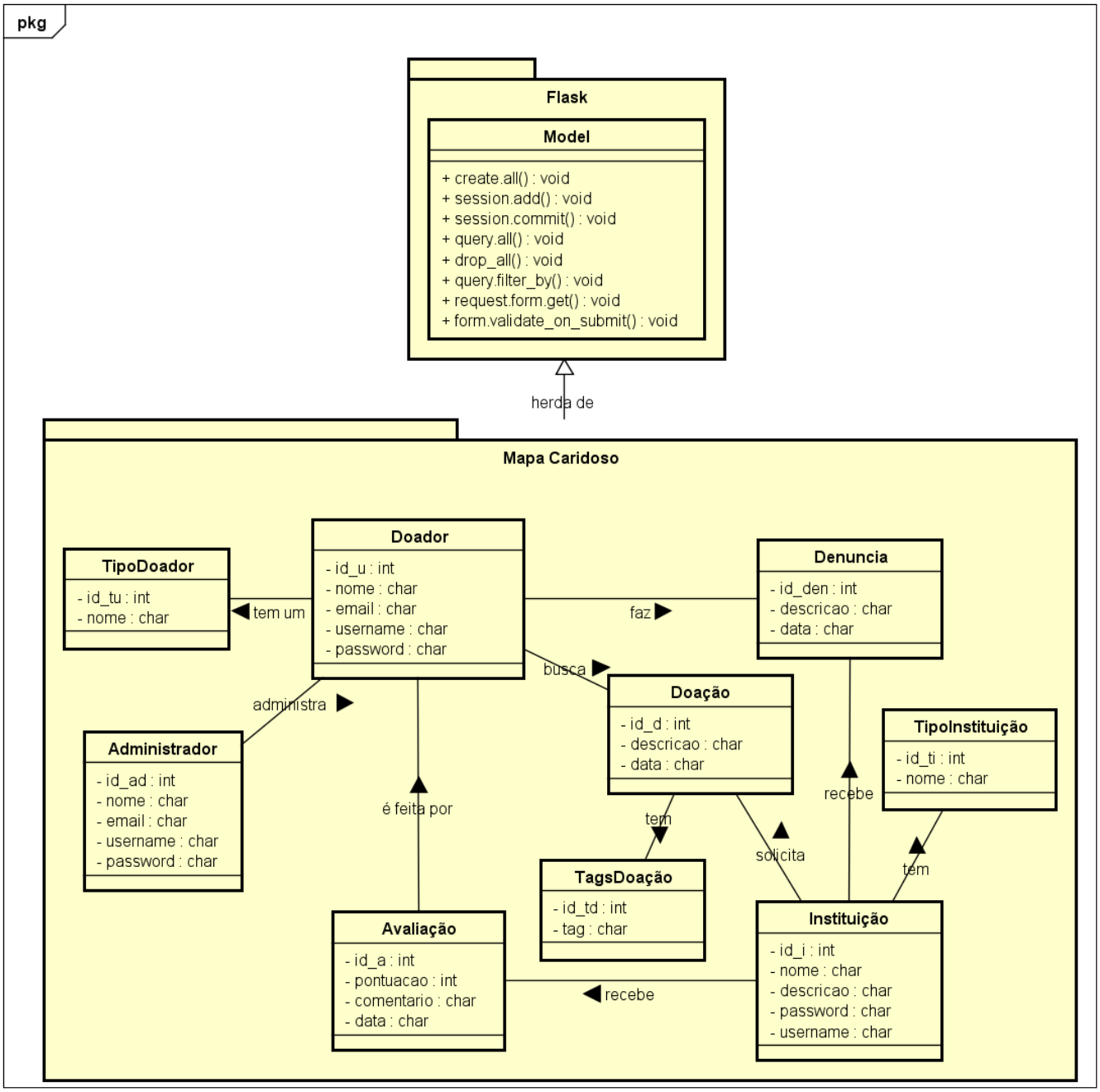
Figura 1~~9~~:0: Diagrama de Caso de Uso.

FONTE: *Elaborada pelos autores*

Em segundo lugar, podemos considerar que o diagrama de caso de uso, visto acima, foi o diagrama que melhor retratou e facilitou o processo de desenvolvimento de Software. Como sua utilização é voltada para o cliente do processo, ele traz consigo ~~as 4 pessoas~~ os 4 autores que ~~poderãopodem~~ utilizar o Mapa Caridoso. De início, o modelo mostra o visitante, que é a pessoa que terá o primeiro contato com o sistema online em serviço. Junto de sua representação, há 3 balões que relatam as 3 funcionalidades que terá para com o projeto, definidas na análise de requisitos. Posteriormente, o visitante, ao optar pelo 'cadastro de usuário' (uma de suas funcionalidades), remete a segunda pessoa que utilizará nosso projeto, o 'Doador'. O doador é uma pessoa que tem algumas funcionalidades a mais do que o visitante, pelo fato de ser uma pessoa cadastrada em nosso sistema. Vale ressaltar que todas essas funcionalidades foram listadas pela análise de requisitos prévia do sistema.

Acaso o visitante optar pelo ‘cadastro de instituição’ (outra de suas funcionalidades), remete à terceira pessoa que utilizará nosso sistema, a ‘Instituição’. Da mesma forma do doador, a instituição terá funcionalidades a mais comparadas às do visitante, sendo algumas diferentes às do doador. Por fim, o diagrama relata o ‘Administrador’ do sistema, quarta pessoa do projeto, capaz de inúmeras ações devido suas funcionalidades, por ser aquela que administrará todos os dados que serão armazenados no banco de dados pelo projeto. Seu cadastro não é feito diretamente no sistema online em serviço, mas sim pelos próprios desenvolvedores do projeto, direto no banco de dados do Mapa Caridoso.

Figura 119: Diagrama de Classes.



FONTE: *Elaborada pelos autores*

Em terceiro lugar, o diagrama de classes traz consigo todas as classes que foram implementadas no sistema. Devido a UML (Unified Modeling Language – linguagem de Modelagem Unificada, linguagem-padrão para a elaboração da estrutura de projetos de software) ser bem irrestrita com a modelagem, o diagrama de classes foi um diagrama que representamos como trabalhamos para com o micro framework Flask. Naturalmente, cada classe no modelo é retratada com suas funcionalidades, seus métodos, advindas da análise de requisitos. Em nosso diagrama, cada classe mostra seus atributos, mas não apresenta os métodos utilizados, pelo fato de que não precisamos criar nenhum. De forma mais clara e por

ter sido uma de nossas explicações para a escolha da utilização do Flask, o micro framework leva consigo, em utilizá-lo, os pré-definidos ‘Models’, que são os modelos que o micro framework transporta. Devido o framework já trazer consigo as ações pré-implementadas em toda sua dimensão, nós aproveitamos algumas para reimplementar em nosso projeto, fazendo com que cada classe herde diretamente seus métodos do ‘Model’ do Flask. Essa representação foi definida de forma mais clara no diagrama acima. Os métodos utilizados do Flask no projeto foram listados na representação do ‘Model’ no modelo. Cada seta no diagrama de classe representa que as classes em cada extremidade se relacionam diretamente.

Por fim, o sistema foi revisado e modificado em busca da aceitação pela análise de requisitos, e publicado na plataforma online PythonAnyWhere, disponibilizando para o uso ~~a usabilidade~~ de todos que desejarem.

6 CONSIDERAÇÕES FINAIS

O uso do sistema Mapa Caridoso ~~estar~~está ~~á~~ à disposição de toda a população, ficando a cargo dos indivíduos a escolha ou não pela utilização. ~~Através~~Por meio desse~~o~~ projeto, instituições de caridade informarão suas necessidades, a ponto de serem visualizadas no API do Google Maps, de forma que o intermédio, entre pessoas interessadas em auxiliar essas empresas, se torne mais rápido. Logo, proporciona maior praticidade, segurança, economia no tempo de buscas e conforto para o usuário.

O projeto t~~eve~~me ~~como~~ objetivo a implementação de um sistema online de cadastro de instituições filantrópicas e doadores em busca que as informações sobre as necessidades dessas empresas de caridade estivessem em uma plataforma propícia para tal, facilitando a dissipação da informação pela sociedade, concentrando-se em ~~apenas~~ um local mais centralizado. Através do uso do Mapa Caridoso, espera-se alcançar tal objetivo, tornando-se uma plataforma usual contemporaneamente. Nesse âmbito, percebe-se vantagens, como: maior nível de conforto, comodidade e praticidade para os usuários, reduzindo o tempo gasto pela busca na rede mundial.

6.1 TRABALHOS FUTUROS

Nosso projeto alcançou vários objetivos preambulares, mas há possíveis trabalhos ainda não desempenhados, dos quais podemos citar:

~~Conseguir um maior conhecimento do micro framework Flask para que seja possível retornar do banco de dados a localização das instituições, para que assim, automaticamente, um ponto a mais de localidade seja adicionado no mapa do sistema.~~

- Cadastrar instituições que possuem mais de um endereço, para facilitar o acesso de doadores que farãoirão fazer a entrega dos donativos pessoalmente, procurando o ponto mais perto de sua casa.
- Realizar doações financeiras através do sistema, para melhor comunicação entre o doador e a entidade, ampliando e abrangendo as formas de auxílio às instituições.
- Possibilitar a rentabilidade do sistema para os desenvolvedores. ~~Através do tópico anterior, poder atingir o ponto no qual o sistema se torne rentável para os desenvolvedores do mesmo.~~
- ~~Realizar um ranking entre os usuários cadastrados (doadores), no qual, o objetivo será a maior quantidade de doações possíveis. Sendo assim o doador poderá também cadastrar seus donativos, entregues pessoalmente, com a respectiva confirmação da contribuição pela entidade contemplada.~~

7 REFERÊNCIAS

Massote, Alysson. et al. ***Infância Brasileira e Contextos de Desenvolvimento****. 1 ed. São Paulo: Casa do Psicológico Livraria e Editora Ltda e 2002 EDUFBA, 2002.*

BAUMAN, Zygmunt. ***Modernidade Líquida****. ZAHAR. Disponível em: < https://farofafilosofica.com/2017/02/24/zygmunt-bauman-e>. Acesso em: 15 de Dezembro de 2017.*

FREITAS, Horácio. ***A filosofia Moral em Kant:*** *Introdução à filosofia moral em Kant. Disponível em: < http://filomoniz.blogs.sapo.pt/14052.html>. Acesso em: 15 de Dezembro de 2017.*

LOBO, Fábio Resende. ***O que é Front-End e Back-End****: Quem mexe com código é programador? Saiba mais sobre as profissões denominadas Front-End e Back-End - conheça as diferenças. Cruzeiro do Sul, RS. Disponível em: <https://www.fabiolobo.com.br/o-que-e-front-end-e-back-end.html>. Acesso em: 28 de nov de 2017.*

W3SCHOOLS. ***JavaScript HTML DOM - Changing CSS****. Disponível em: <https://www.w3schools.com/js/js_htmldom_css.asp>. Acesso em: 28 de nov de 2017.*

W3SCHOOLS. ***HTML5 Tutorial****. Disponível em: <https://www.w3schools.com/html/default.asp>. Acesso em: 28 de nov de 2017.*

W3SCHOOLS. ***CSS Tutorial****. Disponível em: <https://www.w3schools.com/css/default.asp>. Acesso em: 28 de nov de 2017.*

W3SCHOOLS. ***SQL Tutorial****. Disponível em: <https://www.w3schools.com/sql/default.asp>. Acesso em: 28 de nov de 2017.*

TOMAS, Giles; JONES, Glenn; PERCIVAL, Harry; HO, Conrad. ***Python coding on the web: 7,169,530 consoles served!****. Disponível em: <https://www.pythonanywhere.com/>. Acesso em: 28 de nov de 2017.*

RONACHER, Armin. ***Flask is a microframework for Python based on Werkzeug, Jinja 2 and good intentions. And before you ask: It's BSD licensed!.*** *Disponível em: <http://flask.pocoo.org/>. Acesso em: 28 de nov de 2017.*

HIDAYAT, Ariya; KEITH-MAGEE, Russel; MALCHIMP; NOVA, Kris; YOU, Evan; MAPBOX; FRAZELLE, Jess. ***A better way to work together:*** *GitHub brings teams together to work through problems, move ideas forward, and learn from each other along the way. Disponível em: < https://github.com/>. Acesso em: 28 de nov de 2017.*