Luciano Kenji Sugiura

Trabalho de Formatura Supervisionado

BEERSPOT

Vivendo uma startup

Orientado por Fabio Kon

Sumário

1 Introdução **4**
1.1 Motivações 4
1.2 Objetivos 4
1.3 Capítulos 5

2 *Startups* **7**
2.1 O que é uma *startup*? 7
2.2 *The Lean startup* 7
2.2.1 O método *Lean Startup* 8

3 Tecnologias **11**
3.1 Servidores na nuvem 11
3.2 Ubuntu Server 11
3.3 Banco de dados PostgreSQL 11
3.4 Servidor HTTP nginx 12
3.5 Linguagem Ruby 13
3.6 Arcabouço Ruby on Rails 15
3.7 Puma - o servidor de aplicação Ruby 16
3.8 Apple iOS e Cocoa Touch 16
3.9 Google Android SDK 17

4 A *startup* Beerspot **18**
4.1 Mercado de cervejas especiais 18
4.2 Os problemas 18
4.3 Modelo de negócios e *Business Model Canvas* 19
4.4 Produtos e funcionalidades 22

5 Hipóteses iniciais e o primeiro MVP **23**
5.1 Definindo as hipóteses iniciais 23
5.2 Definindo o primeiro MVP 23

6 Produto e desenvolvimento **25**
6.1 Divulgação e consumo de ofertas 25
6.2 O modelo de dados 25
6.3 Ruby on Rails e o mapeamento objeto-relacional 28
6.4 Páginas de administração 29
6.5 API para clientes móveis 29
6.6 Aplicativo para iOS 31
6.6.1 Visão geral do aplicativo 31

6.6.2 Ofertas exclusivas . . . 33
6.6.3 Geração de cupons . . . 37
6.6.4 Outras ofertas . . . 41
6.6.5 Meus cupons . . . 43
6.6.6 Perfil . . . 45
6.6.7 Cadastro e login . . . 48
6.6.8 *Feedback* . . . 52
6.6.9 Oferta esgotada e validações . . . 54
6.7 Aplicativo para Android . . . 58
6.8 Painel do lojista . . . 58
6.9 *Test Driven Development* . . . 59
6.10 A equipe Beerspot . . . 59
6.11 O *design* de interfaces . . . 60

7 Conclusões **61**
7.1 O Trabalho de Formatura Supervisionado . . . 61
7.2 O Beerspot no ecossistema de *startups* e seu futuro . . . 61

8 Parte subjetiva **63**
8.1 O projeto como um complemento do BCC . . . 63
8.2 Disciplinas importantes . . . 63
8.3 O projeto e o futuro após o curso . . . 64

1 Introdução

O projeto Beerspot foi escolhido como tema para o Trabalho de Conclusão de Curso principalmente pela possibilidade de usar todo o conhecimento adquirido durante o curso e aplicá-lo em um projeto de uso real, com clientes e usuários reais, e com a possibilidade de se tornar uma empresa comercialmente próspera.

Durante o projeto, procurou-se aproximar sua execução ao estado-da-arte, tanto em relação à metodologia empregada na criação da *startup*, quanto em relação às tecnologias que envolvem os produtos criados para sua execução.

1.1 Motivações

- Interesse por programação para aplicativos para celular

  O autor do projeto (no momento em que este foi desenvolvido) possuía quase 4 anos de experiência em desenvolvimento de aplicativos para os sistema operacional iOS. O grande interesse pela área e a experiência colaboraram fortemente para a escolha do tema.

- Expansão do conhecimento para todas as áreas que envolvem a criação de um sistema completo e de uso real

  Durante o curso e durante a vida profissional, o autor teve pouco contato com diversas áreas que considera essencial para tornar seus conhecimentos amplos o suficiente para concretizar projetos completos e funcionais. O projeto torna-se então uma ótima oportunidade para ganhar experiência nas diversas tecnologias envolvidas.

- Empreendedorismo

  As oportunidades que a área de computação proporciona para empreendedores são enormes, pois possibilitam a execução de ótimas ideias com pouquíssimo custo, apenas com tempo e dedicação. O interesse por empreendedorismo pelo autor e essas oportunidades na área foram grandes motivadores para a escolha do projeto.

- Cervejas especiais

  O mercado de cervejas especiais está em crescimento acelerado nos últimos anos, e o gosto pela área também influenciou a escolha do tema.

1.2 Objetivos

- Conhecer e implementar todas as partes de um sistema de uso real

Ao fim do projeto, deseja-se obter um sistema pronto para uso real, com uma implementação robusta e confiável em todas as suas partes, desde a configuração do servidor até a implementação dos produtos que estarão à disposição de clientes e usuários da *startup*.

- Vivenciar o processo de criação de uma *startup*

  Através da execução deste projeto, objetiva-se obter como resultado a estrutura de uma *startup* real, compreendendo um modelo de negócios, uma estratégia de execução e um produto a ser utilizado comercialmente.

- Aprender e praticar a metodologia *Lean Startup*

  A metodologia *Lean Startup* tem se destacado como bom conjunto de práticas para obter resultados otimizados na criação e execução de *startups*. O projeto também visa o estudo e a execução do conjunto de práticas descritos nessa metodologia.

1.3 Capítulos

A seguir, uma descrição breve sobre a organização dos capítulos desta monografia.

- *Startups*

  Neste capítulo será estudado o conceito de *startups* e a metodologia para sua criação descrita no livro *The Lean Startup*, de Eric Ries.

- Tecnologias

  Será apresentado um estudo aprofundado sobre cada uma das tecnologias envolvidas para a implementação dos produtos da *startup*.

- A *startup* Beerspot

  Aqui será definido o Beerspot como uma *startup*, seus objetivos, definição e detalhamento do modelo de negócios, quais produtos serão implementados e quais serão as funcionalidades oferecidas pelos produtos.

- Hipóteses iniciais e o primeiro MVP

  Uma vez apresentados os objetivos da *startup*, os produtos a serem implementados e a metodologia *Lean Startup*, serão definidas as hipóteses iniciais do Beerspot e como estas serão validadas através do primeiro Produto Viável Mínimo.

- Produto e desenvolvimento

  Este capítulo detalhará tecnicamente como foi a implementação dos produtos definidos no capítulo anterior, as etapas envolvidas, metodologias e resultados

obtidos. Também serão apresentados os resultados obtidos através de entrevistas com alguns usuários.

- Conclusões

  Após a execução descrita nos capítulos anteriores, serão apresentados os resultados obtidos no projeto, quais objetivos foram cumpridos, o aprendizado e experiência adquiridos, qual o estado atual do projeto e os próximos passos para a criação de uma startup sólida e com potencial de sucesso.

- Parte subjetiva

  Capítulo que abordará a visão pessoal do autor em relação ao curso e ao projeto.

2 *Startups*

2.1 O que é uma *startup*?

A definição mais geral de *startup* diz que "Uma startup é uma empresa nova, até mesmo embrionária ou ainda em fase de constituição, que conta com projetos promissores, ligados à pesquisa, investigação e desenvolvimento de ideias inovadoras" [5].

Segundo Eric Ries, em seu livro *The Lean Startup*, "Uma *startup* é uma instituição humana desenhada para criar um novo produto ou serviço sob condições de extrema incerteza."

A utilização do termo se popularizou com o aparecimento em massa das "empresas ponto com", que se caracterizam por tentar atingir o mercado com ideias inovadoras de produtos que se utilizam principalmente da internet. Empresas que se encaixam nas definições acima, portanto, são comumente referidas pelo termo *startup*.

Atualmente há um destaque para as *startups* que provêem seus serviços através de aplicativos para celular, um segmento que se mantém em expansão desde o aparecimento dos primeiros *smartphones*.

Uma *startup* tipicamente começa com pouco ou nenhum recurso, inicia suas operações de forma tímida e, em pouco tempo, alcança uma fatia de mercado considerável graças ao seu caráter inovador e tecnológico. Nessa fase, a *startup* começa a estabelecer seu modelo de negócios, que é a forma ou estratégia que define como a *startup* vai gerar lucros. Quando esse processo é bem executado e a *startup* atinge um bom nível de sucesso, chega um momento em que o crescimento pode ser acelerado com a captação de investimentos. É nessa hora que investidores, atraídos pelo potencial da *startup*, se tornam sócios do negócio através do recebimento de uma fatia da empresa e aplicam dinheiro para acelerar seu crescimento. Esse dinheiro deve ser estrategicamente empregado para otimizar o crescimento, seja aplicando-o em publicidade, mão de obra qualificada ou qualquer outro setor que traga o melhor retorno possível.

Todas estas atividades, descritas informalmente acima, foram estudadas e algumas literaturas surgiram visando a otimização desse crescimento tão desejado. Dentre elas, destaca-se o livro *The Lean Startup*, de Eric Ries, o qual foi objeto de estudo deste trabalho.

2.2 *The Lean startup*

Dadas as complexidades intrínsecas às ideias inovadoras para atingir o mercado com sucesso, algumas teorias em relação ao desenvolvimento de *startups* foram elaboradas,

e a metodologia *Lean Startup* se destacou no mercado desde a sua apresentação, no livro *The Lean Startup*, de Eric Ries, em 2011.

2.2.1 O método *Lean Startup*

A adoção de métodos clássicos para a criação de produtos inovadores e/ou revolucionários costuma obter resultados totalmente aquém do que se observa nas grandes e consolidadas empresas em geral. A receita básica de se ter uma gestão bem feita, um time brilhante, tecnologia de ponta, a ideia certa no momento certo simplesmente não funcionava para produtos/serviços com grau extremo de incerteza.

Muitas empresas novas, então, optaram por não adotar metodologia alguma, como forma de se concentrar apenas no produto, deixá-lo perfeito sob a visão dos criadores, deixando de adotar qualquer modelo de gestão no processo de criação ("*Just-do-it*"), o que também não foi razão de sucesso na maioria dessas empresas.

Ries, então, procurou estudar e entender como obter um processo de gestão que se encaixe com a necessidade das *startups*. Desse estudo, surgiu o *The Lean Startup Method*, com os seguintes princípios:

1. Empreendedores estão por toda parte

   Um empreendedor não é, necessariamente, aquele que cria sua empresa em uma garagem. Todos que estão envolvidos na criação de uma *startup* (como definida por Ries), desde empresas iniciantes até equipes de inovação dentro de grandes empresas, são empreendedores. A abordagem *Lean Startup* pode se aplicar a todos eles, qualquer que seja o segmento da indústria.

2. Empreendedorismo é gestão

   Startups são instituições, não apenas um produto, o que requer uma gestão especificamente desenhada para seu alto grau de incerteza. Trabalhar preocupando-se apenas em finalizar o produto sem cuidar do modelo de gestão não é viável para uma *startup*, bem como adotar um modelo clássico que não abrange o grau de incerteza que as *startups* enfrentam.

3. Aprendizado validado

   Startups existem não pelo produto ou serviço, mas pelo aprendizado de como criar um negócio sustentável. Isto é validado por uma metodologia, através de experimentos frequentes que possibilitam ao empreendedor testar se o que se quer produzir é algo realmente demandado pelo mercado. Assim, o progresso de uma *startup* não é medido simplesmente pela qualidade do produto ou serviço desenvolvido ou pelos números absolutos gerados pelo negócio, mas sim por quanto Aprendizado Validado é adquirido ao longo do tempo.

4. Construir-Medir-Aprender

   A atividade fundamental da *startup* é, a partir da hipótese inicial, construir um produto com um mínimo de esforço possível que possa medir como consumidores reagem, aprender se a hipótese é real e decidir se é preciso mudar ou continuar. Um bom processo numa *startup* é o que possibilita acelerar esse ciclo. Novamente, o Aprendizado Validado é o foco, e o ciclo Construir-Medir-Aprender é a atividade que visa a acelerar esse aprendizado.

5. Inovação responsável

   Inovação costuma ser o foco de *startups*, e para manter a inovação como um item sustentável e que gera uma rentabilidade crescente, é preciso aperfeiçoar sempre o processo de mensurar progressos, muitas vezes tedioso, definindo marcos e priorizando o trabalho. Isso só é possível com um modelo de gestão específico para *startups*, com pessoas que mantém essa inovação sustentável.

Em *The Lean Startup*, Ries divide o livro em três partes:

1. "Visão": define o que é uma *startup*, seus atores e atividades, além de um apresentar um novo modo de mensurar o progresso da *startup*, chamado "Aprendizado Validado";

2. "Direção": descreve em profundidade o método *Lean Startup*, detalhando o processo de Construir-Medir-Aprender;

3. "Aceleração": mostra como acelerar o ciclo de Construir-Medir-Aprender, otimizando ao máximo o valor do tempo e trabalho da *startup*.

A literatura sobre administração do último século é extremamente rica, devido à abundância de estudos e casos de sucesso. Mas essa literatura não contempla o que as *startups* hoje enfrentam: o caos e o extremo grau de incerteza.

O progresso de um processo de manufatura é medido pela qualidade dos produtos produzidos. O método Lean *startup* usa uma unidade diferente de progresso, chamado *validated learning*, ou Aprendizado Validado. Usando uma metodologia bem definida para validar hipóteses que partem do plano inicial de uma *startup*, é possível descobrir e eliminar as fontes de desperdício de esforços, concentrando-os no que realmente traz valor para o cliente.

Esse método fornece um guia claro ao empreendedor quando este precisa tomar decisões importantes como: quando investir ou não em processo; como formular, planejar e criar uma infraestrutura; quando fazer sozinho ou procurar um parceiro; quando aceitar um *feedback* e quando manter a visão original; e quando começar a escalar o negócio. Acima de tudo, o método permite que se faça previsões testáveis.

As ações de uma *startup* que segue o método *Lean Startup* são baseadas no ciclo Construir-Medir-Aprender, detalhado a seguir.

- Construir

  A partir da visão inicial da *startup*, formular uma hipótese inicial sobre o problema a ser resolvido.

  Então é preciso entender se o problema realmente existe sob a visão dos consumidores.

  Para isso, em vez de se basear em (ou gastar recursos com) pesquisas de mercado ou qualquer outro tipo de suposição, realizar experimentos empíricos para entender na prática se o comportamento do consumidor está alinhado com a hipótese inicial.

  Construir um Produto Viável Mínimo (*Minimum Viable Product* ou MVP) e realizar os experimentos.

- Medir

  Observar através das métricas adequadas se a hipótese corresponde à realidade. Note que essa validação é embasada em fatos, no comportamento real do consumidor e não em suposições a seu respeito. Nem mesmo pesquisas para identificar as necessidades do consumidor são tão fortes quanto a observação do seu comportamento. Consumidores, de modo geral, não conseguem transmitir suas necessidades reais através de pesquisas, somente a utilização do produto ou serviço na prática é que nos possibilita observar se as hipóteses levantadas são reais.

- Aprender

  Utilizar o resultado destes experimentos para guiar os próximos esforços, identificando o que deve e o que não deve ser feito. Nessa fase, se identifica claramente que muitas das hipóteses iniciais não são reais, que muitas funcionalidades que pareciam claramente interessantes não são usadas na prática e evita-se que estas não tomem mais recursos da *startup*, evitando seu desperdício.

Seguir o ciclo Construir-Medir-Aprender evita o erro mais grave que uma *startup* pode cometer: construir com sucesso e perfeição um produto que ninguém usa.

3 Tecnologias

Neste capítulo, serão descritas todas as tecnologias utilizadas neste projeto. Note que estas descrições são independentes da aplicação no projeto. Para saber como cada tecnologia foi aplicada, veja a Seção 4.4.

3.1 Servidores na nuvem

Para manter o sistema disponível remotamente, foi utilizado uma instância de servidor rodando na chamada "nuvem".

Entende-se por "computação em nuvem" um grande grupo de servidores remotos que fornecem de forma centralizada serviços de armazenamento e/ou processamento.

A hospedagem de aplicações *web* foi muito facilitada por serviços na nuvem. Estes serviços possibilitam a criação de máquinas virtuais - ou instâncias de servidor, cujos recursos computacionais são distribuídos entre os servidores remotos na nuvem, tornando a manipulação de recursos, como memória e processamento, extremamente fácil e flexível.

Para o projeto, foi utilizada uma máquina virtual na nuvem oferecida pela empresa Digital Ocean [6], com 512MB de memória, 20GB e um núcleo de processamento, pelo valor aproximado de cinco dólares ao mês. A Digital Ocean, bem como outras diversas empresas de serviços na nuvem, disponibiliza uma mudança na quantidade de recursos extremamente rápida: segundo a empresa, pode-se aumentar a capacidade de memória, disco e processamento em apenas um minuto.

3.2 Ubuntu Server

O Ubuntu é uma distribuição do sistema operacional GNU/Linux muito popular, criado com base em outra distribuição (Debian). A versão Ubuntu Server 14.04 LTS foi utilizada como sistema operacional do servidor do projeto.

3.3 Banco de dados PostgreSQL

Um sistema gerenciador de banco de dados é responsável pelo armazenamento de informação e pelas operações (escrita e leitura) realizadas sobre elas.

O PostgreSQL é um sistema gerenciador de banco de dados relacional lançado em 1995, é um *software* livre (sob a licença PostgreSQL Licence) e foi escolhido por recomendação do orientador do projeto.

Algumas características importantes do PostgreSQL:

- é capaz de processar grande quantidade de acessos concorrentes;

- oferece replicação do banco de dados, para melhor escalabilidade;
- implementa o especificação SQL:2011 da linguagem SQL;
- segue as propriedades ACID (Atomicidade, Consistência, Isolamento e Durabilidade), garantindo que as transações são processadas confiavelmente;
- trabalha com transações, possibilitando voltar o estado dos dados à forma anterior quando uma operação é interrompida antes de ser finalizada.

3.4 Servidor HTTP nginx

nginx [12] é um *software* livre, servidor HTTP de alto desempenho, lançado em 2004, que tem se tornado referência devido ao seu desempenho superior em relação ao Apache. Esse desempenho superior deve-se principalmente ao fato de o nginx ter sido implementado com foco na execução concorrente e pouca utilização de memória.

Desde o aparecimento do nginx, diversas comparações com o Apache foram realizadas para demonstrar a superioridade em performance do nginx, como o exemplo a seguir do Dreamhost.

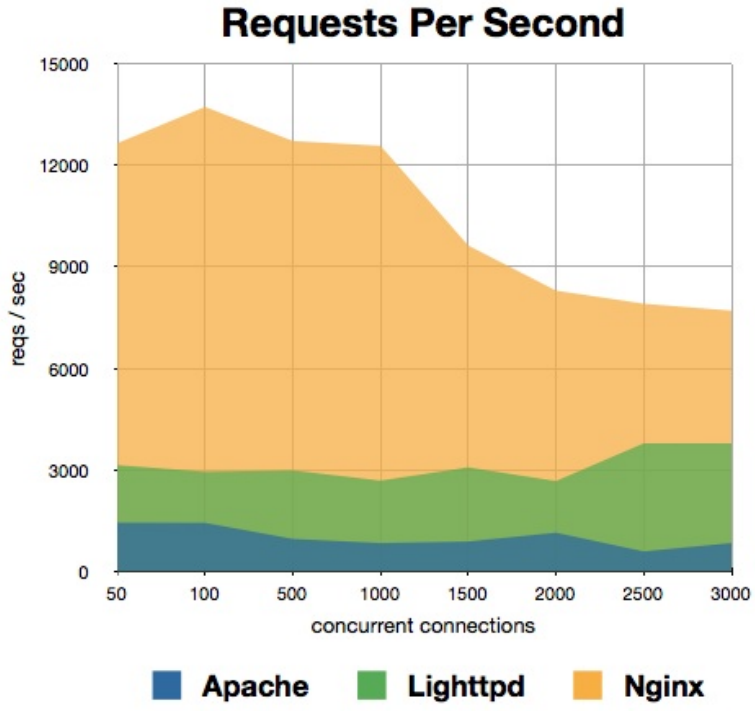

Figura 1: Comparação de desempenho de *web* servers pela Dreamhost [7]

Apesar do maior desempenho e adoção por *sites* de tráfego intenso como o Facebook, Netflix e Wordpress, o Apache ainda tem a maior participação no mercado, segundo o *site* Webnyze [22]:

Number of websites using Apache vs. Nginx

Segment	Apache websites	Nginx websites
Webnyze Universe	**8,227,005**	**2,143,006**
Alexa top 1M	490,716	206,044
Alexa top 100K	38,228	26,928
Alexa top 10K	3,119	3,140
Alexa top 1K	251	301
Alexa top 100	20	28

Figura 2: *Market share* Apache vs nginx segundo a Webnyze [2]

3.5 Linguagem Ruby

Segundo o *site* oficial [21], Ruby é "uma linguagem de programação dinâmica, software livre, focada em simplicidade e produtividade. Possui uma sintaxe elegante que é natural na leitura e fácil na escrita."

Criada em 1995 por Yukihiro Matsumoto, que usa o codinome "Matz", foi inspirada nas linguagens favoritas do autor (Perl, Smalltalk, Eiffel, Ada e Lisp) para ser uma linguagem que equilibra programação funcional e programação imperativa.

A popularidade da linguagem Ruby cresceu fortemente na comunidade de *softwares* livres, também graças à popularidade de *softwares* escritos em Ruby, particularmente o arcabouço Ruby on Rails.

Algumas características importantes da linguagem Ruby:

- Tudo é um objeto

  Em grande parte das linguagens, números e outros tipos primitivos não são tratados como linguagens. Em Ruby, todos os tipos dados como primitivos também são tratados como objetos, possuindo propriedades (chamadas variáveis de instância) e ações (chamadas métodos). Isso permite construções ricas com tipos primitivos como a seguinte:

```
5.times { print "Meu projeto é escrito em Ruby!"}
```

- Ruby é flexível

  Todas as classes em Ruby podem ser alteradas ou estendidas, a cargo do programador. É possível, por exemplo, adicionar o método `soma` à class `Numeric`:

```
class Numeric
  def soma(x)
    self.+(x)
  end
end
```

  tornando possível a utilização:

```
y = 5.soma 6
# y agora é igual a 11
```

- Blocos ou *closures*

  Closures são funções (ou referências para funções) combinadas com um conjunto de referências para valores não locais. Ou seja, durante a execução da função de uma *closure*, é possível acessar referências externas ao escopo da própria função.

  Em Ruby, as *closures* tem o nome de blocos, e são responsáveis por deixar a linguagem extremamente poderosa, em relação às linguagens imperativas mais tradicionais como C e PHP.

- Módulos e herança única

  Classes Ruby podem extender apenas de uma outra classe, ao contrário de diversas outras linguagens orientadas à objeto, como o C++.

  Porém, em Ruby existe o conceito de Módulos, que são simplesmente coleções de métodos que podem ser herdados para as classes que o implementam.

- Exceções

  Ruby utiliza exceções (*exceptions*) para lidar com tratamento adequado de erros.

- Coletor de lixo

  Ruby utiliza o recurso chamado coletor de lixo, que mantém uma contagem automática de referências para um objeto, eliminando-o da memória automaticamente quando este não está mais sendo utilizado.

- O interpretador Ruby

  Ruby é uma linguagem dinâmica, ou seja, é interpretada e compilada em tempo de execução. Seu interpretador padrão, conhecido como "MRI" (acrônimo para *Mat'z Ruby Interpreter*) é escrito em C. Porém, existem outras implementações, dentre as quais se destacam o JRuby, que é executado na Java Virtual Machine e herda bibliotecas da linguagem Java, e o Rubinius, implementação desenhada para concorrência. O projeto utiliza o interpretador padrão MRI.

3.6 Arcabouço Ruby on Rails

Ruby on Rails [17] é um arcabouço para aplicações *web* de código aberto muito difundido devido à qualidades atribuídas à própria implementação do arcabouço e à utilização da linguagem Ruby.

É um arcabouço altamente baseado em padrões e paradigmas bem estabelecidos em engenharia de *software*, incluindo:

- *Convention Over Configuration* (convenção sobre configuração)

  Presume que alguns padrões serão seguidos pelo programador, evitando assim o excesso de necessidade de configurações para que o arcabouço comporte-se como esperado. Um exemplo comum é: dada uma tabela "persons" no banco de dados, presume-se que a classe que representa uma linha dessa tabela seja nomeada como "Person".

- *DRY - Don't Repeat Yourself* ("não se repita", em tradução livre)

  Conceito que se resume na frase "Todo pedaço de conhecimento deve ter uma única, não-ambígua e autoritativa representação dentro de um sistema." Na prática, é um princípio que evita repetição de código e seus malefícios.

- *Model-View-Controller* (MVC)

  É um padrão de arquitetura de *software* que estabelece a divisão de responsabilidades em componentes: o *Model*, diretamente os dados do sistema e responsável pela lógica na manipulações destes dados; a *View* pode ser qualquer qualquer forma de representação da informação, como uma página *web*; o *Controller* é responsável por capturar a entrada/solicitação de informações e enviar os comandos necessários aos *Models* e *Views*.

  As interações entre os componentes também são estabelecidos da seguinte forma:

- *Controller*: envia comandos para o *Model* para atualizar seu estado e para as *Views*, para que estas atualizem a representação dos dados dos *Models*.
- *Model*: notifica os *Controllers* e *Views* associados para que estes reflitam o estado atual dos dados.
- *Views*: solicita informações dos *Models* para representar a informação atual.

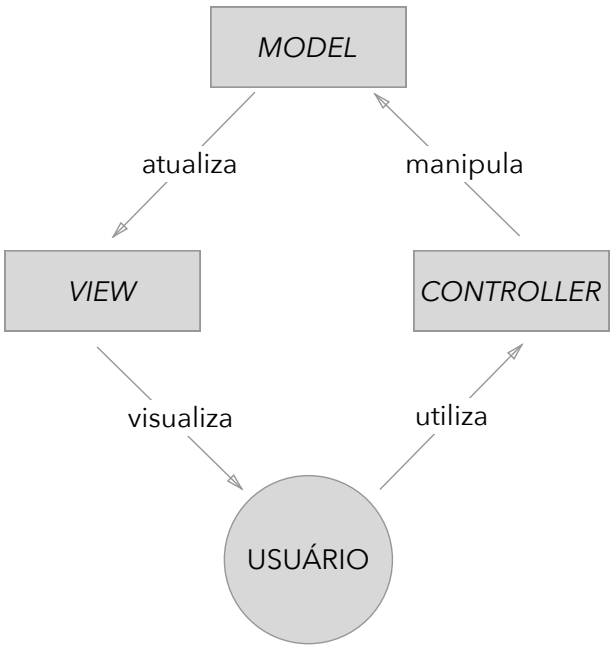

Figura 3: A interação usual entre *Models*, *Views* e *Controllers*

3.7 Puma - o servidor de aplicação Ruby

Além do servidor HTTP, é necessário o uso de um servidor *web* para executar a aplicação Ruby on Rails.

Dentre as opções de servidor *web* disponíveis, destacam-se o Phusion Passenger [13] e o Puma [14]. O servidor Puma foi escolhido após a leitura de diversas pesquisas indicando seu desempenho superior.

3.8 Apple iOS e Cocoa Touch

iOS é o sistema operacional que equipa os aparelhos móveis da Apple: iPhone, iPod Touch e iPad. Foi lançado em 2007 com nome "iPhone OS", junto com o primeiro iPhone.

Cocoa Touch é um arcabouço utilizado para criar *softwares* que rodam no sistema operacional iOS. Utiliza Objective-C, uma linguagem de programação orientada a objetos, baseada na linguagem C.

É formado um por conjunto de arcabouços, provendo de todos os recursos necessários para criar uma aplicação iOS.

3.9 Google Android SDK

Android é o sistema operacional do Google criado para celulares em 2008, atualmente extendido para utilização também em *tablets*, relógios inteligentes TV's e alguns sistemas embarcados como automóveis.

É constituído por um kernel Linux, com componentes escritos em C uma máquina virtual Java chamada Dalvik Virtual Machine, sobre a qual são executadas aplicações Java. O Google disponibiliza um SDK (*Software Development Kit* ou Kit para Desenvolvimento de *Software*) para criação de aplicações, para o desenvolvimento dos aplicativos para a plataforma.

4 A *startup* Beerspot

Neste capítulo serão explicados a área de atuação, os problemas a serem resolvidos pela *startup* e os produtos desenvolvidos na tentativa de resolver estes problemas.

4.1 Mercado de cervejas especiais

O Beerspot nasceu para atuar no mercado de cervejas especiais. A denominação "cerveja especial" não é formalizada, mas é utilizada comumente no mercado cervejeiro para classificar as cervejas produzidas com foco na qualidade, de modo artesanal, ou em escala, sem perder o caráter artesanal no qual foi concebido.

O mercado de cervejas especiais tem crescido fortemente nos últimos anos, como demonstram alguns dados dos *sites* G1 [11] e Exame [4]:

- o Brasil é o 3º maior produtor de cerveja no mundo, com cerca de 13 bilhões de litros por ano;
- em 2009, as cervejas especiais correspondiam a 1,5% 2%. Em 2014, correspondem a 5% 6%;
- entre 2011 e 2013, quase 14000 lojas especializadas no ramo foram abertas no Brasil.

O aparecimento de cervejas especiais em supermercados como os da rede Pão-de-Açúcar, Sam's Club e Záfari, além de elas estarem cada vez mais presentes nos cardápios de bares e restaurantes em São Paulo, evidenciam o crescimento do setor.

4.2 Os problemas

Como toda empresa, seja ela uma *startup* ou não, é preciso encontrar um ou mais problemas a serem resolvidos, estudá-los e criar as melhores soluções para conquistar o mercado. O problemas que o Beerspot tenta solucionar são:

- Divulgação de ofertas

  As lojas especializadas, ou empórios, frequentemente lançam promoções, seja por possuírem estoques próximos ao vencimento, ou simplesmente para atrair mais clientes. Estas ofertas, geralmente apreciadas pelo consumidor, tem pouca visibilidade, uma vez que são divulgadas apenas no ponto de venda ou nas páginas do empório em redes sociais. Portanto, somente quem passa pelo ponto de venda ou visita a página do empório tem contato com essas ofertas. Além das ofertas, também são divulgados lançamentos, eventos e outros, da mesma forma e com o mesmo problema de alcance e visibilidade da divulgação.

- Os aplicativos disponíveis hoje para consulta e compartilhamento de cervejas consumidas são insatisfatórias

  Hoje existem alguns sistemas, tanto *web* quanto em aplicativos para celular, que disponibilizam meio de consulta de rótulos e estilos e divulgação da cerveja que está sendo consumida agora - o chamado *check-in*. Em entrevistas realizadas com algumas pessoas do público consumidor de cervejas especiais, essas ferramentas deixam a desejar em funcionalidades e usabilidade.

- Há pouco conteúdo relevante para quem está começando a apreciar cervejas especiais

  O conteúdo *online* existente hoje não é nada amigável para quem está começando a descobrir esse tipo de bebida. As lojas especializadas normalmente oferecem um atendimento muito bom para quem está começando a aprender sobre estilos e rótulos, e existem alguns cursos para quem quer se especializar, mas há pouquíssimo conteúdo adequado ao consumidor leigo ou novato.

4.3 Modelo de negócios e *Business Model Canvas*

O modelo de negócios de uma empresa é uma representação de todos os elementos que compõe o negócio, a lógica que o norteia e como esses elementos interagem entre si. Essa representação demonstra como a empresa cria valor para o seu público-alvo.

Proposta em 2008 por Alexander Osterwalder, no livro *Business Model Generation* [3], o *Business Model Canvas* é uma técnica para desenvolver um modelo de negócios, colocando seus elementos em uma única folha de papel. Essa folha é subdividida em campos, cada qual representando um elemento do modelo de negócios.

Essa técnica foi adaptada para a metodologia *Lean Startup*, o chamado *Lean Canvas*, o qual foi utilizada neste projeto. Essa adaptação foi criada por Ash Maruya [24], auto de outra obra consagrada no universo das *startups*, o *Running Lean* [18].

O *Lean Canvas* contém os seguintes campos (os nomes originais são mantidos entre parênteses):

- Problema (*Problem*)

  Os problemas principais a serem resolvidos pela *startup*.

- Segmentos de consumidor (*Customer Segments*)

  Os diferentes segmentos de consumidor que a *startup* quer atingir.

- Proposta única de valor (*Unique Value Proposition*)

  Uma mensagem que estabelece por que seu produto ou serviço é diferencial e merece ser adquirido.

- Solução (*Solution*)

  As funcionalidades que o produto ou serviço terão para solucionar os problemas.

- Canais (*Channels*)

  Os meios pelos quais a *startup* alcança o consumidor.

- Estrutura de custo (*Cost Structure*)

  Todos os custos envolvidos na *startup*, incluindo custos de aquisição de consumidor, hospedagem de *website*, distribuição, pessoal etc.

- Métricas-chave (*Key Metrics*)

  Algumas métricas necessárias para medir o progresso da *startup*.

- Vantagem desleal (*Unfair Advantage*)

  Vantagem que o produto ou serviço possui que exclusiva da *startup* e dificilmente pode ser copiada ou comprada por outrem.

Beerspot - My canvas

PROBLEM	SOLUTION	UNIQUE VALUE PROPOSITION	UNFAIR ADVANTAGE	CUSTOMER SEGMENTS
Encontrar ofertas Compartilhar experiências Saber quais rótulos e estilos já provou Fonte de conhecimento/novidade Rótulos exclusivos, serviço de entrega Divulgar ofertas Auto-promoção das lojas **EXISTING ALTERNATIVES** Redes sociais cervejeiras (Brejas, Untappd, BierTab) Ofertas divulgadas em redes sociais	receber ofertas/novidades por aplicativo check-in, publicação em redes sociais catálogo de provados Ofertas exclusivas com entrega divulgação de ofertas e eventos **KEY METRICS** Pageviews de landing page para lojista, número de cadastros Pageviews de landing page de usuário, número de cadastros Pageviews no aplicativo Número de usuários que aproveitaram a oferta (validados na loja)	Conheça, compartilhe e veja ofertas das melhores cervejas Encontre efetivamente seu público alvo **HIGH-LEVEL CONCEPT**	Parcerias personalizadas Ofertas exclusivas, por valor baixo ou por exclusividade **CHANNELS** amigos internet (blogs, fóruns) flyers em empórios	Lojistas especializados (empórios) Lojistas genéricos (mercados) Usuários com alto poder aquisitivo Usuários heavy (alta frequência de consumo) Usuários novatos (baixa frequência de consumo) **EARLY ADOPTERS** Lojistas interessados Usuários heavy

COST STRUCTURE	REVENUE STREAMS
Fixos: servidor Variáveis: design e marketing Visita a lojistas Entregas Entrevistas de feedback (usuários/lojistas)	Valor cobrado sobre uso de promoção / assinaturas Espaço publicitário para lojistas Entrega como serviço premium Consultoria Big Data

Lean Canvas is adapted from The Business Model Canvas (BusinessModelGeneration.com) and is licensed under the Creative Commons Attribution-Share Alike 3.0 Un-ported License.

Figura 4: O *Lean Canvas* do Beerspot

4.4 Produtos e funcionalidades

Dados os problemas levantados acima, a seguinte série de funcionalidades foi idealizada para resolvê-los.

- Divulgação/visualização de ofertas
- Divulgação/visualização de eventos
- Guia de estilos de cervejas
- Guia de rótulos de cervejas
- *Check-in* - divulgar a cerveja consumida
- Avaliação de cervejas
- Cervejas favoritas (ou carta pessoa de rótulos)
- Conteúdo editorial, com novidades, lançamentos etc.

Estas funcionalidades serão oferecidas através dos seguintes produtos:

- Aplicativo para iOS
- Aplicativo para Android
- Aplicação *web* para lojistas

5 Hipóteses iniciais e o primeiro MVP

5.1 Definindo as hipóteses iniciais

Na Seção 4.2, foram apresentados os problemas que a *startup* identifica como problemas chave no mercado em que vai atuar, e na Seção 4.4, são descritas as funcionalidades que serão construídas para atacar os problemas.

O problema mais importante a ser resolvido é o da divulgação de ofertas. É através dele que projetamos as fontes de renda, logo é essencial que as hipóteses sobre este problema sejam validadas o mais rapidamente possível.

Assim, as hipóteses iniciais derivadas deste problema são:

- consumidores de cervejas especiais são ávidos por ofertas e as procuram frequentemente;
- lojistas querem uma ferramenta para divulgação de ofertas.

5.2 Definindo o primeiro MVP

Para validar estas hipóteses, é preciso definir quais serão as ações e produtos que serão construídos que exijam o mínimo de esforço possível.

Em primeiro lugar, entrevistas com consumidores e lojistas foram realizadas para obter os primeiros dados sobre as necessidades de ambos em relação a ofertas. Essas entrevistas foram importantes como uma primeira verificação de que as hipóteses são realmente desejadas tanto pelos consumidores quanto pelos lojistas. Foram entrevistados (informalmente, sem metodologia definida) 4 lojistas e muitos consumidores ao longo do trabalho, todos eles afirmando que uma ferramenta como a proposta seria útil.

Para prosseguir com a validação, foi definido o produto viável mínimo, que contempla:

- o *backend* do sistema, composto por um servidor rodando na nuvem, o sistema gerenciador de banco de dados e um arcabouço para páginas *web* e *web services* para os aplicativos móveis;
- páginas *web* para controle administrativo;
- páginas *web* com ferramentas para lojistas terem controle das promoções publicadas;
- um aplicativo para iOS, com autenticação de usuário (*login*) por e-mail e senha ou uso da rede social Facebook, exibição de ofertas e geração de cupons para utilização dessas ofertas;

- um aplicativo para Android, com as mesmas funcionalidades da versão para iOS do item anterior. Dada a experiência do autor com iOS, definiu-se que a versão Android seria implementada posteriormente à versão iOS.

6 Produto e desenvolvimento

Neste capítulo, serão detalhadas as etapas de desenvolvimento do produto e o todos os seus processos envolvidos.

6.1 Divulgação e consumo de ofertas

A funcionalidade chave deste primeiro MVP é a divulgação de ofertas pelos lojistas e o consumo pelos clientes. Para que ela funcione de modo adequado, foi desenvolvido um mecanismo de cupons, que funciona da seguinte forma:

- O lojista cria uma promoção, contendo um *kit* composto por cervejas, copos, dentre outros itens.
- Cada promoção tem uma quantidade definida de *kits*, definida pelo lojista.
- Cada promoção tem datas de início e fim pré-determinadas pelo lojista.
- A promoção é exibida nos aplicativos.
- O usuário interessado na promoção em questão solicita um cupom pelo aplicativo, definindo quantos *kits* quer adquirir.
- O cupom é gerado pelo sistema com um código de verificação único e exibido para o usuário pelo aplicativo.
- O sistema atualiza a quantidade disponível de *kits* disponíveis para aquisição.
- O usuário, com o cupom gerado, adquire os *kits* na loja mostrando o cupom ao lojista.
- O lojista, via painel de controle do sistema, anota o código do cupom, marcando-o como utilizado.

6.2 O modelo de dados

Para que estas interações sejam possíveis, foi criado um modelo de dados com as entidades descritas brevemente a seguir. Veja o Diagrama Entidade Relacionamento mostrado na Figura 5 para detalhes sobre o relacionamento entre essas entidades.

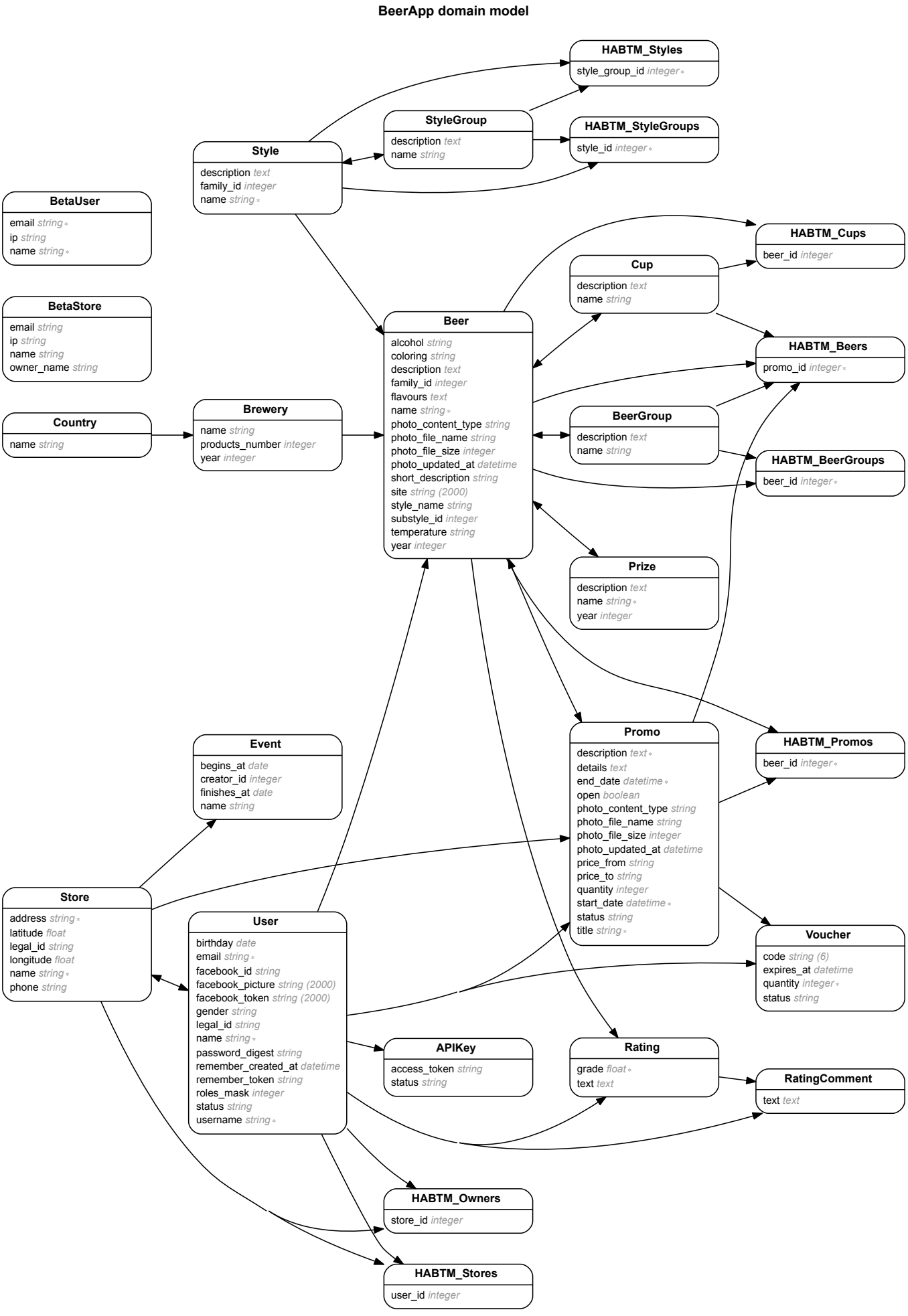

Figura 5: Diagrama Entidade Relacionamento

- *User*

  Usuário do sistema, seja ele consumidor, dono de loja ou administrador.

- *Beer*

  A entidade que representa uma cerveja.

- *BeerGroup*

  Agrupamento de cervejas.

- *Style*

  Estilo de cerveja.

- *StyleGroup*

  Agrupamento de estilos de cerveja.

- *Brewery*

  Cervejaria.

- *Country*

  País, usado para determinar o país onde uma cervejaria está instalada.

- *Cup*

  Um tipo de copo, usado para indicar qual o tipo ideal para cada cerveja.

- *Prize*

  Prêmio recebido por uma cerveja.

- *Promo*

  Promoção, representa a oferta.

- *Rating*

  Avaliação de uma cerveja por um usuário.

- *RatingComment*

  Comentário sobre uma avaliação.

- *BetaUser*

  Entidade usada para armazenar cadastros de usuários a partir da página inicial do *site*.

- *BetaStore*

  Entidade usada para armazenar cadastros de lojas a partir da página inicial do *site*.

- *Events*

  Criado para uso futuro na divulgação de eventos.

As entidades com prefixo *HABTM* no diagrama são tabelas auxiliares nos relacionamentos do tipo *has and belongs to many*.

O diagrama foi gerado a partir das classes Ruby do projeto, através do *software* Rails ERD [15].

6.3 Ruby on Rails e o mapeamento objeto-relacional

O modelo descrito na Seção 6.2 foi utilizado através do mapeamento objeto-relacional do arcabouço Ruby on Rails.

Um mapeamento objeto-relacional (ou simplesmente ORM, de *Object-Relational Mapping*) é uma técnica que conecta objetos de uma aplicação a tabelas de banco de dados relacional. Utilizando um ORM, as propriedades e relacionamentos de entidades são facilmente representadas através de atributos do objeto correspondente. Além disso, o acesso aos dados do banco de dados torna-se extremamente simples, sem a necessidade da escrita de código SQL para tal.

O Ruby on Rails utiliza o arcabouço chamado Active Record para realizar este mapeamento. No contexto do Ruby on Rails, o Active Record corresponde ao "M" no padrão "Model-View-Controller" (veja o item "Model-View-Controller").

Dessa forma, cada entidade descrita na Seção 6.2 torna-se uma classe Ruby, e cada instância desta classe representa uma linha na tabela do banco de dados representada por essa classe.

Para a criação das tabelas correspondentes às classes ou entidades, o Active Record utiliza migrações. Uma migração é escrita como uma classe Ruby, onde se descreve todos os dados da tabela a ser criada.

Abaixo um exemplo de migração para a criação do modelo Promo:

```
class CreatePromo < ActiveRecord::Migration
  def change
    create_table :promos do |t|
      t.string :title
      t.text :description
      t.date :start_date
      t.date :end_date
      t.integer :status
```

```
      t.timestamps
    end
  end
end
```

Foram escritas migrações para cada entidade, além de inúmeras migrações para editá-las conforme a necessidade.

6.4 Páginas de administração

Para controle administrativo do sistema, especificamente do dados, foi utilizado o *software* ActiveAdmin [1]. O ActiveAdmin cria, para cada entidade, um conjunto de páginas que possibilita: visualizar um índice de todos os itens da entidade; visualizar detalhes de um elemento da entidade; criar e editar atributos do elemento e apagar elementos.

A utilização do ActiveAdmin foi essencial para agilizar no desenvolvimento e obter um controle total dos itens do banco de dados através de uma interface *web*.

Um exemplo de página de administração pode ser visto na Figura 6.

6.5 API para clientes móveis

Para alimentar os aplicativos móveis com os dados do sistema, foram implementados *web services* utilizando a arquitetura REST (*Representational State Transfer*). Seguir essa arquitetura implica seguir uma série de padrões, sendo os mais relevantes neste projeto:

- Cada entidade acessível para os clientes, chamada de recurso no padrão REST, possui uma URI específica para acesso. Por exemplo, o recurso Promo é acessível pela URI *http://<servidor>/promos*.
- Os verbos HTTP são utilizados para determinar a ação sobre o recurso, dentre os quais:
  - GET http://<servidor>/<recurso> Exibe uma listagem dos recursos.
  - POST http://<servidor>/<recurso> Cria um novo recurso.
  - GET http://<servidor>/<recurso>/<id> Exibe os detalhes de um recurso com o identificador <id>.
  - PATCH/PUT http://<servidor>/<recurso>/<id> Atualiza os dados do recurso com o identificador <id>
  - DELETE http://<servidor>/<recurso>/<id> Remove o recurso com identificador <id>.

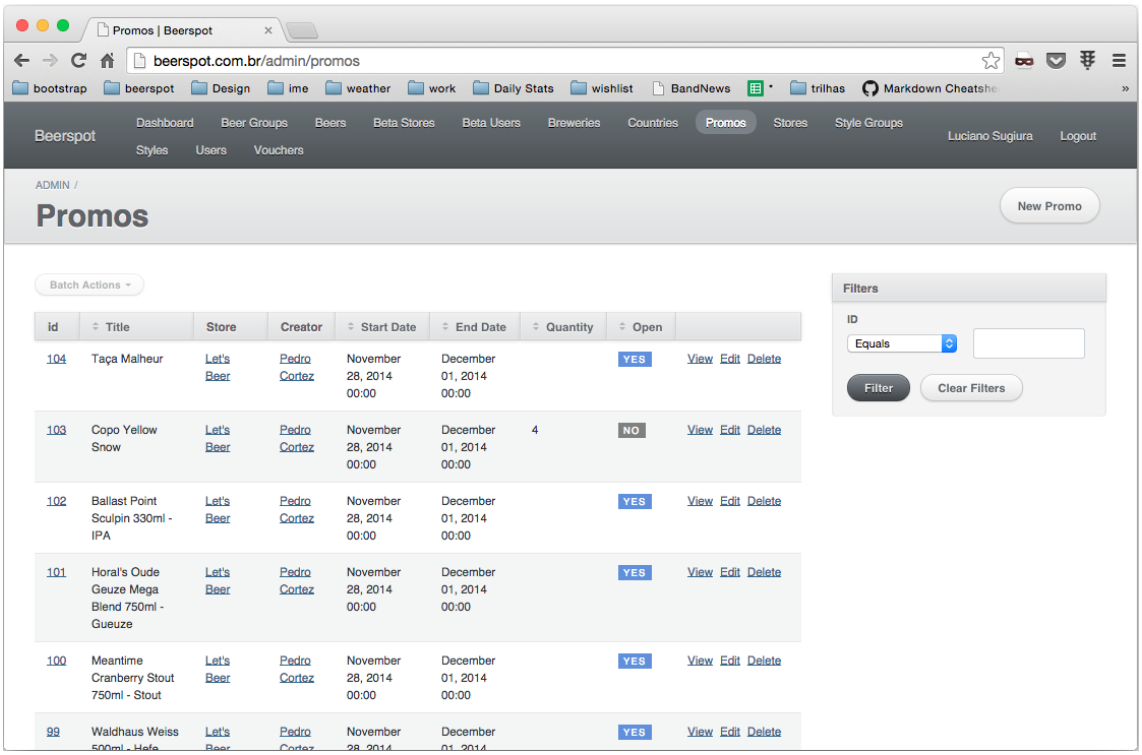

Figura 6: Página gerada pelo ActiveAdmin para controle de promoções (entidade *Promo*)

Os dados de resposta dos *web services* são entregues no formato JSON (*JavaScript Object Notation*), um formato enxuto e amplamente utilizado.

O Ruby on Rails, mais o componente JBuilder [10] foram especialmente úteis para essa implementação, facilitando muito a construção das respostas no formato JSON.

Um exemplo de resposta no formato JSON, para a chamada do *web service* com URI http://beerspot.com.br/api/v1/promos:

```
{
  "status": 0,
  "promos": [
    {
      "id": 95,
      "title": "Ballast Point Wahoo",
      "description": "Uma witbier turva, refrescante, com aroma e sabor
       de cascas de laranja e semente de coentro, seguindo as tradições
        belgas.",
```

```
      "start_date": "2014-11-28 00:00:00 +0000",
      "end_date": "2014-12-01 01:59:00 +0000",
      "created_at": "2014-11-28 01:25:06 +0000",
      "status": "active",
      "photo": "http://beerspot.com.br/system/promos/photos/000/000/095/
      original/Ballast_Point_Wahoo_White.jpg?1417137906",
      "details": "Garrafa com 330ml.\r\n\r\nValor exclusivo para
       usuários Beerspot.\r\nPara aproveitar a oferta, é necessário
        apresentar o cupom gerado, no caixa da loja.\r\nValidade da
         oferta: enquanto durarem os estoques.",
      "price_from": "R$ 20,00",
      "price_to": "R$ 10,00",
      "store": {
        "id": 1,
        "name": "Let's Beer",
        "phone": "11 2589-9695",
        "address": "R. Joaquim Távora, 961 - Vila Mariana, São Paulo - SP",
        "latitude": -23.5865094,
        "longitude": -46.6416502
      },
      "beers": []
    }
  ]
}
```


O uso desses *web services* está limitado a clientes que possuem um identificador, ou *token*. Atualmente, a criptografia das chamadas e respostas não está sendo utilizada, mas será implantada futuramente, utilizando o *Secure Socket Layer*.

6.6 Aplicativo para iOS

O aplicativo para iOS foi desenvolvido com as funcionalidades necessárias para o primeiro MVP: telas para a exibição de ofertas, autenticação e geração de cupons.

6.6.1 Visão geral do aplicativo

O aplicativo foi dividido em abas, como mostra a Figura 7.

Figura 7: Tela inicial do aplicativo iOS.

As abas correspondem às seguintes telas/funcionalidades:

- Exclusivas

Ofertas exclusivas do Beerspot. Estas ofertas exigem que o usuário gere um cupom para a utilização e mostre o mesmo para o lojista no ato da compra. Não são divulgadas em outros meios senão o aplicativo Beerspot.

- Outras

  Ofertas divulgadas publicamente na loja, veiculadas também pelo Beerspot, mas sem necessidade de gerar cupons.

- Meus Cupons

  Mostra uma lista de todos os cupons já gerados pelo usuário.

- Perfil

  Dados da conta do usuário, atualmente utilizada somente para entrar e sair da conta.

- Feedback

  Aba criada para usuários *beta* poderem responder a um questionário sobre o aplicativo.

6.6.2 Ofertas exclusivas

As ofertas exclusivas são o foco principal deste MVP. Através delas, validaremos o mecanismo utilizado para consumo de ofertas, bem como seu poder atrativo no mercado de cervejas especiais.

Na tela de ofertas exclusivas (Figura 7), toques no botão “VER DETALHES” ou na foto da oferta abrem a tela de detalhes da oferta, mostrada nas Figura 8, Figura 9 e Figura 10.

Figura 8: Tela de detalhes da oferta: valores, título e descrição.

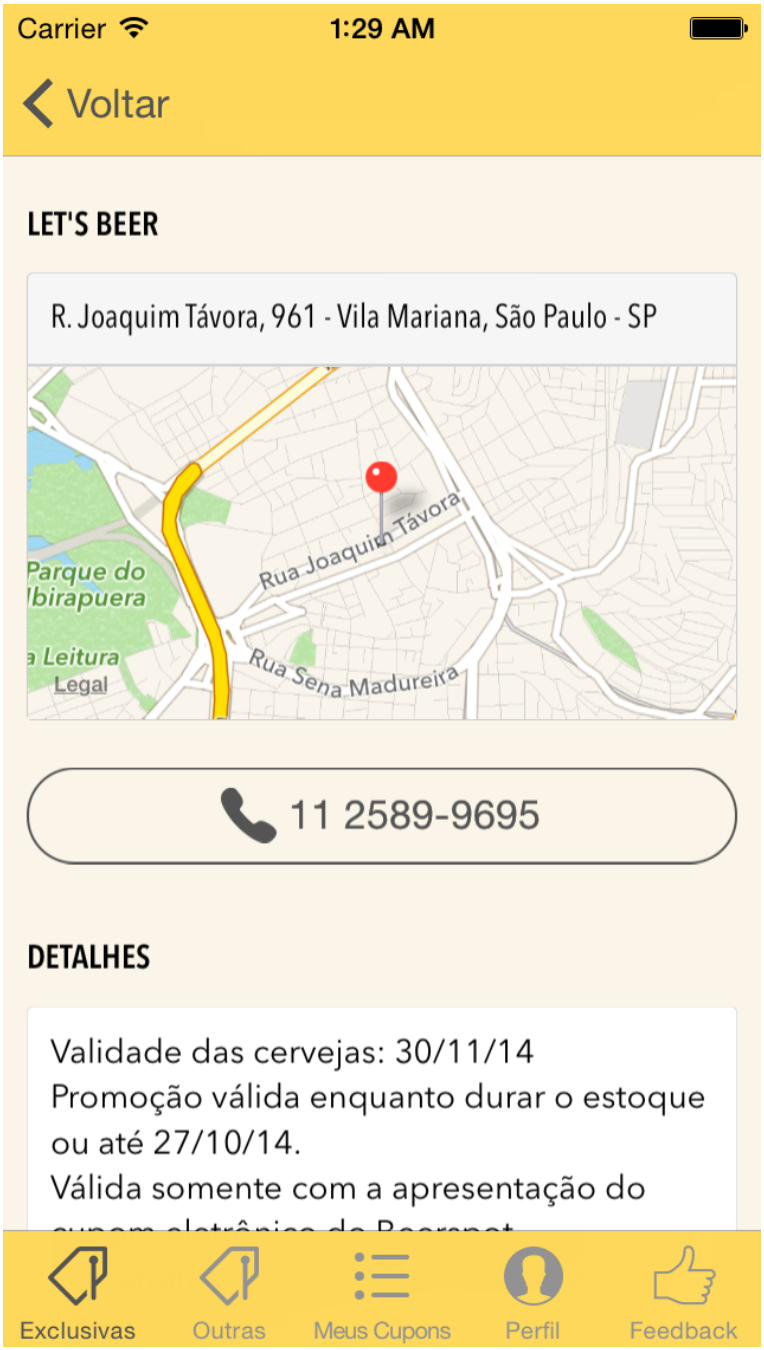

Figura 9: Tela de detalhes da oferta: detalhes da loja.

Figura 10: Tela de detalhes da oferta: detalhes da oferta e descrição breve das cervejas participantes.

Na tela de detalhes da oferta, são apresentados os valores do *kit*, o título completo, uma descrição do *kit*, detalhes da loja (nome, endereço, mapa e telefone), um texto

com detalhes da oferta e as cervejas participantes da oferta.

6.6.3 Geração de cupons

Nas telas de listagem de ofertas (Figura 7) e detalhes da oferta (Figura 8), é possível gerar cupons através do botão "QUERO APROVEITAR". Ao tocar nele, é apresentada a tela de confirmação. Ao clicar em "GERAR CUPOM", é solicitada a quantidade de *kits* desejados e, ao confirmar, o sistema gera um código, exibido em seguida. Este processo é mostrado em Figura 11, Figura 12 e Figura 13.

Figura 11: Gerar cupom: tela inicial para a geração do cupom.

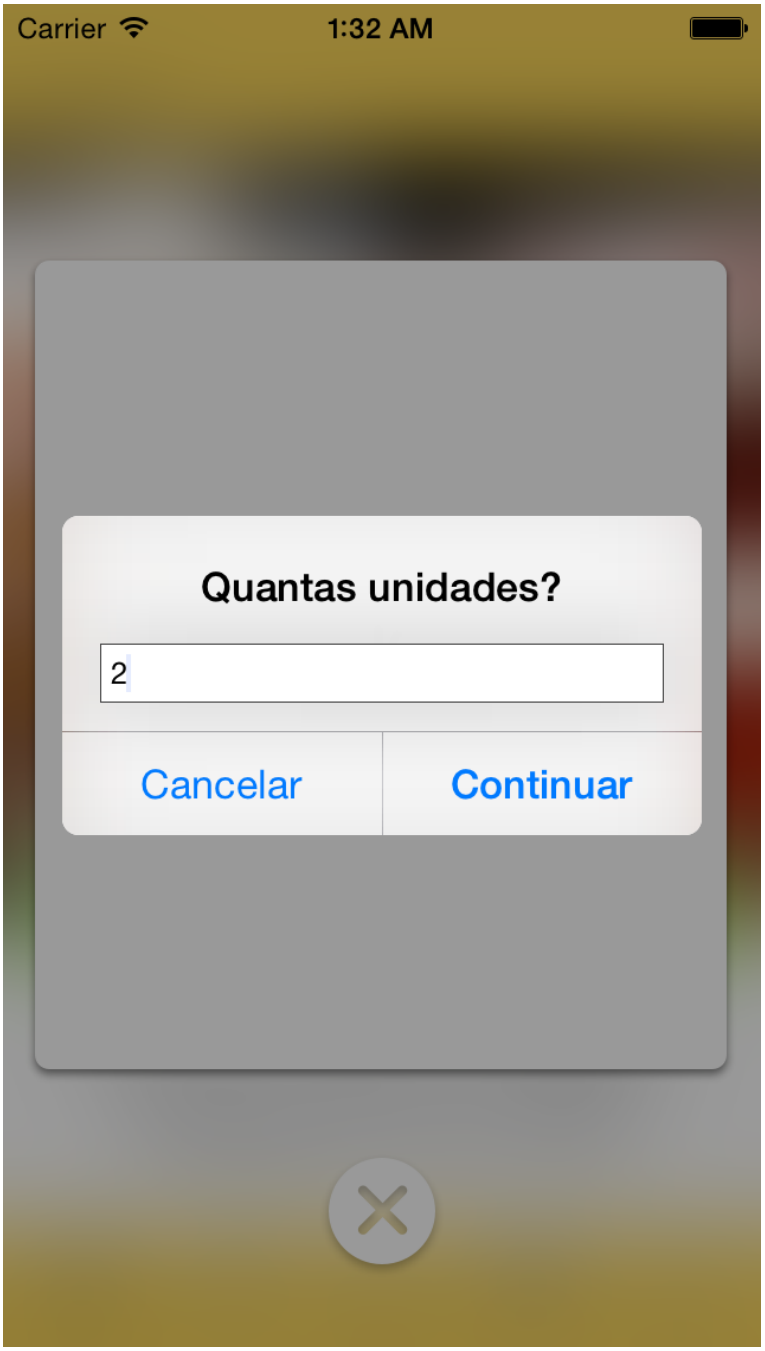

Figura 12: Gerar cupom: inserir quantidade de *kits* desejados.

Figura 13: Gerar cupom: código gerado pelo sistema e exibido no aplicativo.

6.6.4 Outras ofertas

Além das ofertas exclusivas, foi identificada a necessidade de divulgar ofertas abertas ao público, ou seja, exibidas na própria loja, sem exclusividade nem controle de quantidade.

Para estas ofertas, foi criada uma tela específica, com uma exibição simplificada das ofertas, como mostra a Figura 14.

Figura 14: Outras ofertas: ofertas sem exclusividade, exibidas de modo simplificado.

6.6.5 Meus cupons

Nesta tela, todos os cupons exibidos pelo usuário são listados, com alguns detalhes mais o status de cada cupom, como mostrado na Figura 15. Os status possíveis são ativo, utilizado e expirado. Um cupom expira em um dia e, ao expirar, a quantidade reservada retorna à oferta para futuras utilizações. Ao tocar em um item da listagem, o usuário visualiza a tela do cupom gerado, com o código para utilização (Figura 13);

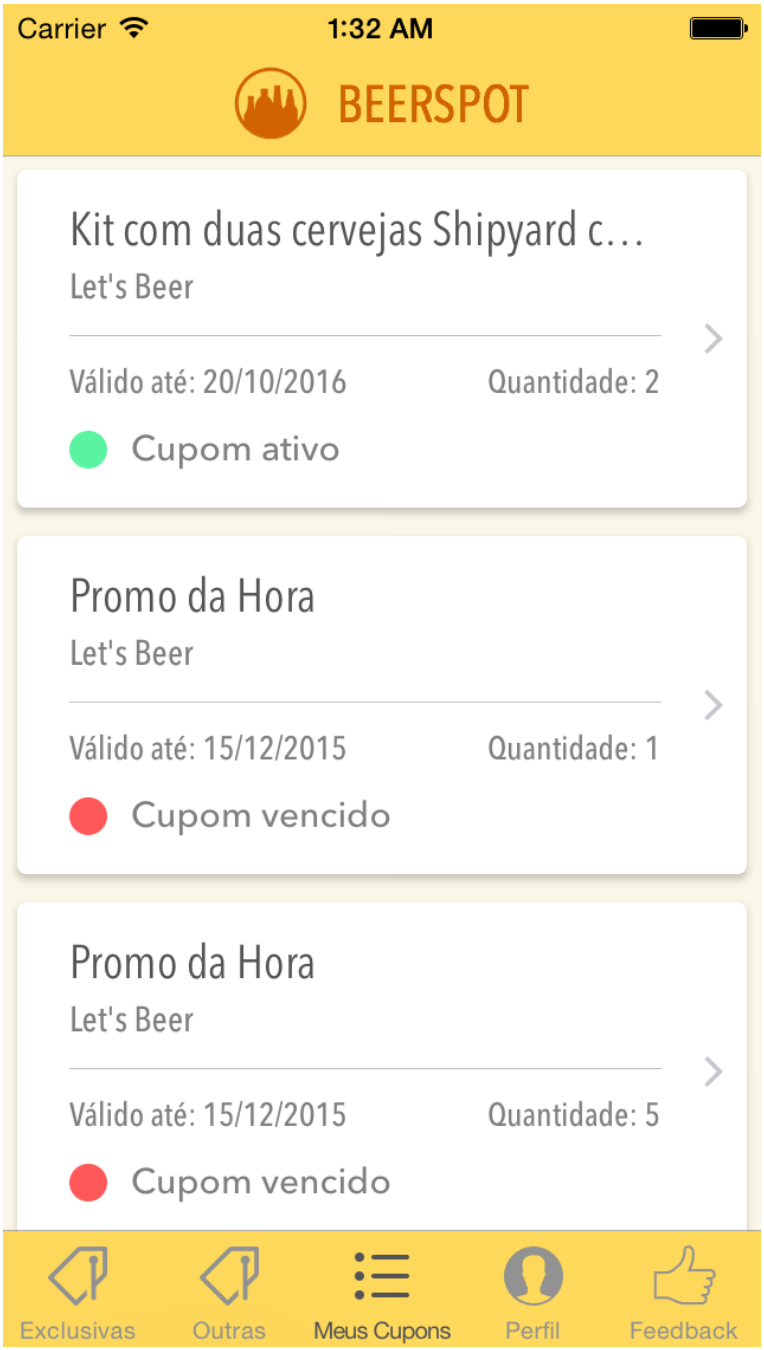

Figura 15: Meus cupons.

6.6.6 Perfil

Na tela de perfil, quando o usuário está logado no sistema, são exibidos o nome, a foto (quando o login foi feito via Facebook) e um botão para logout (Figura 16). Quando nenhum usuário está logado, é exibido o botão para login (Figura 17).

Figura 16: Tela de perfil com usuário logado.

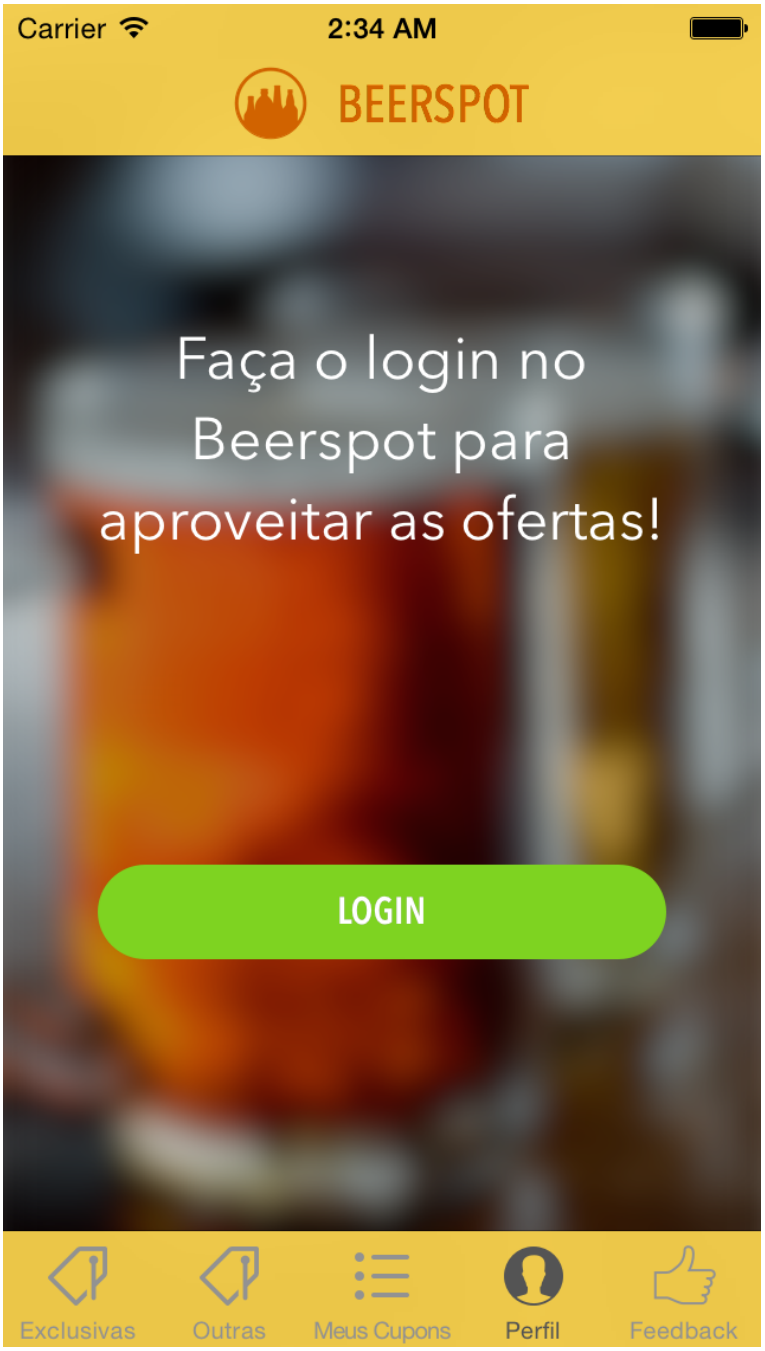

Figura 17: Tela de perfil sem usuário logado.

6.6.7 Cadastro e login

O cadastro do usuário pode ser realizado por uma conta no Facebook ou por um formulário com nome, e-mail e senha.

O login pode ser realizado também pela conta no Facebook, ou fazendo a autenticação por e-mail e senha.

Estas funcionalidades são mostradas em Figura 18, Figura 19 e Figura 20.

Figura 18: Login: tela principal, com opção de uso de conta do Facebook.

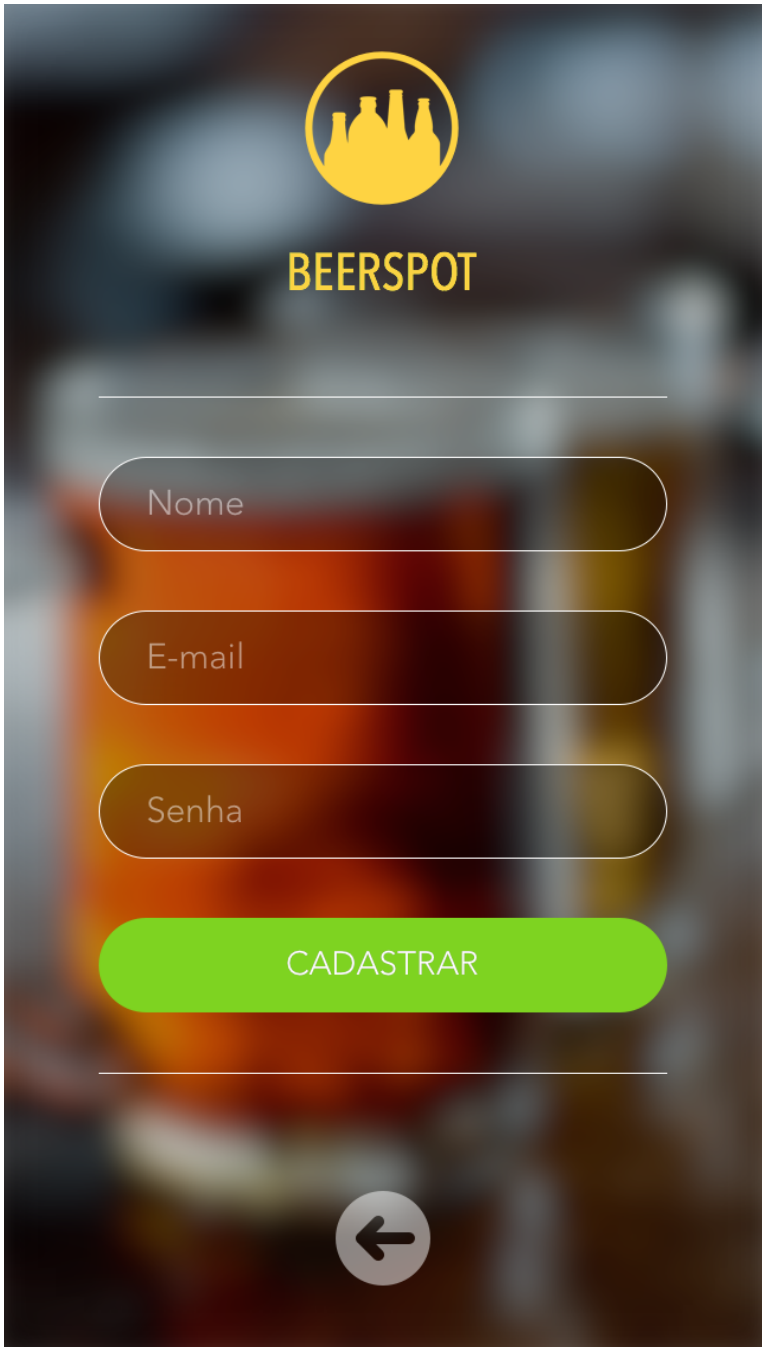

Figura 19: Login: cadastro via formulário.

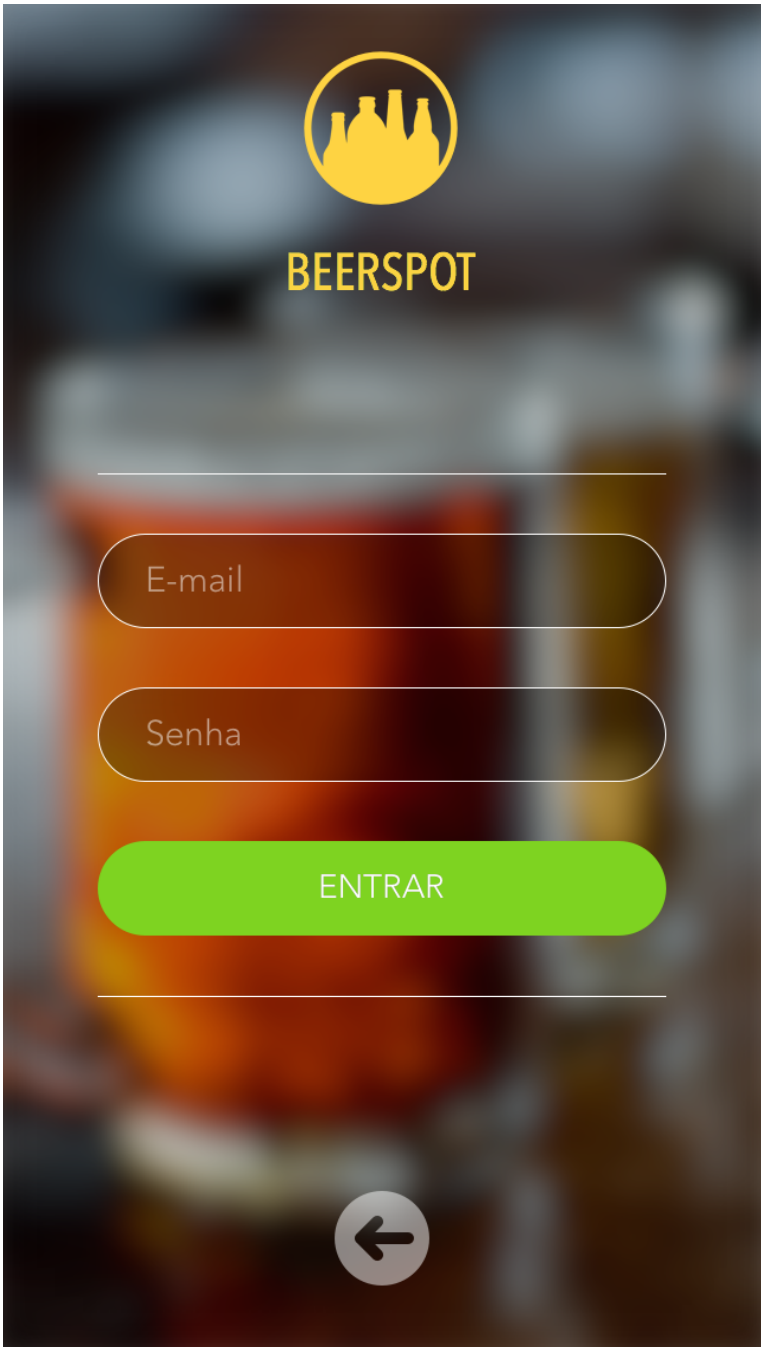

Figura 20: Login: autenticação por e-mail e senha.

6.6.8 *Feedback*

Para fins de pesquisa com usuários, foi criada também uma tela de *feedback*. Nela é exibido um formulário simples, através do Google Forms [8], como mostra a Figura 21.

Figura 21: O formulário para *feedback*.

6.6.9 Oferta esgotada e validações

Quando a quantidade de *kits* de uma oferta acaba, é exibido um selo na listagem de ofertas, e não é mais possível gerar cupons, como mostra a Figura 22.

Figura 22: Uma oferta esgotada.

Para evitar problemas de concorrência, são realizadas validações também no ato da geração de cupons. Quando a quantidade de *kits* é menor que a solicitada, é mostrada a mensagem mostrada na Figura 23 e quando a oferta está esgotada, a

mensagem na Figura 24.

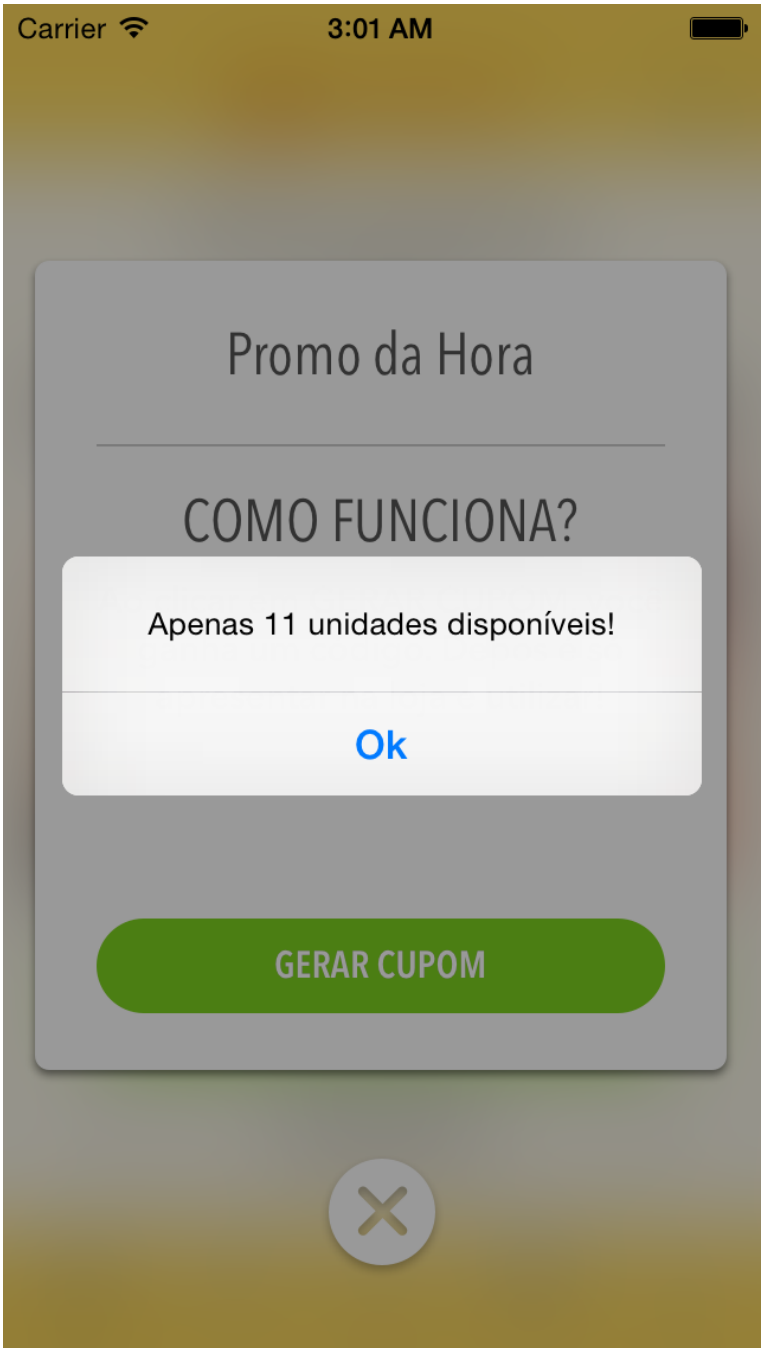

Figura 23: Solicitando uma quantidade maior que a disponível.

Figura 24: Solicitando cupom para uma oferta esgotada.

6.7 Aplicativo para Android

O aplicativo para Android está sendo desenvolvido com as mesmas funcionalidades do aplicativo para iOS.

6.8 Painel do lojista

O painel do lojista está em desenvolvimento a partir do protótipo mostrado na Figura 25.

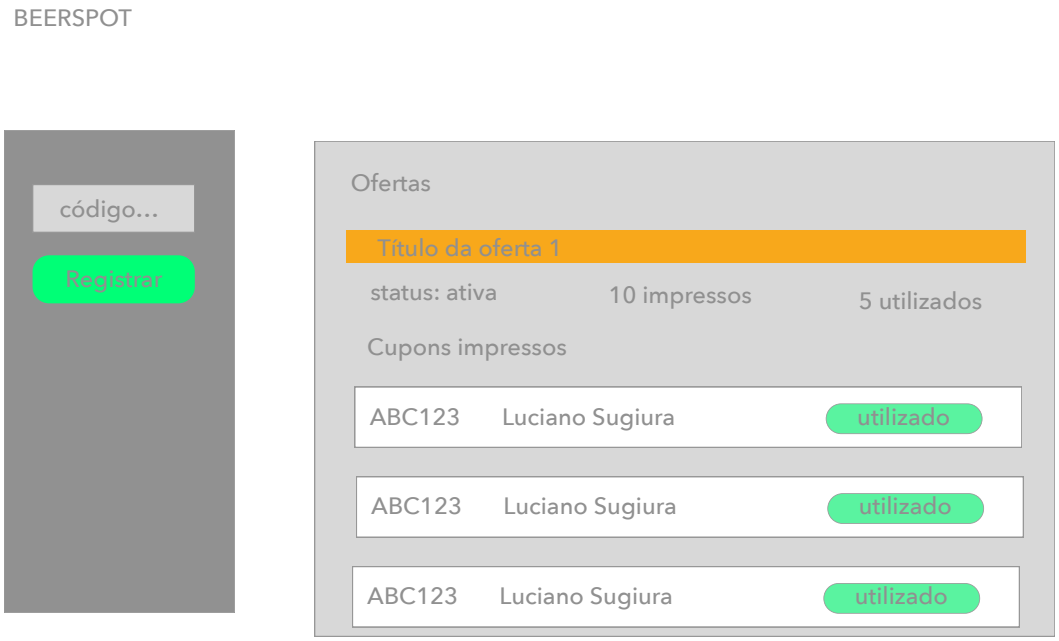

Figura 25: Protótipo do painel para lojista.

No painel, serão exibidas as ofertas ativas para a loja em questão, com detalhes como o status da oferta, quantos cupons foram impressos, quantos foram utilizados, uma lista dos cupons gerados e um campo para marcar um cupom como utilizado a partir do código.

6.9 *Test Driven Development*

A implementação do *backend*, os modelos (classes que definem entidades), *controllers* (responsáveis pela construção das respostas de páginas e *web services*) e *views* (lógica para montagem das respostas) foram implementados seguindo a técnica chamada *Test Driven Development*, ou simplesmente TDD. Nesta técnica, uma implementação segue rigorosamente esta ordem:

- define-se um conjunto de testes para esta funcionalidade. Executar os testes evidentemente deve resultar em erros;
- implementar a funcionalidade;
- executar os testes, verificando que a funcionalidade os resolveu.

Esta abordagem força uma cobertura de praticamente a totalidade do código por testes, fazendo com que a qualidade do *software* chegue a um nível altíssimo, uma vez que qualquer implementação que insira erros em qualquer parte do sistema é identificada pela falha no conjunto de testes

Durante o projeto foram vivenciados inúmeros casos desse tipo, em que a implementação de um componente não relacionado a outro gerou erros neste, o que não seria percebido facilmente sem a abordagem TDD.

Para o projeto Ruby on Rails, foi utilizado o arcabouço para testes RSpec [16].

Os aplicativos não receberam cobertura de testes, o que é muito desejado e está planejado para o futuro do projeto.

6.10 A equipe Beerspot

O projeto Beerspot como uma *startup* está sendo executado por 3 pessoas:

- o autor do trabalho, Luciano Kenji Sugiura, atuando como líder de produto e desenvolvedor;
- o orientador do trabalho, Fabio Kon, atuando como consultor do projeto, e
- Pedro Cortez, formado como Beer Sommelier pelo Instituto da Cerveja [9], atuando como diretor de negócios e estratégia.

Atuar em equipe também foi objetivo do projeto como no desenvolvimento de qualquer *startup*, o que foi realizado através de inúmeras reuniões ao longo do projeto. Estas reuniões, em sua maioria entre os 3 integrantes, resultou em definições cada vez mais precisas sobre a *startup*, em especial o modelo de negócios e as definições de escopo do projeto como TCC e do primeiro MVP.

6.11 O *design* de interfaces

Um dos desafios encontrados pelo autor foi a criação de interfaces de usuário para o aplicativo para iOS. Dada a pouca experiência, foi necessário buscar ferramentas, aprendizado e inspiração para criar as primeiras telas do aplicativo.

A ferramenta utilizada para prototipação das telas foi o Sketch 3, da empresa Bohemian Coding [19]. Ela é paga (99 dólares), foi desenvolvida especificamente para a prototipação de telas e já possui uma comunidade extremamente ativa de usuários. É possível encontrar diversos trabalhos abertos e de uso gratuito que servem como base para a prototipação de novas telas. Um exemplo é o *site* especializado Sketch App Resources [20].

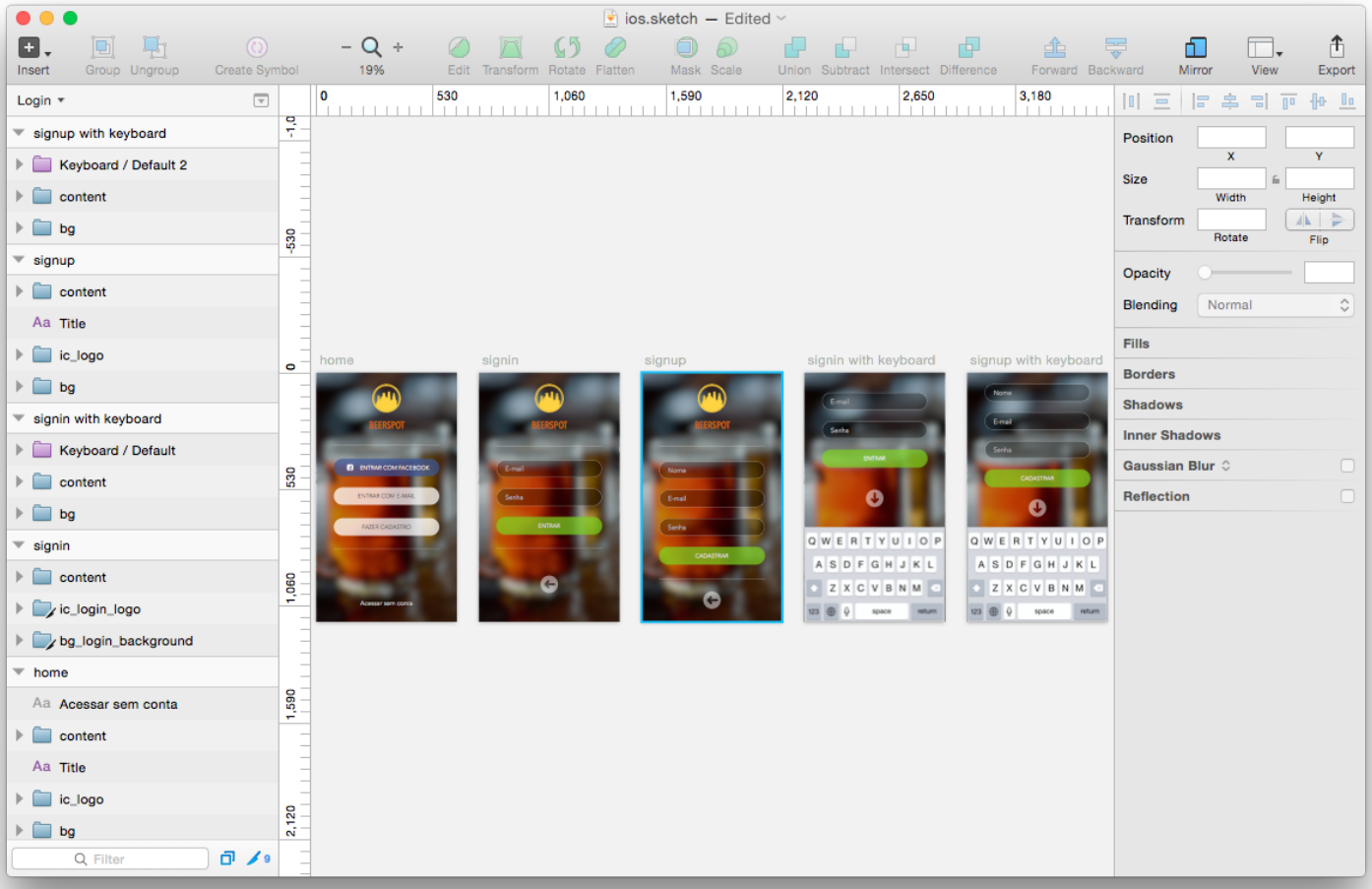

Figura 26: Prototipando as telas de login com o Sketch.

7 Conclusões

7.1 O Trabalho de Formatura Supervisionado

O aprendizado sobre as várias tecnologias envolvidas, sua dificuldade e a experiência obtida eram objetivos do projeto e foram plenamente cumpridos. Esse aprendizado e a implementação foram os itens que mais consumiram tempo no projeto, deixando claro como a capacitação técnica é essencial para a criação de uma *startup*. Os prazos estimados para o término de várias etapas não foram cumpridos devido à essa dificuldade, mas isso ajudou a melhorar as estimativas de tempo de implementação ao longo do tempo, o que é fundamental para a *startup*.

O objetivo de obter a experiência ao criar um MVP e colocá-lo à prova no mundo real, incluindo as etapas envolvendo clientes, foi bem cumprido, apesar de o tempo após implementação até o término do TCC não ter sido suficiente para realizar várias iterações de produto e relatar seus resultados. Porém, a metodologia desenvolvida em *The Lean Startup* foi vivenciada em boa parte e será fundamental para futuras iterações do projeto, seguindo o processo Construir-Medir-Aprender.

Desde o início, o mercado alvo da *startup* foi sendo estudado e seus problemas e necessidades constantemente observados, para poder embasar cada vez mais fortemente a validação de que o produto é realmente algo quisto pelos consumidores.

Atualmente, o primeiro MVP está sendo testado e está sendo bem recebido, tanto pelos lojistas, quanto pelos consumidores, praticamente validando positivamente as primeiras hipóteses e possibilitando a continuidade do negócio.

O ciclo Construir-Medir-Aprender, portanto, tem se mostrado extremamente valioso neste primeiro momento, e sua aceleração está sendo tratada como alvo para melhorar a *startup* rápida e constantemente.

Durante o trabalho, foram realizadas diversas reuniões com o orientador, sempre de modo produtivo, resultando em definições de escopo cada vez mais detalhadas e estratégias mais bem definidas.

Assim, considera-se que o trabalho foi bem realizado e sucedido.

7.2 O Beerspot no ecossistema de *startups* e seu futuro

Um ecossistema de *startups* é um conjunto de todos os elementos envolvidos para o nascimento e sucesso de *startups*. Dentre estes elementos estão as ideias, invenções e pesquisas, as próprias *startups* em seus vários estágios, os empreendedores, investidores, mentores e consultores, as universidades e outros diversos colaboradores. Um estudo aprofundado foi realizado em Israel [23] para entender quais são os fatores e elementos chave responsáveis por promover o crescimento de um ecossistema de *startups* bem sucedido. Este estudo resultou na criação de um arcabouço conceitual,

que identifica os elementos chave para o desenvolvimento bem sucedido do ecossistema de *startups* no contexto de Israel.

O Beerspot nasceu também com o objetivo de se tornar uma *startup* de sucesso. O projeto será continuado, espera-se que sejam obtidos ótimos resultados e que, inclusive, ele seja destaque e colabore no ecossistema de *startups* em São Paulo.

8 Parte subjetiva

Neste capítulo descreverei como foi o desenvolvimento do projeto em relação a todo o conteúdo adquirido durante o curso.

8.1 O projeto como um complemento do BCC

Como descrito na introdução, um dos claros objetivos foi: conhecer e implementar todas as partes de um sistema de uso real. Por si só, esse objetivo demonstra a minha principal insatisfação em relação ao curso.

Apesar de ainda não considerar ter muita experiência no mercado de trabalho da área, considero o curso defasado em relação ao aprendizado de tecnologias de ponta.

Tentarei explicar essa defasagem com um exemplo prático. No projeto, havia uma dúvida em relação à utilização da linguagem Python, com arcabouço Django, ou da linguagem Ruby, com arcabouço Ruby on Rails. Os critérios para escolha do Ruby on Rails foram embasados somente em pesquisas na internet. Ambas são bastante modernas e oferecem soluções baseadas nas melhores práticas. No curso, aprendi pouquíssimo sobre arcabouços (apenas na disciplina Middleware) e quais são os melhores padrões utilizados por eles, qual o estado da arte e quais são as melhores opções existentes. Em vez disso, estudei muito sobre mecânica, termodinâmica e relatividade. Por quê? Na minha concepção, física está na grade por exercitar conceitos importantes que aprendemos nas disciplinas mais elementares. Mas por que exercitar conceitos na física e não e áreas da computação importantes, como esta dos arcabouços *web*?

Isso se repete em outras áreas. A disciplina de banco de dados expõe conceitos importantíssimos, mas a utilização de um sistema gerenciador de banco de dados como o PostgreSQL, para aplicação destes conceitos, não é obrigatória. Banco de dados são utilizados em quase todo tipo de sistema computacional, e seu aprendizado prático também é pouquíssimo priorizado no curso.

Por estes motivos, criar um projeto que exercite diversas áreas cujo conceito aprendi, mas a prática não existiu, foi bastante benéfico e importante para minha formação.

8.2 Disciplinas importantes

As disciplinas base foram absolutamente necessárias para uma compreensão plena das implementações dos produtos do projeto. Em “MAC0110 Introdução à computação”, “MAC0122 Princípios de desenvolvimento de algoritmos” e “MAC0323 Estrutura de dados”, o aprendizado de conceitos fundamentais e sua prática através dos EPs foram essenciais para a minha formação como programador. Um exemplo

claro é a utilização de ponteiros nas estruturas de dados. A linguagem Objective-C utiliza ponteiros extensivamente, o que seria dificultado se eu não tivesse adquirido esses conceitos.

"MAC0412 Organização de computadores" foi importante por trazer detalhes de utilização de hardware que não são explanados em nenhuma outra disciplina. Entender melhor como processador, memória, registradores e cache interagem entre si trouxe diversos benefícios na compreensão de implementações que lidam com recursos mais escassos.

Infelizmente não cursei a disciplina "MAC0441 Programação Orientada a Objetos". Estudei seus conceitos brevemente e considero indispensável para a formação de um aluno de BCC. Os *design patterns* estudados são extensamente utilizados e, o mais importante, refletem o estado da arte em muitos casos de implementação. Aprendemos uma infinidade de teoremas e técnicas matemáticas obrigatoriamente, mas os *design patterns*, que são técnicas na criação de *software*, são deixados como disciplina optativa, o que acredito ser algo muito equivocado. Lido diariamente com *patterns*, e o estudo de seus conceitos com certeza ajudaria no meu projeto.

"MAC0316 Conceitos Fundamentais de Linguagens de Programação" ajudou muito no entendimento de diferenças entre linguagens como Java, Objective-C e Ruby. No curso, foi utilizada a linguagem Scheme que, por ter um paradigma diferente, ajudou na diversificação dos conhecimentos sobre linguagem e no aprendizado de linguagens novas.

"MAC0422 Sistemas Operacionais" foi importante principalmente por tratar de conceitos que não teríamos na simples utilização dos sistemas. Escalonamento, concorrência e gestão de recursos são conceitos importantes e que obtemos, em grande parte, nessa disciplina.

8.3 O projeto e o futuro após o curso

Programação tem segmentos diversos e ter feito um curso como o BCC no IME dá, com certeza, uma ótima bagagem conceitual para trabalhar em qualquer tópico na área. O segmento de aplicativos, em especial, está em crescimento constante desde a sua concepção, e é um segmento bastante atraente, tanto do ponto de vista de oportunidades quanto de interesse pelo assunto.

O desenvolvimento deste projeto forneceu uma experiência bastante ampla, dando autonomia para que eu possa iniciar qualquer tipo de projeto no segmento sem maiores dificuldades.

Portanto, dados o interesse pelo segmento e a experiência adquirida, pretendo continuar o projeto desenvolvido e seguir com um aprendizado constante para prosperar na área.

Referências

[1] ActiveAdmin. http://activeadmin.info/.

[2] Apache vs nginx. http://webnyze.com/market-share/load-balancers/apache-vs-nginx/index.html.

[3] Business Model Generation. http://businessmodelgeneration.com/book?_ga=1.39630113.34535240.1417283694.

[4] Com o copo cheio. http://exame.abril.com.br/revista-exame-pme/edicoes/0076/noticias/com-o-copo-cheio.

[5] Companhia startup. https://pt.wikipedia.org/wiki/Companhia_startup.

[6] Digital Ocean. https://www.digitalocean.com/.

[7] Dreamhost Web Server Performance Comparison. http://wiki.dreamhost.com/Web_Server_Performance_Comparison.

[8] Google Forms. http://www.google.com/forms/about/.

[9] Instituto da cerveja. http://institutodacerveja.com.br/.

[10] Jbuilder. https://github.com/rails/jbuilder.

[11] Mercado de cervejas premium no Brasil está em franca fermentação. http://g1.globo.com/rj/regiao-serrana/noticia/2014/04/mercado-de-cervejas-premium-no-brasil-esta-em-franca-fermentacao.html.

[12] nginx. http://nginx.org/.

[13] Phusion Passenger. https://www.phusionpassenger.com/.

[14] Puma Web Server. http://puma.io/.

[15] Rails ERD. https://github.com/voormedia/rails-erd.

[16] RSpec. http://rspec.info/.

[17] Ruby on Rails. http://rubyonrails.org/.

[18] Running Lean. http://runninglean.co/.

[19] Sketch 3. http://bohemiancoding.com/sketch/.

[20] Sketch App Resources. http://www.sketchappsources.com/.

[21] The Ruby Programming Language. https://www.ruby-lang.org/.

[22] Webnyze. http://webnyze.com/.

[23] Fabio Kon, Daniel Cukier, Claudia Melo, Orit Hazzan, and Harry Yuklea. A Panorama of the Israeli Software Startup Ecosystem. Março 1 2014.

[24] Ash Maruya. Why Lean Canvas vs Business Model Canvas? http://practicetrumpstheory.com/why-lean-canvas/.