Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

WELLINGTON CASSIANO MONTEIRO DIAS

APLICAÇÃO WEB NA INFRAESTRUTURA DO GOOGLE UTILIZANDO GOOGLE APP ENGINE

Assis
2012

Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

WELLINGTON CASSIANO MONTEIRO DIAS

APLICAÇÃO WEB NA INFRAESTRUTURA DO GOOGLE UTILIZANDO GOOGLE APP ENGINE

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Graduação.

Orientador: Fernando Cesar de Lima.

Área de Concentração: Informática.

Assis
2012

Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

FICHA CATALOGRÁFICA

DIAS, Wellington Cassiano Monteiro

Aplicação Web na Infraestrutura do Google Utilizando Google App Engine / Wellington Cassiano Monteiro Dias. Fundação Educacional do Município de Assis – FEMA – Assis, 2012.

46p.

Orientador: Fernando Cesar de Lima.

Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de Assis – IMESA.

1.Google App Engine. 2.Computação em Nuvem. 3.Java.

CDD: 001.6
Biblioteca da FEMA

Fundação Educacional do Município de Assis
Instituto Municípal de Ensino Superior de Assis - IMESA

APLICAÇÃO WEB NA INFRAESTRUTURA DO GOOGLE UTILIZANDO GOOGLE APP ENGINE

WELLINGTON CASSIANO MONTEIRO DIAS

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Bacharelado em Ciência da Computação, analisado pela seguinte comissão examinadora:

Orientador: Fernando Cesar de Lima.

Analisador: Prof. Dr. Alex Sandro Romeo de Souza Poletto.

Assis
2012

DEDICATÓRIA

Dedico este trabalho, ao meu pai Luiz Carlos Dias, minha mãe Vera de Jesus Dias, por todo seu apoio e incentivo, e pelos seus importantes ensinamentos.

AGRADECIMENTOS

Agradeço a Deus, por me dar forças para não desistir durante a caminhada para conclusão deste curso.

A minha família, que sempre me apoiou durante toda minha vida, e não diferente antes mesmo da escolha do curso.

Ao meu orientador, Fernando Cesar de Lima, pela orientação e todo apoio e incentivo para realização deste trabalho.

A Fundação Educacional do Município de Assis, por toda estrutura prestada durante todos esses anos e tornado capaz a realização deste curso.

A todos os meus professores, pelos importantes conhecimentos adquiridos durante a minha formação.

Aos amigos, que sempre me apoiaram e estiveram comigo, mesmo nos momentos difíceis.

Nunca deixe que lhe digam que não vale a pena
Acreditar nos sonhos que se tem
Ou que seus planos nunca vão dar certo
Ou que você nunca vai ser alguém

Renato Russo
(1960-1996)

RESUMO

A tecnologia está sempre inovando a forma como trabalhar em diversos seguimentos, como não poderia faltar também inovou a forma de distribuição de *software*, utilizando o que chama de computação em nuvem. Atualmente empresas disponibilizam vários tipos de recursos pela *web*. O número de usuários utilizando esses recursos é cada vez maior devido ao avanço tecnológico e o fácil acesso.

A empresa Google, que é conhecida pelo sistema de busca na *web*, também disponibiliza várias outras ferramentas, uma delas é Google *App Engine (*GAE*)*, que permite a implantação de aplicativos na infraestrutura do Google.

Este trabalho apresenta um estudo de caso envolvendo uma aplicação *web* de compra e venda de veículos. A aplicação *web* utiliza o GAE, para disponibilizar a aplicação em nuvem, tendo em vista que ela estará disponível para qualquer usuário que tenha acesso a *Internet*.

Palavras Chave: Google *App Engine*; Computação em Nuvem; Java.

ABSTRACT

The technology is always innovating the way to work in different segments, such as could not miss also innovated the form of software distribution, using what he calls cloud computing, companies currently offer several types of resources on the web. The number of users using these resources is increasing due to technological advancement and easy access.

The Google company, which is known by the system of web search, also provides several other tools, one of them is Google App Engine (GAE), which allows the deployment of applications on Google's infrastructure.

This work presents a case study involving a web application for buying and selling vehicles. The web application uses the GAE to provide the application cloud in order that it will be available to any user who has access to internet.

Keywords: Google App Engine; Cloud Computing; Java.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
GAE	Google *App Engine*
IAAS	*Infraestruture as a Service*
IDE	*Integrated Development Environment*
JAR	Java *Archive*
JDO	Java *Data Object*
JPA	Java *Persistence* API
JSP	Java *Server Page*
JVM	Java Virtual *Machine*
PAAS	*Platform as a Service*
POJOs	*Plain Old* Java *Objects*
POO	Programação Orientada a Objeto
SAAS	Software *as a Service*
SDK	Software *Development Kit*
UML	*Unified Modeling Language*
XML	*Extensible Markup Language*
WAR	*Web Application Archive*

SUMÁRIO

1. INTRODUÇÃO

Na atualidade muito se fala no termo *Cloud Computing,* ou seja, Computação em Nuvem, que é a ideia de utilização de recursos computacionais pela *web*, sem saber ao certo sua localização física, por isso a ideia de nuvem, possui vantagens de não ter a preocupação com *hardware* e nem localização física dos aplicativos. De acordo com (TAURION, 2009) é uma ideia extremamente sedutora: utilizar os recursos ociosos de computadores independentes, sem preocupação com localização física e sem investimentos em *hardware*.

A Google, empresa que é muito conhecida pelo seu sistema de busca na *web*, e por disponibilizar diversas Interfaces de Programação de Aplicativos (API – *Application Programming Interface*), também oferece o Google *App Engine* (GAE), uma Plataforma como Serviço (PaaS – *Platform as a Service*), que são recursos de *hardware* e *software*, oferecidos ao usuário desenvolvedor de aplicações, de forma a executar e disponibilizar suas aplicações em nuvem.

Com o GAE é possível executar suas aplicações na infraestrutura da Google. Ela oferece suporte a diversas linguagens de programação como Java*, Python, Go*, entre outras. Em seu ambiente de execução Java, é fornecido uma Maquina Virtual Java 6 (JVM – *Java Virtual Machine*), uma interface de Java *Servlets*, e suporte para interfaces padrão para o armazenamento de dados (GOOGLE, 2012).

A linguagem de programação Java apresenta uma série de características que a torna muito atrativa para o desenvolvimento de aplicações. Essas características abrangem a orientação a objetos, independência da plataforma, robustez, segurança, confiabilidade e conectividade com a *web* (LEMA; PERKINS, 1996 e DEITEL, 2003). Isto proporciona o desenvolvimento de projetos com maior qualidade, portabilidade e com menor tempo de desenvolvimento.

Foi implementado uma aplicação *web* com o intuito de mostrar seu funcionamento no GAE. As PaaS tem se tornado uma ótima forma de economia no contexto de gastos com *hardware* para manter a aplicação sempre funcionando.

1.1 OBJETIVOS

O objetivo é desenvolver uma aplicação de compra e venda de veiculos usando a linguagem *Java* para mostrar o funcionamento do GAE. A aplicação *web* em Java foi desenvolvida seguindo os conceitos para se utilizar o GAE, já que não será necessário a utilização de banco de dados comuns. A ferramenta do Google utiliza seu próprio banco de dados para armazenar suas informações, com isso foi necessário à utilização de JPA (*Java Persistence API*) que é uma forma de persistência, usada para mapear os objetos Java comuns (POJOs – *Plain Old Java Objects*).

1.2 JUSTIFICATIVAS

O desenvolvimento deste projeto de pesquisa se justifica, tendo em vista, que com o passar dos anos muito se fala em Computação em Nuvem, e sempre se busca facilidades, como implantação e manutenção de aplicativos sem a preocupação com *hardware* e local físico para servir as aplicações.

Para uma aplicação rodar em diversos computadores ao mesmo tempo é necessária uma boa infraestrutura para atender a demanda, e toda vez que aumenta o número de cliente, deve-se melhorar os recursos necessários para continuar com qualidade de serviço, mas utilizando GAE o desenvolvedor deve se preocupar apenas no desenvolvimento do aplicativo, a administração dos recursos é mais simples, podendo aumentar a capacidade de forma dinâmica usando somente o necessário.

Devido à dificuldade em manter o aplicativo sempre com a mesma qualidade, mesmo com aumento no número de usuários é que se justifica a utilização do GAE.

1.3 MOTIVAÇÕES

A motivação para a escolha desse trabalho é, usar a plataforma do GAE para mostrar uma forma de utilização de recursos em nuvem para rodar aplicações, já que nos dias atuais muito se fala em ferramentas da Google e em *Cloud Computing*.

1.4 ESTRUTURA DO TRABALHO

Este trabalho está dividido em quatro capítulos, são eles:

1. INTRODUÇÃO: Apresenta a proposta de trabalho objetivos, justificativa e motivações.
2. FUNDAMENTAÇÃO TEÓRICA: Apresenta as ferramentas e linguagens utilizadas e todo conhecimento básico necessário para desenvolvimento do aplicativo.
3. DESENVOLVIMENTO DO APLICATIVO: Apresenta a modelagem a implementação e a implantação do aplicativo.
4. CONCLUSÃO: Apresenta as considerações finais e os trabalhos futuros.

2. FUNDAMENTAÇÃO TEÓRICA

Para o desenvolvimento desse aplicativo são necessários alguns fundamentos teóricos. Neste capítulo serão apresentados resumidamente os conceitos sobre as tecnologias utilizadas.

2.1 CLOUD COMPUTING

Cloud Computing ou Computação em Nuvem o termo traduzido para o português, refere-se à utilização de recursos providos da *internet*, sem saber ao certo o sua localização exata, por esse motivo é dito que os recursos então nas nuvens.

É grande o seu interesse por diversos fatores, como a utilização de recursos da nuvem ou até mesmo executar aplicações diretamente na *web*.

Com o aumento da banda larga, *internet* mais rápida, a computação em nuvem cresce junto, por ser de fácil acesso aos recursos disponibilizados por diversas empresas na *web*.

Segundo (TAURION, 2009), pode-se dizer que a Computação em Nuvem é um conjunto de recursos com capacidade de processamento, armazenamento, conectividade, plataformas, aplicações e serviços disponibilizados na *internet*.

A Figura 1 mostra o acesso à nuvem por diferentes dispositivos, já que é necessário apenas ter acesso à *internet*.

Figura 1 – Visão geral de uma nuvem computacional (SOUSA, MOREIRA e MACHADO, 2009)

Existem três modelos de serviços de computação em nuvem, como mostra a Figura 2.

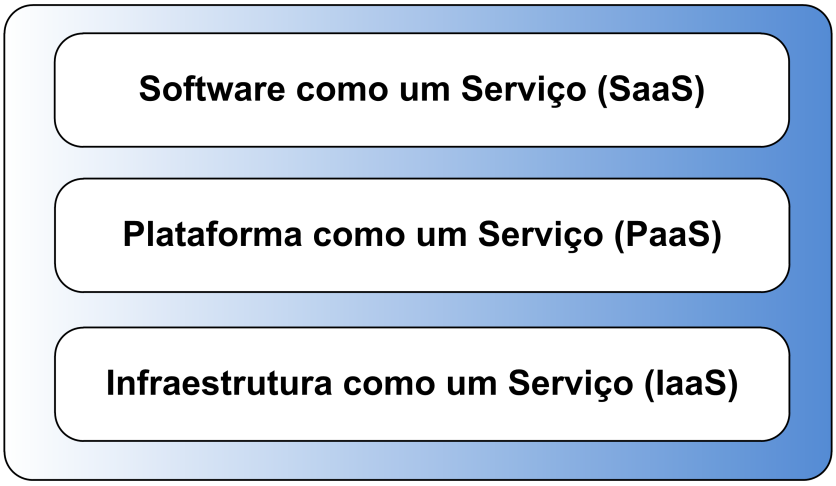

Figura 2 – Modelos de Serviços (SOUSA, MOREIRA e MACHADO, 2009)

2.1.1 IaaS

Infraestructure as a Service ou Infraestrutura como Serviço, é a primeira camada onde o provedor oferece de forma transparente seu armazenamento, processamento e outros recursos.

Segundo (SOUSA, MOREIRA e MACHADO, 2009) o IaaS é a parte responsável por prover toda a infraestrutura necessária para a PaaS e o SaaS.

Exemplos de IaaS:

- Eucalyptus: Projeto de código aberto que permite a criação de uma infraestrutura para uso em nuvem.
- Amazon EC2: Ambiente para desenvolvimento que permite que usuários tenham controle total aos recursos computacionais.

2.1.2 PaaS

Plataform as a Service ou Plataforma como Serviço, é a segunda camada, consiste de uma infraestrutura para implementar aplicativos em nuvem. A administração de sistemas operacionais, redes e servidores não são feitos pelo usuário, no entanto pode controlar os aplicativos implantados. O GAE é um exemplo de PaaS.

2.1.3 SaaS

Software as a Service ou Software como Serviço é a camada mais alta, é quando empresas disponibilizam programas para usar pela *web*, como exemplos têm o *Google Docs*, que oferece algumas funções do programa *Microsoft Office*, assim é possível visualizar, por exemplo, um documento com extensão .doc, sem ao menos ter instalado o programa processador de texto em seu PC.

Ao utilizar o *software* pela *web* o usuário não tem que se preocupar com instalação, manutenção e *upgrade* tudo fica a cargo da empresa que oferece o *software* seja ele pago ou não.

Possui vantagem de pagar apenas uma licença e utilizar em qualquer PC que se conecte a *Internet*, e no caso de assinatura mensal ser de fácil controle de acesso, pagando apenas o que usa.

Com o SaaS o cliente não adquire o produto em si, apenas utiliza as funcionalidades acessando de qualquer *browser*. Alguns exemplos de Saas na Figura 3.

Figura 3 – Exemplos de SaaS, Google Apps (GOOGLE, 2012)

2.2 LINGUAGEM JAVA

O projeto da linguagem Java iniciou-se em 1991 pela *Sun Microsystems*, baseando-se nas linguagens existente C e C++, o projeto passava por dificuldades, pois o mercado de dispositivos eletrônicos inteligentes não estava evoluindo como o esperado, o que alavancou a linguagem foi que em 1993 a popularidade da *World Wide Web* teve um alto crescimento, que levou a *Sun* utilizar Java na criação de páginas da *Web* com conteúdo dinâmico (DEITEL, 2003).

A Programação Orientada Objetos (POO) é muito utilizada nos dias de hoje, e uma linguagem que se destaca é o Java, por sua portabilidade, facilidade de programação, eficiência e por ser uma tecnologia gratuita.

A plataforma Java é constituída de um grande número de tecnologias, cada uma provê uma porção distinta de todo o ambiente de desenvolvimento e execução de *software*. Este conjunto de tecnologias é composto por três plataformas principais, conforme (LUCKOW; MELO, 2010):

- ***Java Standard Edition*** (Java SE), base da plataforma, onde inclui o ambiente de execução e as bibliotecas comuns;

- ***Java Enterprise Edition*** (Java EE), voltada para o desenvolvimento de aplicações corporativas e para *internet*.
- ***Java Micro Edition*** (Java ME), voltada para o desenvolvimento de aplicações para dispositivos móveis e embarcado.

A grande vantagem da plataforma é a de não estar presa a um único sistema operacional ou *hardware*, pois seus programas rodam através de uma máquina virtual que pode ser usada em qualquer sistema que suporte a linguagem C++.

Neste projeto será utilizada JEE, pois ela contém bibliotecas que fornecem funcionalidade para implementar *software* Java distribuído, tolerante a falhas e multi-camada, baseada amplamente em componentes modulares executando em um servidor de aplicações. Por essas características, o JEE foi desenvolvido para suportar uma grande quantidade de usuários simultâneos. De acordo com (PERRONE, 2003; GONÇALVES, 2009), entre essas funcionalidades tem-se JSP e Servlets:

- **JSP** (*Java Server Pages*) são páginas Java embebidas em HTML. Desta forma a página dinâmica é gerada pelo código JSP (GONÇALVES, 2007).
- ***Servlets*** são classes Java, desenvolvidas de acordo com uma estrutura bem definida, e que, quando instaladas junto a um servidor que implemente um servlet container, podem tratar requisições recebidas de clientes (GONÇALVES, 2007).

2.3 JAVASCRIPT

JavaScript é uma linguagem de programação leve, interpretada e com recursos de orientação a objetos (FLANANGAN, 2002). Seus scripts são utilizados em páginas HTML para torná-las mais dinâmicas. Apesar de algumas palavras reservadas serem iguais ao do Java, JavaScript não é Java.

Segundo (FLANANGAN, 2002) JavaScript é uma linguagem de programação completa com todos os recursos, tão complexa quanto qualquer uma e mais complexa do que algumas.

2.4 MVC

Model View Controller (MVC) é uma arquitetura usada para dividir a programação em três camadas, tornando mais fácil o desenvolvimento e manutenção dos aplicativos.

Segundo (SAMPAIO, 2003) MVC é uma estratégia de separação de camadas de software que visa desacoplar a interface de seu tratamento e de seu estado. Suas camadas são:

- **Model** (Modelo): É responsável por alterar o estado e fornecer à View os valores atuais.
- **View** (Visualização): responsável pela apresentação. Captura ações do usuário e comunica ao Controller para que sejam executadas.
- **Controller** (Controlador): Recebe solicitações da View, invoca as regras de negócio necessárias, comanda a alteração do estado do Model.

O Servlet fica responsável pela lógica de negócio e o JSP responsável pela apresentação dos resultados do Servlet.

2.5 XML

A linguagem de marcação XML surgiu da crescente demanda de extensões, bem como da necessidade de interoperabilidade, escalabilidade e flexibilidade. Um programa que utiliza XML compreende a descrição de dados, tornando-se possível seu processamento por uma aplicação. XML tem sido cada vez mais utilizada por

desenvolvedores de aplicações devido ao suporte que ela oferece tanto a interoperabilidade quanto funcionalidade da *web* (MOULTIS e KIRK, 2000). A simplicidade da XML faz com que ela seja acessível a todos os níveis de programadores, tornando-a cada vez mais difundida e utilizada.

A XML é linguagem de marcação que utiliza *tags* para delimitação das informações. Ela foi desenvolvida para ser utilizada em marcações de documentos. Isso facilita declarações mais precisas do conteúdo e resultados mais significativos de busca por meio de múltiplas plataformas. O XML também vai permitir o surgimento de uma nova geração de aplicações de manipulação e visualização de dados via internet (ANDERSON et al., 2001).

Como a XML não possui um conjunto pré-definido de *tags* ou elementos, ela é considerada uma meta-linguagem para descrição de linguagens de marcação. Desta forma, num documento XML as *tags* são usadas para definir o significado dos dados, podem ser definidos livremente de acordo com o domínio dos dados e da aplicação (DAUM e MERTEN, 2002).

A estrutura de um documento XML é formada por pares de *tags* ou elementos que indicam o começo e o fim do documento, e entre as *tags* as informações. A Figura 4 mostra uma estrutura básica de um documento XML onde <pais> </pais> é o elemento pai, e as *tags* que estão entre ela são os elementos filhos.

```
<pais>
    <codigo>1</codigo>
    <nome>Brasil</nome>
        <estado>
            <codigo>1</codigo>
            <nome>Sao Paulo</nome>
                <cidade>
                    <codigo>1</codigo>
                    <nome>Assis</nome>
                </cidade>
        </estado>
</pais>
```

Figura 4 – Estrutura de Documento XML

Um documento XML pode ter a organização de seus elementos definida basicamente de duas formas, conforme (BRANDLEY, 2002):

- **DTD** (*Document Type Definition*) é um artefato opcional para se especificar formalmente um conjunto de regras para definição da estrutura de documentos XML. Estas regras definem quais elementos podem ser usados e definem onde eles podem ser aplicados em relação a cada outro elemento. Além disso, especifica a hierarquia e a granularidade do documento;

- **XML Schema**, consiste num conjunto de regras com o objetivo de padronizar a estrutura de um documento XML bem formado, porém com algumas facilidades a mais como, por exemplo, o suporte a tipos de dados e extensão de tipos.

2.6 HTML

A linguagem HTML (*HiperText Markup Language)* é usada no desenvolvimento de páginas *web*, ela pode ser interpretada pelo *Browser*, se destaca por ser de fácil manipulação e estrutura.

Segundo (RAMALHO, 1996) HTML é uma linguagem especializada, dedicada à exibição e acesso de páginas *Web*. Possui tags como: <home>, <body>, entre outros, que são os comandos da linguagem HTML.

2.7 CSS

Cascading Style Sheets (CSS) ou folha de estilos é usada na criação de estilos de páginas *web*, utilizar CSS facilita na construção e manutenção do layout da página, podendo mudar o estilo de todas as páginas do site alterando o estilo a elas relacionado, estilos como: alinhamento de texto, margens, cor de texto, tamanho da fonte, etc.

O CSS não é uma linguagem de programação nem uma linguagem de marcação. É uma sequencia de declarações de propriedades, e seus valores para a manipulação da forma como o conteúdo de uma página *web* será exibido (GOMES, 2010).

2.8 BIGTABLE

O Google possui sua própria forma de armazenamento de dados, não convencional, ou seja, orientado a colunas e não os comumente bancos relacionais.

Suas particularidades não são mostradas aos desenvolvedores, por exemplo, no caso do GAE basta que se entenda de JPA, e se faça a configuração do arquivo *persistence.xml* que já estará conectado.

Bigtable é também um noSQL, por ser um banco de dados que não utiliza código SQL .

Muito utilizado para grande armazenamento de dados tais como *petabytes*, como exemplo de programa que usa o *bigtable* pode destacar o Google *Earth*, que é um programa que necessita de grande armazenamento de imagens.

2.9 GOOGLE APP ENGINE

Google *App Engine* é uma PaaS, que facilita no desenvolvimento de aplicativos e simplifica a portabilidade, usado para rodar aplicativos na *web*, oferece suporte a diversas linguagens, são elas: Java, Ruby, Python, Go. Não é necessário se preocupar com a parte física de servidores, apenas fazer o *deploy* da aplicação e o mesmo já estará pronto para uso.

Algumas de suas funcionalidades são:

- Oferecer serviços escalonáveis de manipulação de imagens, armazenamento de dados JPEG e PNG cortar, inverter, redimensionar.

- Usar contas do Google para autenticar usuários e obter emails que esteja utilizando a aplicação.
- Testar a versão nova enquanto a antiga está em execução.
- Pode configurar com seu domínio ou usar o domínio gratuito *appspot.com*

Com o GAE é possível identificar quando é um administrador ou um usuário comum, auxiliando na criação de área restrita para administradores. À medida que aumenta o tráfego e armazenamento é de fácil manutenção.

2.9.1 Ambiente Java

O GAE possibilita a construção e execução de aplicativos Java *web* na infraestrutura do Google, ele oferece suporte ao Java 5 e Java 6 os aplicativos são executados usando a JVM que contém interface de Java Servlet e JSP.

Utiliza estrutura WAR padrão para armazenamento da aplicação (*servlets*, jsp, arquivos html, etc.) o GAE facilita no entendimento.

Sandbox ambiente seguro onde é executado a JVM impede a interferência entre aplicativos. Garante que os aplicativos possam realizar apenas ações que não interfiram com o desempenho e a escalabilidade de outros aplicativos (GOOGLE, 2012).

A Figura 5 mostra a visão geral de suporte a Java do GAE.

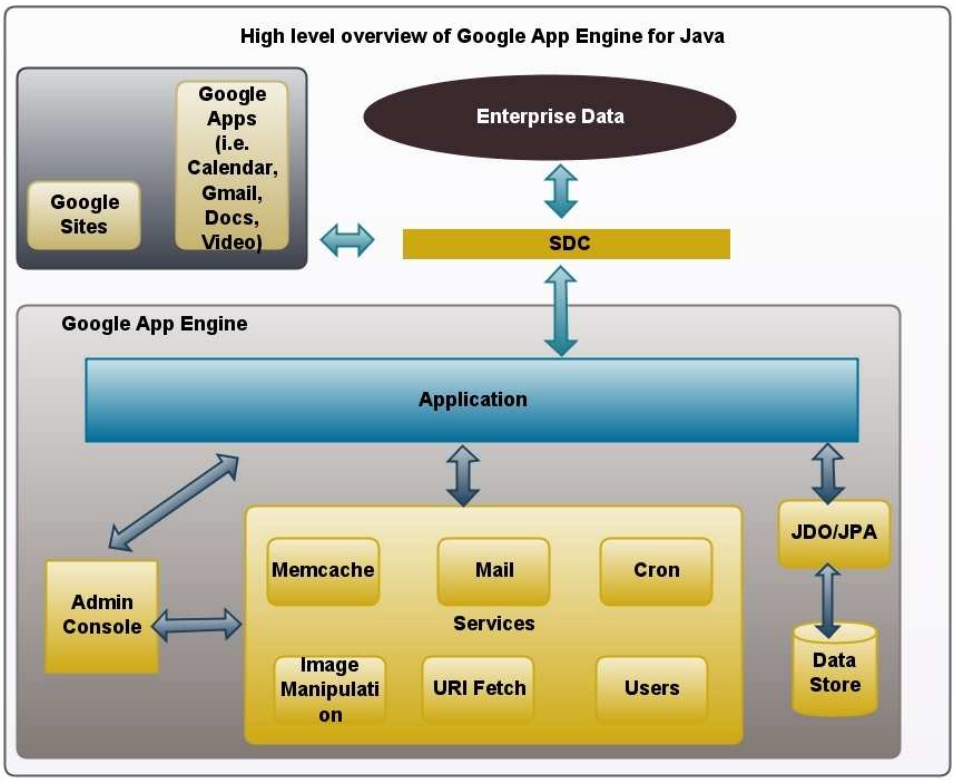

Figura 5 – GAE para Java (BYTEONIC, 2012)

2.9.2 Armazenamento de dados

DataStore também conhecido como *Bigtable*, é um banco de dados confiável e escalonável para armazenamento de dados persistente, contém duas interfaces Java usadas no armazenamento de objetos são elas:

- JPA: Facilita a portabilidade do aplicativo para diferentes bancos de dados já que não usa código especifico de banco de dados, mas trabalha com objetos, em seu padrão JPA interage com banco de dados relacional, porém no GAE não se usa banco de dados relacional por esse motivo há recursos que não são compatíveis.
- JDO: Como JPA o JDO também é um padrão Java para persistência de dados, e também não é totalmente compatível com o GAE.

A Plataforma de Acesso *DataNucleus* é usada para dar suporte ao JPA e JDO.

O *Memcache* é um armazenamento rápido em *cache*, a interface Java implementa o *JCache*.

2.9.3 Cotas e Limites para uso

As solicitações de domínio gratuitas recebe o nome de nome-aplicacao.appspot.com onde o domínio sempre será appspot.com, a cada conta pode-se ter até 10 aplicativos.

Existe uma opção de versão onde é possível testar uma nova versão de aplicativo sem a necessidade de torna - lá padrão, assim caso não seja conveniente, pode deixar a versão atual.

O Google aloca uma quantidade de recursos para seu aplicativo gratuitamente e automaticamente, ou seja, existe um limite de uso, caso exceda a quantidade pode-se requerer uma maior quantidade, já nesse caso é preciso pagar uma quantia. É possível controlar os custos assim só paga o que realmente desejar.

Começar a usar o GAE é grátis. Todos os aplicativos podem usar até 500 MB de armazenamento e CPU e largura de banda suficiente para suportar um aplicativo eficiente que ofereça cerca de cinco milhões de visualizações de página por mês, totalmente grátis. Ao ativar o faturamento para seu aplicativo, os limites gratuitos aumentam e você paga somente pelos recursos que ultrapassam os níveis gratuitos (GOOGLE, 2012).

2.9.4 Interface de controle

Em seu ambiente de controle, gerencia todas as aplicações disponíveis, cria novos aplicativos, verifica o armazenamento de dados, *logs* de erros, versões do aplicativo.

A Figura 6 mostra a interface de controle.

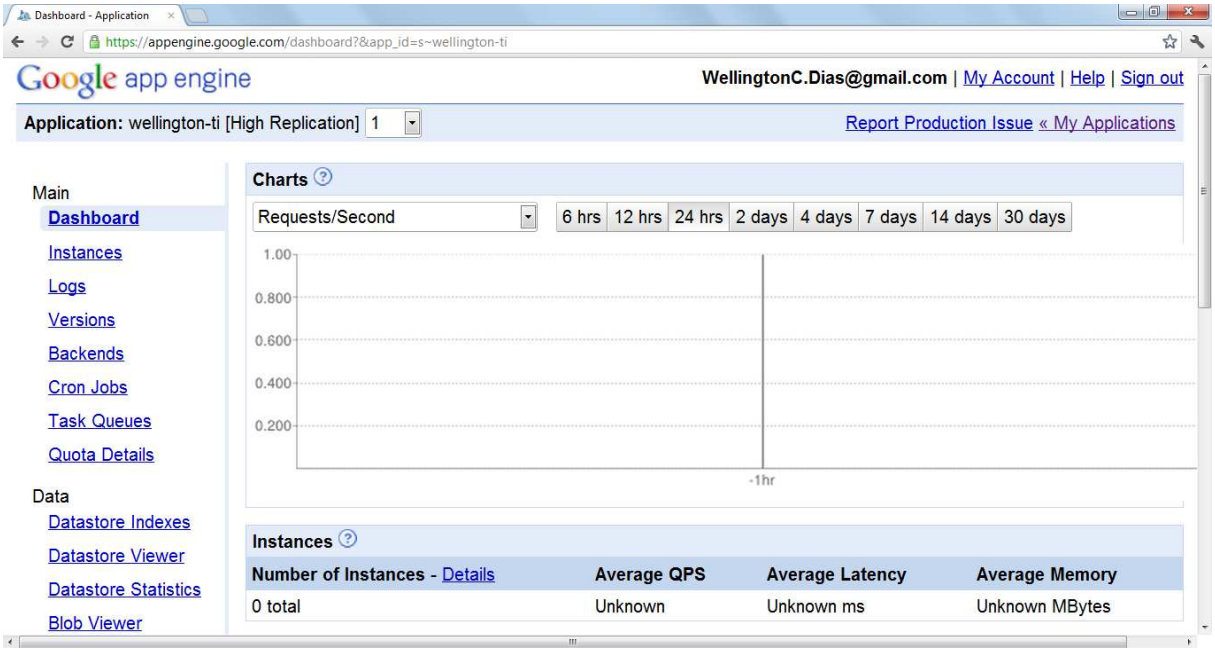

Figura 6 – Interface de Controle (GOOGLE, 2012)

2.10 ECLIPSE

O Eclipse é uma das principais plataforma IDEs (*Integrated Development Environment* – Ambiente Integrado de Desenvolvimento) usada para o desenvolvimento de aplicações em Java, C/C++ , PHP e outras, disponível para sistemas operacionais como Windows, Linux e Mac OSX.

A *Eclipse Foundation* foi criada em janeiro de 2004 como uma corporação sem fins lucrativos independente para atuar como o administrador da comunidade Eclipse (Eclipse, 2012).

Suas versões recebe os nomes:

- Eclipse Callisto (versão 3.2) em 2006.
- Eclipse Europa (versão 3.3) em 2007.
- Eclipse Ganymede (versão 3.4) em 2008.
- Eclipse Galileo (versão 3.5) em 2009.
- Eclipse Helios (versão 3.6) em 2010.
- Eclipse Indigo (versão 3.7) em 2011.
- Eclipse Juno (versão 3.8) em 2012.

Os nomes são referentes à astronomia, por exemplo, Calisto é uma lua de júpiter.

2.10.1 Plugin GAE

Inclui SDK completo do GAE para desenvolvimento e teste do aplicativo sem ser necessário estar na *web*, apenas no momento da implantação. Executa localmente para testes na fase de desenvolvimento. Simula o armazenamento de dados e as restrições do *sandbox*.

Após o desenvolvimento do aplicativo, o *plugin* possui a opção de enviar para a *internet* sendo necessário apenas estar logado.

Com o *plugin* algumas etapas na hora da criação do projeto são feitas automaticamente, tais como inclusão de JARs (*Java ARchive* – Arquivo Java) e organização das pastas em diretórios específicos.

O *plugin* adiciona assistentes de novos projetos e configurações de depuração na IDE Eclipse para projetos do GAE (GOOGLE, 2012).

A Figura 7 mostra o *plugin* GAE instalado no Eclipse Helios.

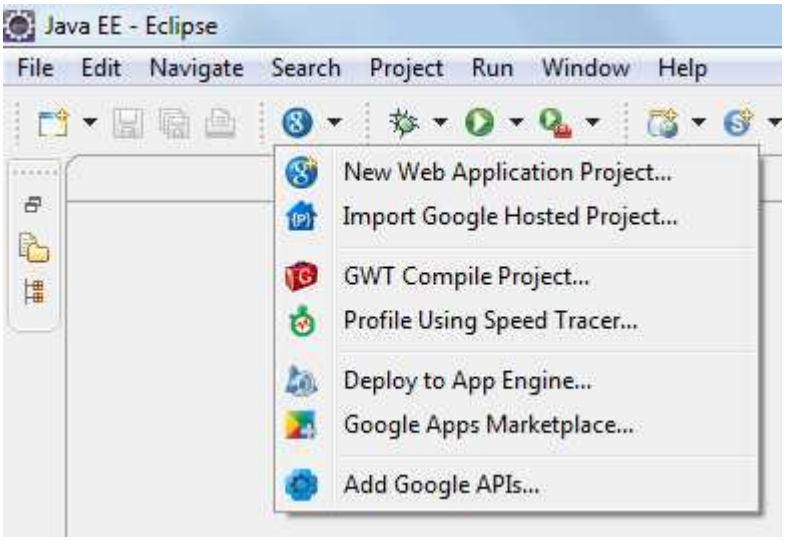

Figura 7 – *Plugin* GAE para Eclipse

Onde se nota as opções de criação do projeto "*New Web Application Project*" e de implantação do aplicativo "*Deploy to App Engine*" como foi dito anteriormente.

3. DESENVOLVIMENTO DO APLICATIVO

Neste capítulo, será feita uma descrição do desenvolvimento da integração de aplicativos através do GAE. Esta descrição envolve a modelagem, especificação, implementação e a implantação do aplicativo.

3.1 DESCRIÇÃO DO PROBLEMA

Nesse trabalho foi desenvolvido um aplicativo *web* com tecnologia Java utilizando o GAE, com o intuito de disponibilizar consultas para compra e venda de veículos. O estudo de caso foi feito para uma revendedora. A principal funcionalidade desse aplicativo é que os usuários possam efetuar seu cadastro e de seu veiculo via *web*, caso queiram vendê-los. Outra funcionalidade é que os veículos cadastrados podem ser visualizados por outros usuários interessados em comprar. Na Figura 8 pode ser observada a estrutura geral desse aplicativo *web*.

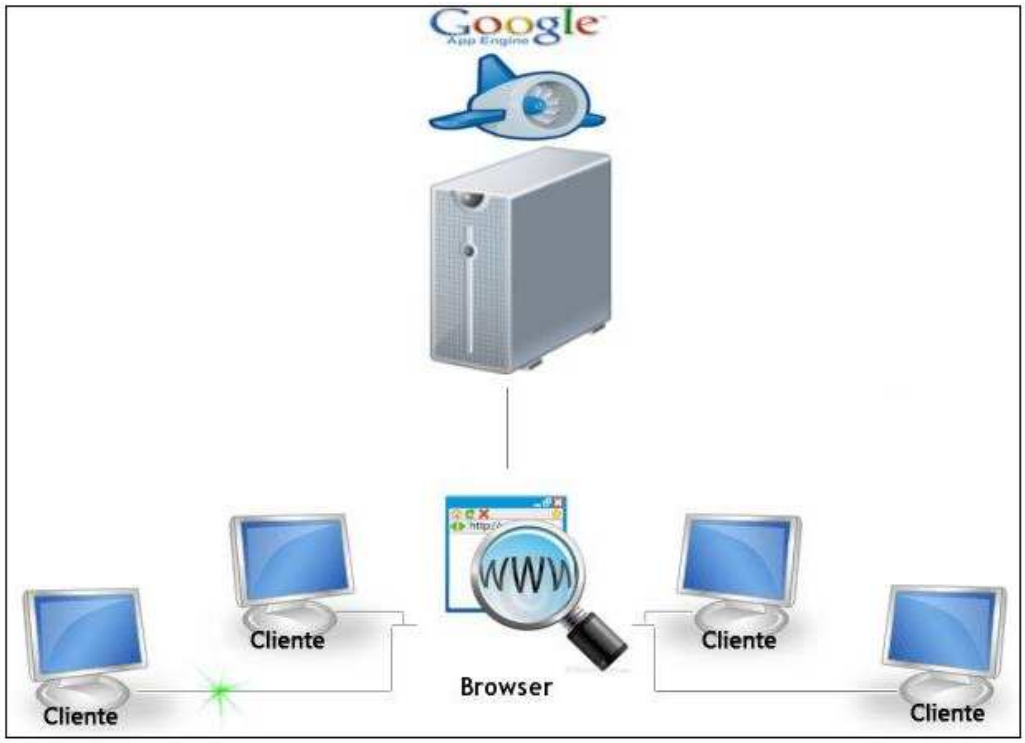

Figura 8 – Estrutura Geral do Sistema Web

3.2 MODELAGEM DO PROBLEMA

Será apresentada nesta seção a modelagem do aplicativo, dando uma visão geral das ferramentas utilizadas. A Figura 9 mostra a modelagem do problema.

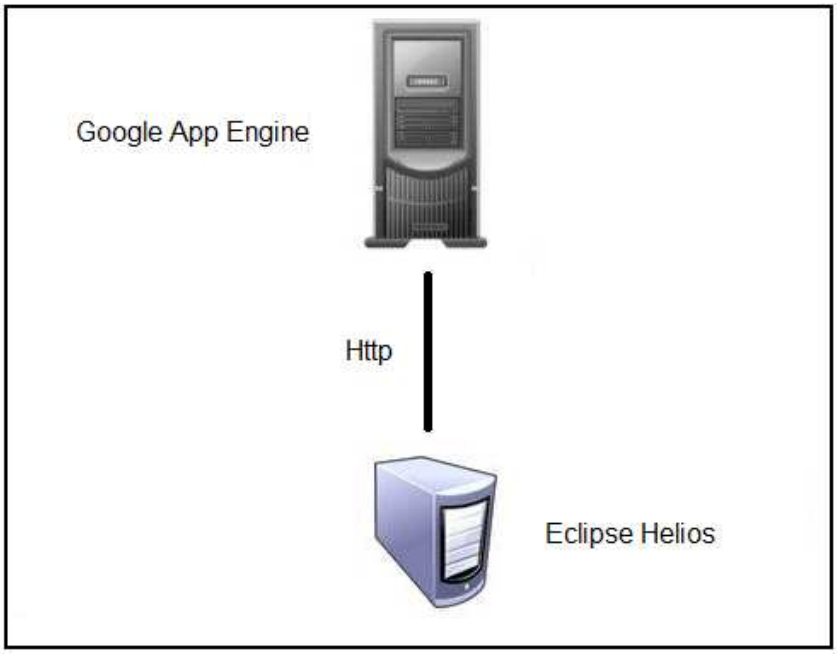

Figura 9 – Modelagem do Problema

3.3 DESENVOLVIMENTO DO APLICATIVO

O desenvolvimento deste aplicativo se deve em função da carência de serviços de compra e venda de veiculo pela *internet* para pequenas empresas, e foi impulsionado pela grande demanda destes serviços. Atualmente, empresas de diversos setores disponibilizam seus produtos pela *web*. O estudo de caso será para uma revendedora de veículos.

3.3.1 Especificação

Para a especificação do projeto, foi utilizada a UML (*Unified Modeling Language*), com auxílio da ferramenta ArgoUML, sendo criados os diagramas de casos de uso, classes e atividades.

3.3.1.1 Diagrama de Caso de Uso

Diagrama de Caso de Uso é uma forma de representar os principais cenários do aplicativo, usando atores e casos de uso para representar qual a ligação entre eles, de modo a definir os requisitos funcionais do sistema. A Figura 10 apresenta uma visão geral do diagrama de casos de uso do aplicativo.

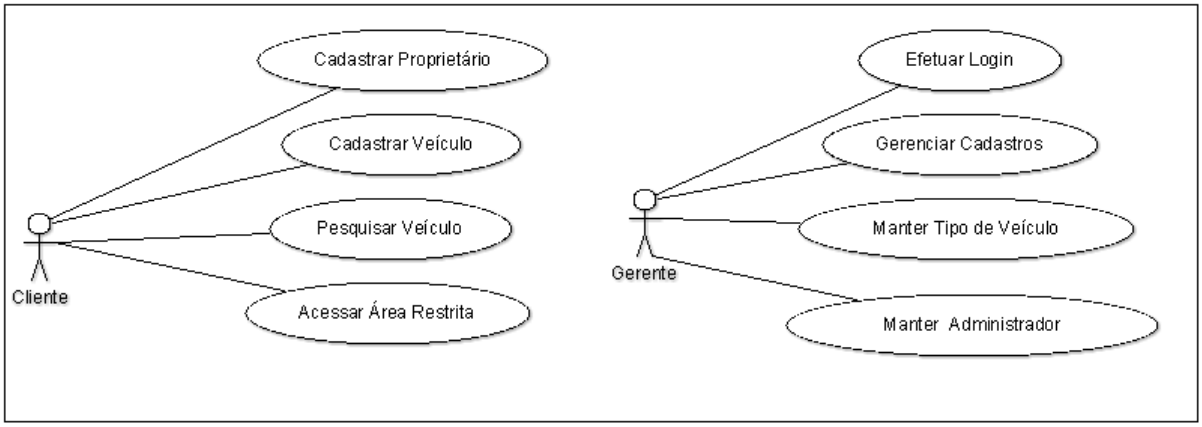

Figura 10 – Diagrama de Casos de Uso Geral

Neste diagrama de casos de uso têm-se dois atores e 8 casos de uso:

- **Atores:**

Usuário: responsável pela compra e venda de veículos;
Gerente: responsável pela validação dos dados do usuário;

- **Casos de uso:**

Cadastrar Proprietário: responsável pelo cadastro dos dados do proprietário;

Cadastrar Veículo: responsável pelo cadastro dos dados do veículo;

Pesquisar Veículo: responsável pela pesquisa de veículos;

Acessar Área Restrita: responsável pela área onde o usuário faz suas alterações;

Efetuar Login: responsável pela autenticação do administrador;

Gerenciar Cadastros: responsável pela validação dos dados do proprietário e do veículo;

Manter Tipo Veículo: responsável pelas operações de cadastrar, pesquisar, excluir e atualizar do tipo veículo;

Manter Administrador: responsável pelas alterações do administrador;

3.3.1.2 Diagrama de Classes

O diagrama de classes representa as principais classes utilizadas e está separado em pacotes:

- **Pacote *beans***

- **TipoVeiculo:** classe usada para representar os tipos de veículos.
- **Veiculo:** classe usada para representar os veículos.
- **Proprietario:** classe usada para representar o proprietário.
- **Adminstrador:** classe usada para representar o administrador.

A Figura 11 mostra a representação das classes do pacote *beans*.

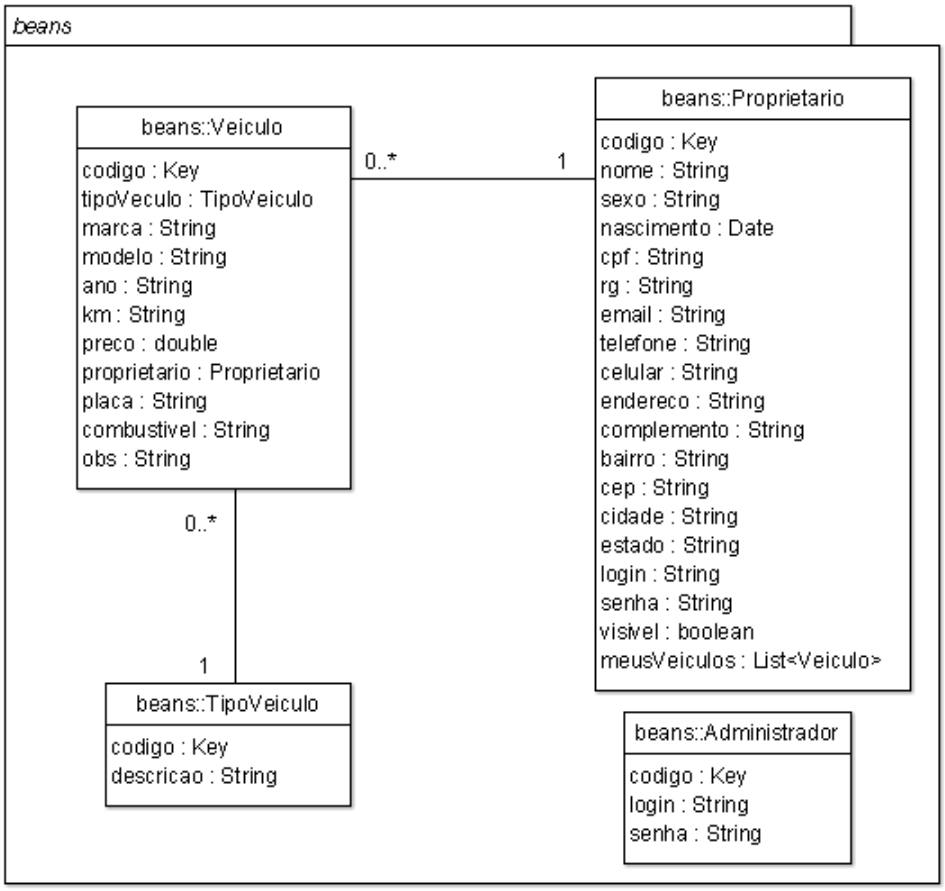

Figura 11 – Diagrama de Classes - Pacote *beans*

- **Pacote *dao***

- **TipoVeiculoDao:** classe usada para comunicação entre a classe TipoVeiculo do pacote *beans* e o banco de dados.
- **VeiculoDao:** classe usada para comunicação entre a classe Veiculo do pacote *beans* e o banco de dados.
- **ProprietarioDao:** classe usada para comunicação entre a classe Proprietario do pacote *beans* e o banco de dados.
- **AdministradorDao:** classe usada para comunicação entre a classe Administrador do pacote *beans* e o banco de dados.

A Figura 12 mostra a representação das classes do pacote *dao*.

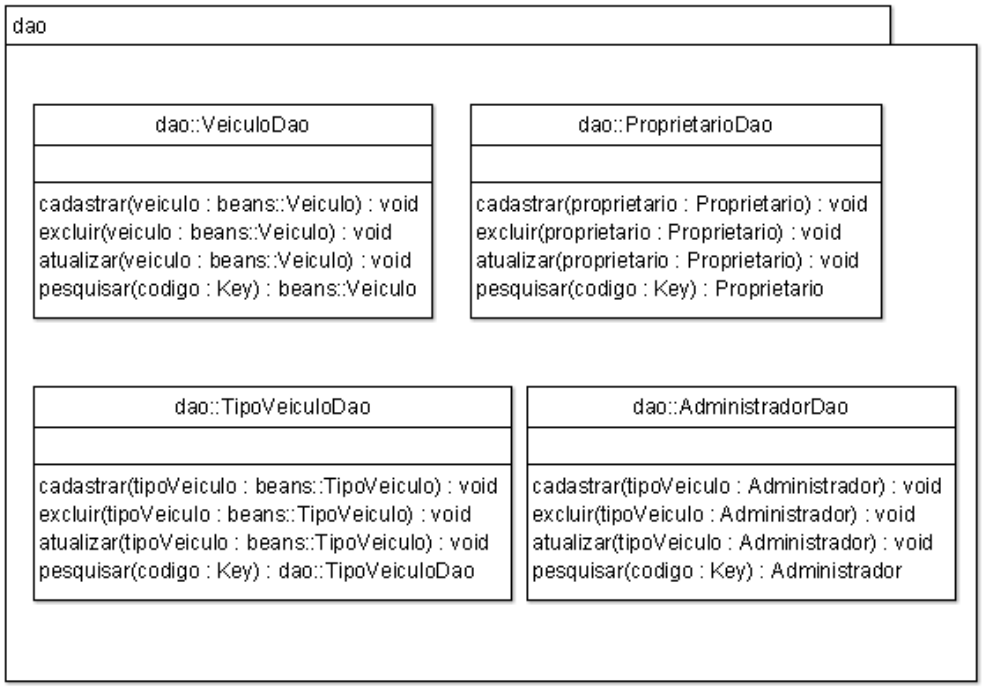

Figura 12 – Diagrama de Classes - Pacote *dao*

- **Pacote *util***

- **EMF:** classe usada para criar uma instancia para persistência no banco de dados.

A Figura 13 mostra a representação da classe do pacote *util*.

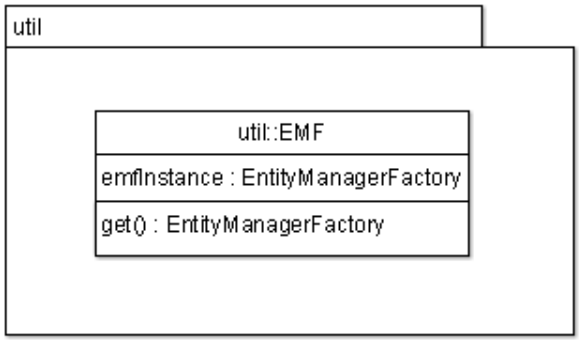

Figura 13 – Diagrama de Classes - Pacote *útil*

3.3.1.3 Diagrama de Atividades

O Diagrama de Atividades é usado para modelar a ordem que ocorrem as atividades no sistema, pois existe um caminho lógico para que o fluxo aconteça, existem atividades que possui precondições para serem executadas. A Figura 14 mostra o diagrama de atividades para venda do veículo.

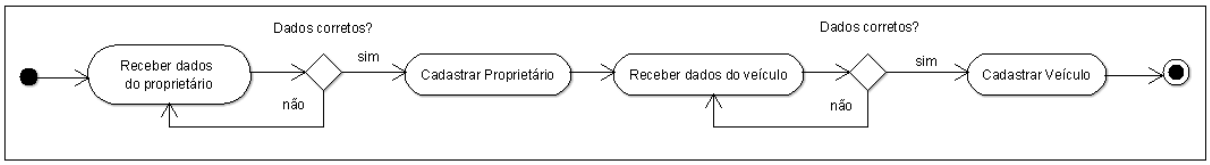

Figura 14 – Diagrama de Atividades de Venda do Veículo

3.4 IMPLEMENTAÇÃO

O Aplicativo foi desenvolvido usando o ambiente de programação Eclipse Helios com *plugin* do GAE.

Foi desenvolvido um aplicativo web capaz de cadastrar proprietários e veículos, e disponibilizar os veículos a quem deseja comprá-los, sendo necessário primeiro que o administrador do sistema valide os dados cadastrados para serem consultados.

Para o desenvolvimento do projeto foi necessário à instalação do *plugin* do GAE para o Eclipse, para o desenvolvimento em sua estrutura. A instalação é feita através do *Install New Software* do Eclipse.

A Figura 15 mostra a página inicial do aplicativo.

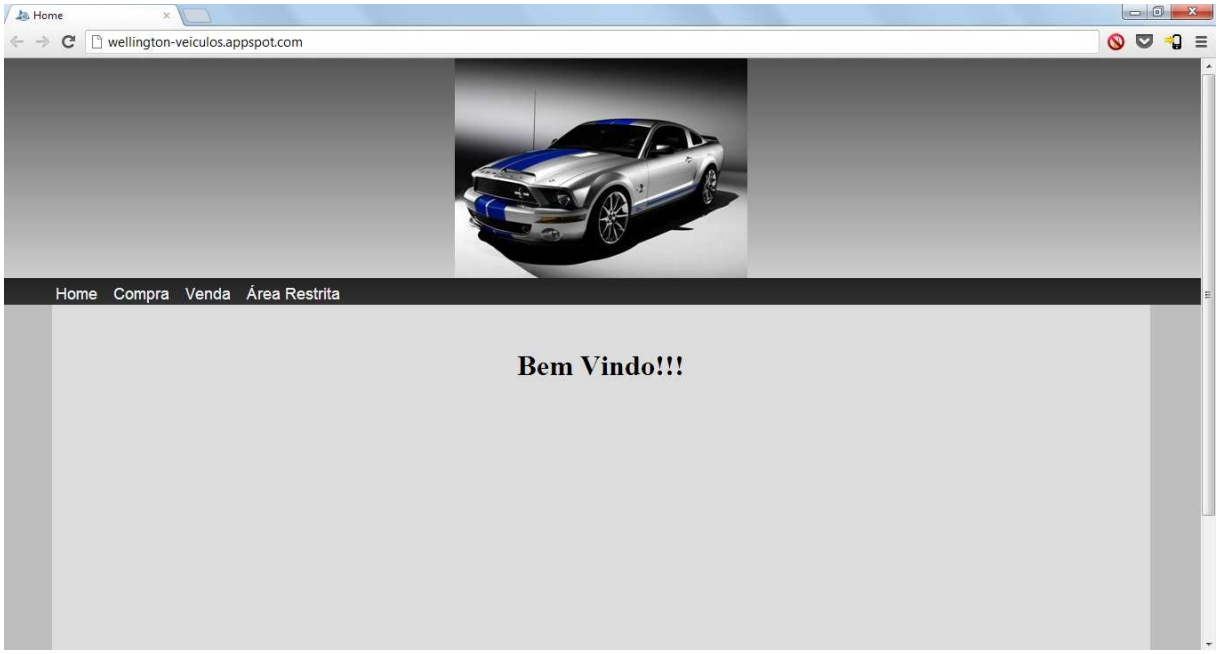

Figura 15 – Página Inicial

3.4.1 Funcionamento das telas cadastrar proprietário e cadastrar veículo

Para o cadastro de proprietário é necessário o preenchimento de todos os campos exceto o campo complemento, depois basta salvar. Para o cadastro de veículo é necessário primeiro confirmar os dados do proprietário e apenas o campo observações não é obrigatório nesse caso. Para que os dados apareçam no site é preciso aguardar a validação do administrador. A Figura 16 mostra o cadastro de proprietário e a Figura 17 mostra o cadastro de veículo.

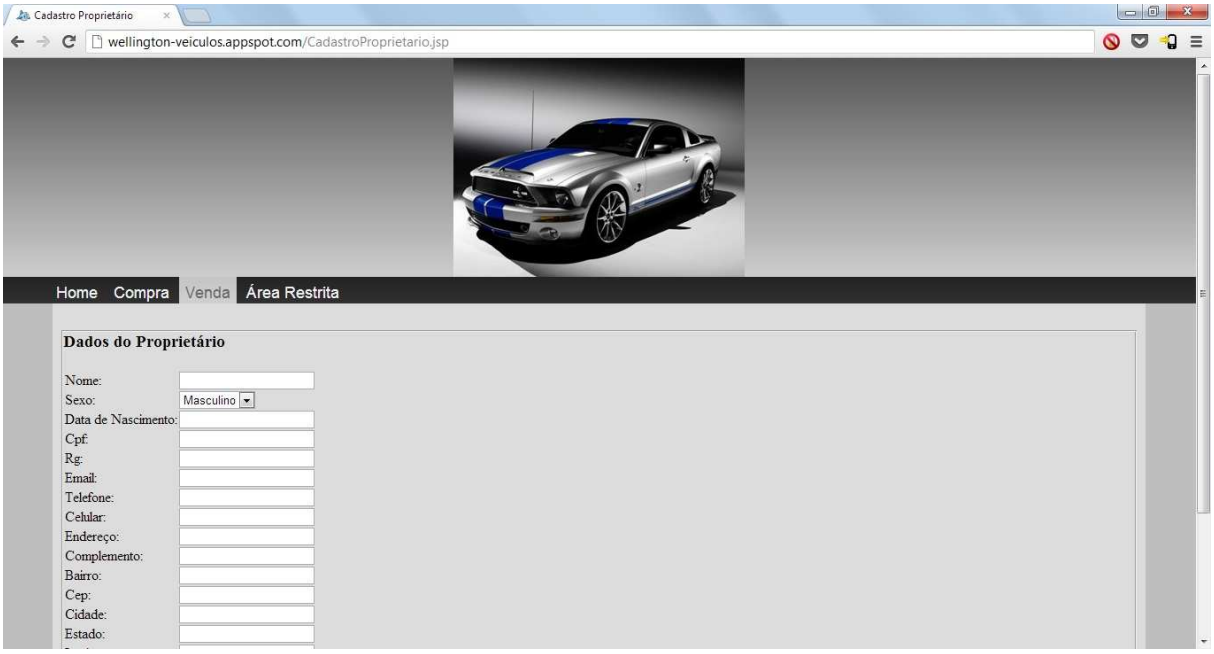

Figura 16 – Cadastro de Proprietário

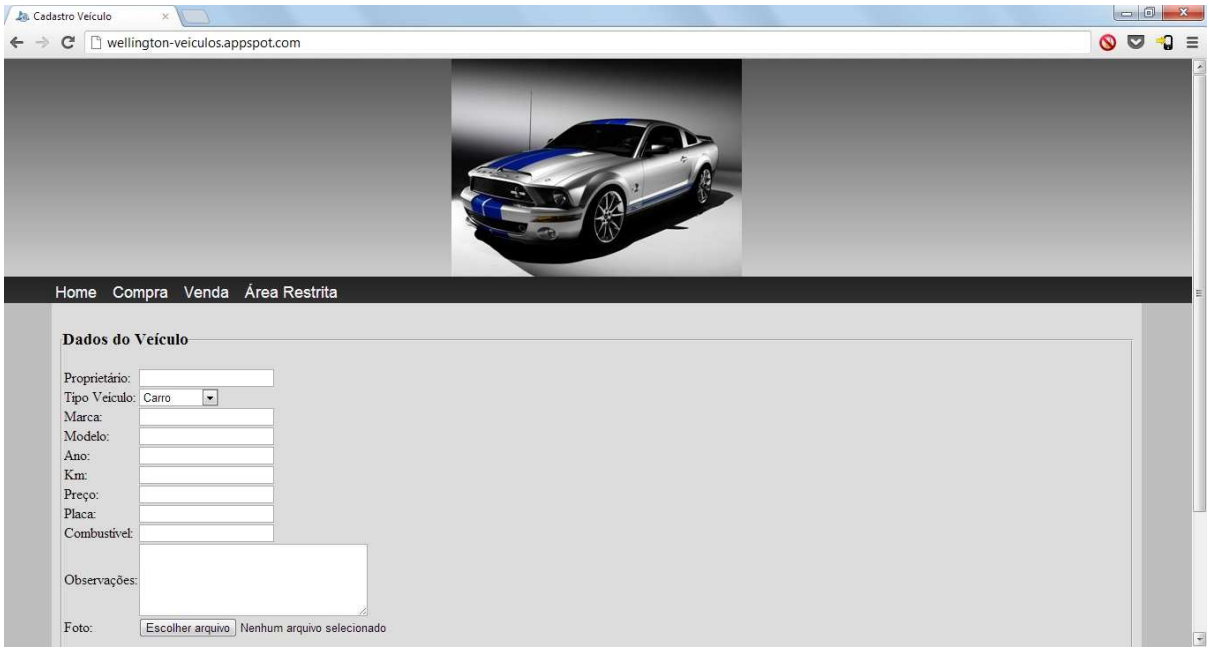

Figura 17 – Cadastro de Veículo

3.4.2 Funcionamento da tela comprar veículo

Para comprar um veículo é necessário fazer uma busca do veículo de seu interesse, onde a busca retorna uma tabela contendo alguns dados dos veículos, com as

características pesquisadas e para informações mais detalhada, basta clicar no botão detalhe referente ao veículo na tabela.

3.4.3 Funcionamento da tela área restrita

A tela área restrita é usada para que o usuário acesse sua área, onde contém as informações de seu cadastro para que possa ser alteradas. As Figuras 18 mostra a página de *login*.

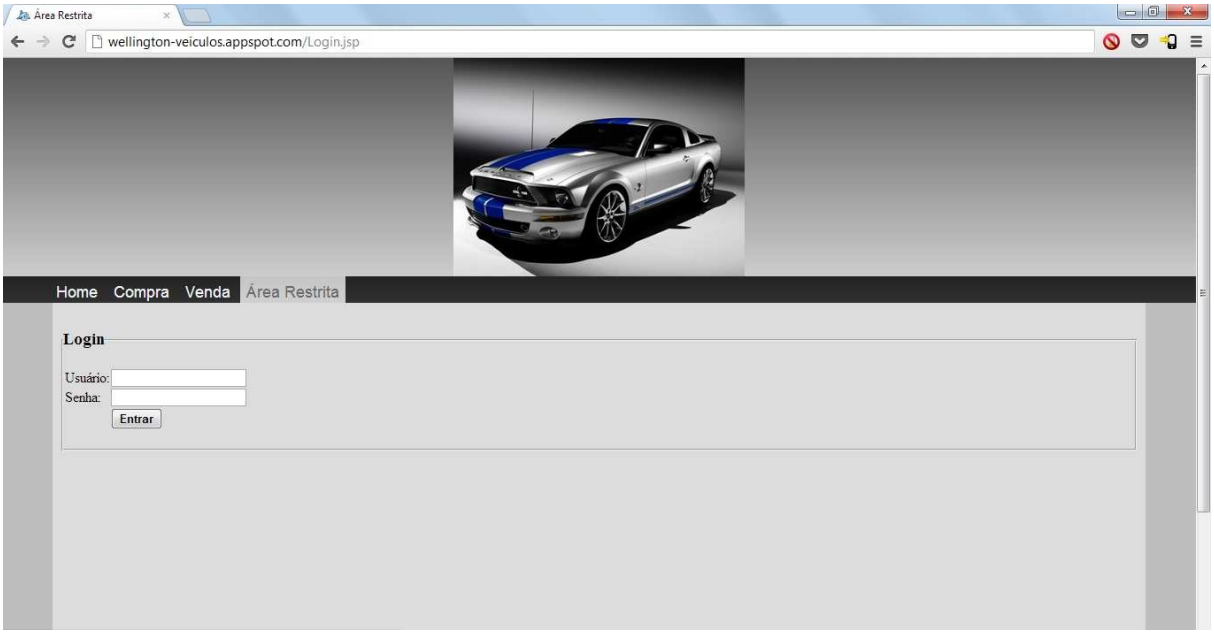

Figura 18 – Efetuar Login

3.4.4 Funcionamento da área do administrador

Todas as operações realizadas como administrador requer antes entrar em sua área efetuando o *login*. A Figura 19 mostra a tela de *login* do administrador.

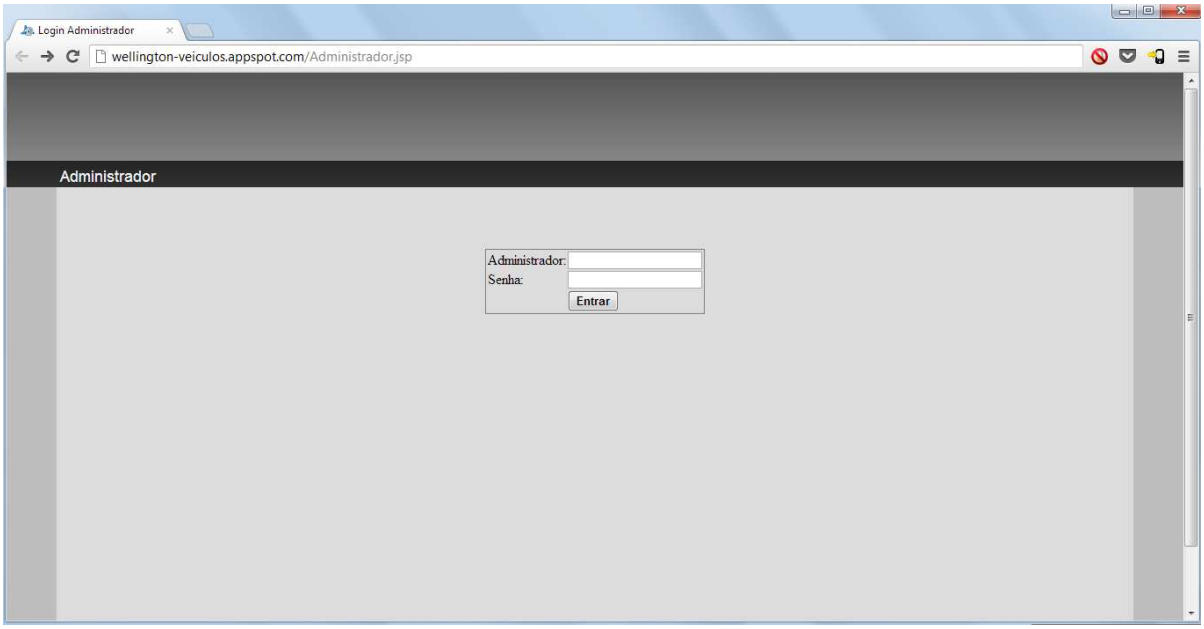

Figura 19 – Login do Administrador

3.4.5 Funcionamento da tela aguardando

Aguardando é a tela que o administrador do sistema entra para verificar quais são os cadastros que ainda não tiveram sua aprovação, para que possam serem exibidos no site, como é a área do administrador é necessário antes fazer o *login* como administrador.

3.4.6 Funcionamento da tela tipo de veiculo

Tipo veículo é a tela usada quando o administrador deseja cadastrar, pesquisar, alterar ou excluir um tipo de veículo, antes de executar qualquer uma dessas tarefas é necessário o *login* do administrador. O cadastro do tipo do veículo é feito preenchendo apenas um único campo, a descrição.

3.4.7 Funcionamento da tela de alterar dados do administrador

Caso desejar, o administrador pode alterar seus dados de *login* e senha, para isso basta entrar como administrador, e ir para a tela onde altera seus dados, informando o *login*, senha e a confirmação da senha.

3.5 IMPLANTAÇÃO

O Aplicativo web foi implantado no *Google App Engine* através do Eclipse, usando a opção *Deploy to App Engine*.

Para a implantação é necessário a criação de uma conta no site appengine.google.com, após criar a conta deve-se clicar no botão *Create Application*, onde será escolhido o nome de identificador do aplicativo com o domínio .appspot.com, como é publico é necessário que esse nome seja único Figura 20.

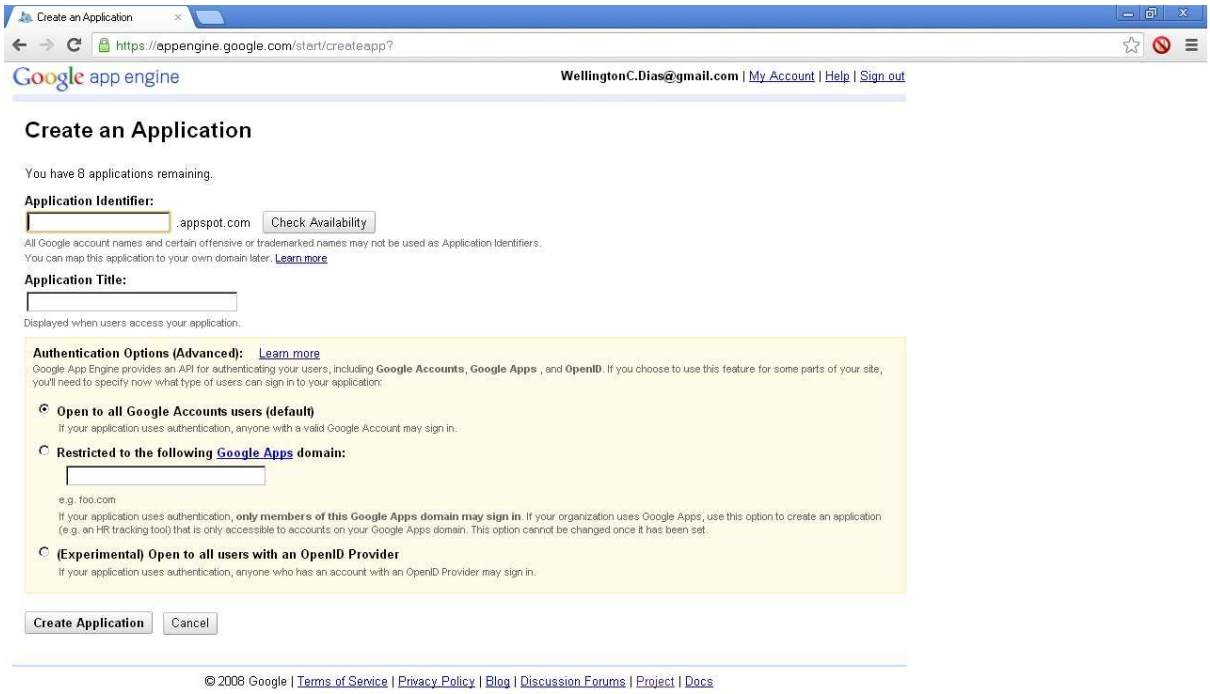

Figura 20 – Criação da estrutura para armazenar o aplicativo

Existe um arquivo no projeto chamado appengine-web.xml, onde deve-se colocar o nome do identificador anteriormente criado no site na *tag* <application></application> e o número da versão na *tag* <version></version>, essa opção de versão possibilita ao desenvolvedor ter várias versões de um mesmo aplicativo na web, podendo alternar a versão exibida facilmente, muito utilizado na hora de colocar alguma nova versão para testá-la e caso não esteja correta basta mudar a versão para a anterior.

4. CONCLUSÃO

4.1 CONSIDERAÇÕES FINAIS

Este trabalho foi motivado pelo uso da tecnologia *Google App Engine* (GAE), que é uma Plataforma como Serviço (PaaS – Platform as a Service), que são recursos de *hardware* e *software*, disponibilizados em nuvem.

Desafios de aprender a desenvolver o aplicativo *web* utilizando o padrão MVC no GAE e JPA para armazenamento de dados.

O aplicativo disponibiliza ao usuário um local para o seu cadastro e seu respectivo veículo, para que possa vendê-lo, caso desejar comprar um veiculo basta executar uma busca com as características que deseja encontrar.

Este trabalho se justifica tendo observado a necessidade de facilitar a compra e venda de veículos, e com o avanço tecnológico o número de usuários conectados na *web* cresce constantemente, sendo assim, um fácil acesso as aplicações *web*.

4.2 TRABALHOS FUTUROS

Implementação de novos aplicativos no *Google App Engine*, já que foram adquiridos os conhecimentos básicos dessa infraestrutura e utilização de outras tecnologias como: *JavaServer Faces* e *RichFaces*.

REFERÊNCIAS

ANDERSON, R. et al. **Professional XML**. 1ª ed. Rio de Janeiro: Ciência Moderna, 2001.

BATES, B. e BASHAM, B. e SIERRA, K., **Use a cabeça Servlets & JSP**, trad. Eveline Vieira Machado, 2ed. Alta Books Editora, 2008.

BRADLEY, N., **The XML Schema Companion**, Addison Wesley, 2002.

BYTEONIC, **Overview of Java Support in google app engine**, http://www.byteonic.com/2009/overview-of-java-support-in-google-app-engine/, acesso em Junho de 2012.

DAUM, B. e MERTER, U., **Arquitetura de sistemas com XML**. 1ª ed. Rio de Janeiro: Campus, 2002.

DEITEL, H.M., **Java como programar**, trad. Carlos Arthur Lang Lisbôa, 4ed. Bookman, Porto Alegre, 2003.

ECLIPSE, F., **About the Eclipse Foundation**, http://www.eclipse.org/org, acesso em Junho de 2012.

FLANAGAN, D**., JavaScript O guia definitivo**, 4ed. Bookman, Porto Alegre, 2002.

GOMES, A.L**., XHTML/CSS: criação de páginas web,** Editora Senac São Paulo, 2010
.

GONÇALVES, A**., Beginning Java EE 6 Platform with GlassFish 3: From Novice to Professional**, Apress, 2009.

GONÇALVES, E**., Desenvolvendo Aplicações Web com JSP Servlets, JavaServer Faces, Hibernate, EJB 3 Persistence e Ajax,** Editora Ciência Moderna Ltda, Rio de Janeiro, 2007.

GOOGLE, INC., **Visão geral de Java no App Engine**, https://developers.google.com/appengine/docs/java/overview?hl=pt-BR, acesso em março de 2012.

LEMAY, L. e PERKINS, C.L., **Teach Yourself Java in 21 Days**, Sams.net Publishing, 1996.

LUCKOW, D.H. e MELO, A.A., **Programação Java para a Web**, Editora Novatec, 2010.

MOULTIS, N.P. e KIRK, C., **XML: black book**. 1ª ed. São Paulo: MAKRON Books, 2000.

PERRONE, P.J., **J2EE Developer´s Handbook**, Indiana: Sam´s Publishing, 2003.

RAMALHO, J.A.A., **Iniciando em HTML**, São Paulo , Markron Books, 1996.

SAMPAIO, C.M.J., **Guia do Java: Enterprise Edition 5: desenvolvendo aplicações corporativas**, Rio de Janeiro , Brasport, 2007.

SOUSA, F.R.C. e MOREIRA, L.O. e MACHADO, J.C., **Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios**, Universidade Federal do Ceará (UFC), ERCEMAPI 2009.

TAURION, C., **Cloud computing: computação em nuvem: transformando o mundo da tecnologia da computação**, Rio de Janeiro: Brasport, 2009.