
title: RELATÓRIO DE CUMPRIMENTO DO OBJETO ETAPA III - Ecossistemas de Software Livre - Agosto 2018

author: Carla Silva Rocha Aguiar (Coordenadora do Projeto)

date: 08 de Agosto de 2018

geometry: "left=3cm,right=3cm,top=2cm,bottom=2cm"

colorlinks: true

Introdução

O presente relatório apresenta o acompanhamento do trabalho realizado no projeto "Ecossistemas de Software Livre", Termo de Cooperação para Descentralização de Crédito, Processo Ofício No 0646/2017/FUB-UnB, Vigência Outubro 2017 à Outubro 2019. O relatório apresentado é referente aos avanços realizados na Etapa III (Abril 2018 à Junho 2018), de acordo com o cronograma do Plano de Trabalho.

Toda alteração no cronograma proposto foi realizada a partir de renegociação com a CGTEC do Ministério da Cultura, e tais alterações estão descritas no relatório.

FASE DE PLANEJAMENTO/EXECUÇÃO

O período de Abril 2018 à Junho de 2018 contemplou as fases de planejamento e execução. Abaixo serão apresentados, brevemente, os principais avanços alcançados no período. Toda a documentação e acompanhamento do projeto está disponibilizado e pode ser acessado na organização do laboratório

[lappis-unb](#), e no

repositório específico do projeto

[lappis-unb/EcossistemasSWLivre](#).

Todo o planejamento e execução das tarefas podem ser acompanhados tanto nas *issues* quanto nas páginas *wiki*.

Na segunda etapa, houve em conjunto a CGTEC a alteração do cronograma do projeto, no qual a frente de trabalho "Catálogo de Software" foi retirado do projeto. No presente relatório está disponibilizado o novo cronograma, com atualização citada.

Abaixo serão apresentados os principais avanços alcançados no período, por pacote de trabalho de acordo com o Plano de Trabalho e cronograma.

Legado em Software Livre

As macro atividades referentes a essa etapa de acordo com o cronograma do plano de trabalho são:

- ☒ Realizar Estudo de Refatoração em software legado
- ☒ Realizar Estudos sobre práticas de DevOps aplicada a software legado

Durante a primeira etapa do projeto foi priorizado a visão "legacy in the box" (legado em uma caixa, tradução literal), no qual o foco foi isolar alguns projetos mantidos pelo Ministério da Cultura por meio de Docker[^docker]. Essa solução gera o benefício de criar ambientes de desenvolvimento e produção estáveis, fazendo com que diminua o tempo de configuração de ambiente. Essa abordagem traz um grande benefício pois possibilita o uso de práticas DevOps mesmo em sistemas legados. Esse modelo de isolar pacotes de software legados através de containers Docker possibilita um pipeline de entrega contínua, deploy contínuo, e diminui a fronteira entre a equipe de infraestrutura e equipe de desenvolvimento.

[^docker]: Docker fornece uma camada adicional de abstração e automação de virtualização de nível de sistema operacional. <http://www.docker.com>

Na segunda etapa do projeto, usamos uma segunda forma de lidar com software legado, sempre com o intuito de aplicar técnicas modernas de Engenharia de Software e padrões de comunidade de software livre, a fim de viabilizar o uso desses projetos legados em outros contextos e em pipelines automatizados. O foco então foi transformar um software legado em software livre, a partir de técnicas de refatoração de código, e suite de testes automatizados. Foi escolhido a API do Salic como estudo de caso, uma vez que esse é um sistema relativamente pequeno, de grande relevância e impacto no ecossistema Salic.

As ações realizadas na terceira etapa (referente ao presente relatório) para atender os objetivos do plano de trabalho foram:

- ☒ Homologar a API do Salic no ambiente do MinC;
- ☒ Estudo de estratégias possíveis para a implementação de novas funcionalidades em um sistema legado/monolítico;
- ☒ Proposta de uma estratégia pra implementação de novas funcionalidades no SALIC (Estudo de caso).

A terceira etapa do projeto teve como objetivo inserir o software refatorado na etapa passada

em um pipeline de entrega e deploy contínua. Após uma homologação interna na própria infraestrutura do laboratório LAPPIS, realizada na etapa anterior, foi realizada a entrega técnica ao ministério, no qual foi apresentada a estratégia de refatoração e a versão inicial da API do Salic. A API foi então homologada na infraestrutura do Ministério da Cultura. Foi configurado um pipeline de deploy/entrega contínua, no qual toda a mudança no código da API passa por um pipeline no Gitlab-CI automatizado até um ambiente de homologação no MinC. Todo o pipeline pode ser acompanhado em <https://github.com/lappis-unb/salic-api>.

Nessa etapa também teve como objetivo principal a realização de uma pesquisa teórica e prática sobre alternativas para lidar com sistemas legados para a adição de novas funcionalidades. Foram estudadas duas possíveis abordagens: (a) adição de novas funcionalidades mantendo a arquitetura monolítica (b) adição de novas funcionalidades utilizando o modelo arquitetural microserviços.

Uma primeira estratégia é manter um repositório e sistema monolítico, somente acrescentando novas funcionalidades respeitando boas práticas de software (qualidade de software, testes unitários, testes de aceitação, etc). Dentre as vantagens dessa abordagem estão:

- **Centralização:** toda a base de código é contida em um único repositório com suas diversas funcionalidades;
- **Visibilidade:** código é visível e pesquisável por todos os engenheiros da organização e contribuidores externos;
- **Sincronização:** o processo de desenvolvimento é orientado a branches, e colaboradores contribuem para branches específicas no repositório;
- **Completeness:** qualquer projeto/funcionalidade do repositório pode ser compilado a partir das dependências presentes no próprio repositório. Dependências são versionadas, projetos devem usar a versão de suas dependências no repositório;
- **Padronização:** um conjunto de ferramentas compartilhadas governam como colaboradores interagem com o código, incluindo testes, pesquisa e revisão de código.

Uma segunda estratégia é adotar uma arquitetura de microserviços, no qual a componentização de novas funcionalidades é feita por meio de serviços. Microserviços são componentes executados de forma independente (out-of-process), que comunicam entre si com mecanismos como requisição de serviço web, ou chamadas de processos remotos. Uma das principais razões de usar serviços como componentes é que serviços são disponibilizados (deploy) de forma independente. Se uma aplicação monolítica (executando apenas um processo) alterar um único componente, isso resulta na necessidade de realizar o deploy de toda a aplicação. Dentre as vantagens dessa abordagem estão:

- **Serviços organizados como capacidade de negócio:** cada serviço é responsável por uma capacidade do negócio, e é desenvolvida full stack (do design, requisitos,

implementação, implantação). Mais próximo ao movimento DevOps;

- **Produtos, não projetos:** time é responsável por todo o ciclo de vida do produto, do projeto à implantação;
- **Endpoints inteligentes:** Comunicação entre microserviços tende a ser altamente desacoplados e altamente coesos. Protocolos mais utilizados HTTP, protocolos RESTful;
- **Governança Descentralizada:** decisões técnicas e negociais são realizadas localmente, dependendo da necessidade de cada microserviço;
- **Gestão de dados descentralizados:** cada microserviço gerencia seu próprio banco de dados e lógica de armazenamento;
- **Automação da Infraestrutura:** todo o pipeline de deploy e entrega contínuos pode ser feito de forma automatizada, filtrada por meio de testes realizados em vários estágios do pipeline;
- **Projetado para falhas:** aplicações são projetadas para serem tolerantes à falha de serviços;
- **Design Evolutivo:** a aplicação pode ser evoluida gradativamente e continuamente, ao contrário de arquitetura componentizada que requer um projeto de como as funcionalidades serão divididas em componentes.

A arquitetura microserviço é um padrão dos sistemas de software modernos, e tem se tornado um padrão nas grandes empresas. Após essa revisão literária e revisão técnica sobre possíveis soluções arquiteturais para lidar com a adição de novas funcionalidades optamos por abordar a solução via arquitetura microserviços. Como estudo de caso, optamos por utilizar as funcionalidades desenvolvidas no pacote de trabalho "aprendizagem de máquina" que serão integradas ao Salic. Ou seja, as funcionalidades de aprendizagem de máquina que serão integradas ao Salic vão ser implementadas como microserviços. A próxima etapa do projeto essa abordagem de tratar com sistemas legados será implementado, colocado em ambiente de homologação/produção, e avaliado.

Práticas de gestão colaborativa

O objetivo de estudar os movimentos, organizações, desenvolvedores e demais stakeholders que atuam na gestão colaborativa de software livre é manter um conjunto de ações de governança digital e comunicação que aproveite ao máximo esse potencial em favor das necessidades do órgão e das metas comuns às organizações parte das comunidades. Esse esforço envolve um trabalho de mapeamento de atores de cada comunidade (atuais e potenciais futuros), assessoria para planejamento conjunto, facilitação de fluxos de comunicação e mobilização, realização de atividades conjuntas para integração, identificação de oportunidades externas, assessoria para comunicação e divulgação ao público externo à comunidade e apoio para solução de conflitos.

Ações programadas para esta etapa de acordo com o plano de trabalho:

- ☒ Identificar grupos de opinião.

As ações realizadas nesta etapa para atender os objetivos do plano de trabalho foram:

- ☒ Reuniões com a equipe técnica da SEFIC, a fim compreender o ecossistema do Salic (maior software mantido pelo ministério)
- ☒ Planejamento de evento para disseminação dos resultados e conhecimento dos laboratórios que colaboram com o Ministério da Cultura
- ☒ Visita técnica do Irio (Serenata de Amor)
- ☒ Apresentação no OSS (Paulo)

Aprendizado de Máquina Lei Rouanet

O principal objetivo é o estudo de técnicas de Aprendizado de Máquina que possam apoiar o sistema de recomendação e fiscalização da Lei Federal de Incentivo à Cultura (Lei Rouanet). Nessa etapa será realizada uma pesquisa exploratória em técnicas de aprendizado de máquina e processamento de linguagem natural. Tais técnicas e algoritmos serão aplicados para melhorar a experiência de usuário (UX) por meio da proposta de chatbots como interface entre os proponentes na Lei Rouanet e o Ministério da Cultura.

Além disso, técnicas de aprendizado de máquinas serão estudadas para automatizar processos nas trilhas de auditorias, tanto na etapa de habilitação e aprovação, quanto na etapa de prestação de contas. O objetivo é auxiliar auditores a encontrar erros, inconsistências e detectar anomalias nas submissões.

Ações programadas para esta etapa de acordo com o plano de trabalho:

- ☒ Realizar Estudo da Lei Rouanet/SALIC;
- ☒ Realizar Estudo de aprendizado de máquina;
- ☒ Realizar Estudo de processamento linguagem natural;
- ☒ Realizar Estudo de chatbots.

As ações realizadas nesta etapa para atender os objetivos do plano de trabalho foram:

- ☒ Reuniões com a equipe técnica da SEFIC para a compreensão do processo de execução da lei Rouanet
- ☒ Levantamento dos principais gargalos de execução da lei Rouanet no Salic
- ☒ Levantamento dos principais dados no Salic usados para monitoramento e controle do Ministério
- ☒ Proposta de solução utilizando algoritmos de aprendizagem de máquina para melhorar a execução da lei Rouanet no Salic

- ☑ Estudo de algoritmos de aprendizagem de máquina/estatísticas para extrair padrões/outliers do banco de dados
- ☑ Estudo de um processo para desenvolver produtos de softwares com módulos de aprendizagem de máquina
- ☑ Proposta arquitetural de microserviços para disponibilizar as soluções de aprendizagem de máquina no Salic
- ☑ Estudo de processamento de linguagem natural para extrair dados de notas fiscais
- ☑ Implementação do conector Rasa com o Rocket.Chat
- ☑ Estudo de usabilidade (UX) de fluxo de conversas em chatbot
- ☑ Proposta de um novo fluxo de conversa da Rouana
- ☑ Validação interna do novo fluxo de conversa da Rouana

Todo o acompanhamento técnico e documentação evidenciando as atividades citadas acima estão disponibilizados nos repositórios do projetos (commits, issues, wiki), sob licença GPL3: <https://github.com/lappis-unb/salic-ml/tree/master/salicml>, <https://github.com/lappis-unb/rouana>, <https://github.com/lappis-unb/EcosystemasSWLivre>.

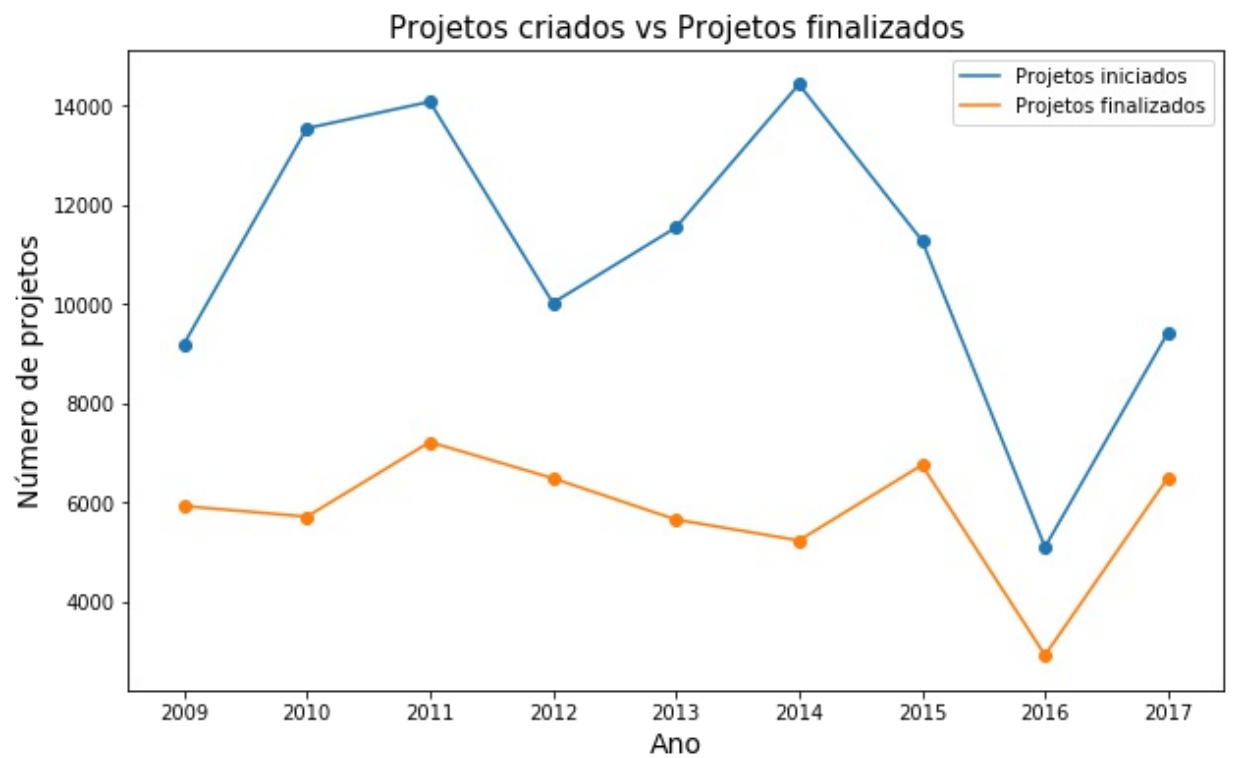
Durante os meses de abril/maio foram realizadas reuniões semanais seja com a equipe gerencial da CGTEC, seja com a equipe gerencial da SEFIC, mas principalmente com a equipe técnica da SEFIC. O objetivo dessas reuniões semanais foi principalmente compreender o processo de execução da Lei Rouanet. Com a equipe técnica, além de compreensão do processo, foram levantados os principais gargalos encontrados pela equipe que acompanha e avalia projetos culturais, principais limitações do SALIC, e principais oportunidades de melhorias sugeridas pelos técnicos/especialistas. Abaixo estão documentados os principais avanços na etapa.

Prestação de Contas

Problema: Tanto a equipe técnica da SEFIC quanto a gerencial reconhece que a etapa de prestação de contas hoje é o maior gargalo na execução da Lei Rouanet para o Ministério. A prestação de contas é dividida em duas grandes fases: 1. comprovação de objeto/física (do projeto em si, verificar se a execução do projeto é consistente com a documentação fornecida, verificar se foi especificado um projeto A e executado um projeto B) e; 2. comprovação financeira (se o capital gasto corresponde aos valores aprovados, item a item). Ambas as etapas são realizadas por funcionários do Ministério da Cultura da SEFIC e avaliação da prestação de conta de projetos é realizada de forma manual usando o SALIC somente para acesso às informações do projeto.

A análise de objeto tem por objetivo verificar se o projeto executado seguiu o planejamento, neste passo não é feita uma análise de custos do projeto, apenas é verificado se o que foi executado segue o objetivo inicial do projeto. Se o projeto for reprovado neste ponto, a

planilha orçamentária nem é analisada e o proponente é obrigado a devolver todo o dinheiro captado. Já análise financeira tem por objetivo verificar se os custos do projeto estão de acordo com a planilha orçamentária vigente para o projeto, se estão de acordo com a IN vigente ou a IN do momento da aprovação do projeto, e se os comprovantes apresentados estão de acordo com os valores e fornecedores declarados. A imagem abaixo evidencia o gargalo na prestação de contas, no qual é apresentado tanto a quantidade de projetos abertos/aprovados, e a quantidade de projetos finalizados.



{width=400px}

Etap	Num de Projetos
Admissibilidade	437
Execução	7486
Prestação de contas	17097

Oportunidades de aprendizagem de máquina: Grande parte da análise é feita baseada na experiencia dos técnicos/especialistas. A grande oportunidade é desenvolver sistemas de recomendação para orientar a tomada de decisão dos técnicos, agilizar o processo de avaliação de projetos, e gerar jurisprudência baseada em avaliações anteriores para respaldar os pareceres do Ministério.

Proposta: Propor uma série de indicadores/métricas, recomendações para orientar a

avaliação de prestação de contas. Os indicadores/métricas validadas com a SEFIC foram:

- Indicadores de qualidade do projeto
 - Complexidade financeira
 - Complexidade temporal
 - Complexidade de serviço
 - Complexidade de fornecedores
- Relação de similaridade entre projetos
- Recomendação de comprovantes com algum tipo de inconsistência

Resultados Preliminares: Alguns experimentos foram realizados, seja para aprender sobre os dados disponíveis no Salic, seja para testar algumas hipóteses e algoritmos. Cada experimento foi registrado na forma de jupyter-notebook, disponível no repositório do projeto <https://github.com/lappis-unb/salic-ml/tree/master/salicml>. No anexo IV é apresentado o template proposto pela equipe para manter a padronização das análises dos dados.

Admissibilidade

Oportunidades de aprendizagem de máquina: A grande oportunidade consiste em recomendar ações ao proponente para que este faça um projeto de melhor qualidade. A recomendação deve ser feita a partir não da IN (Instrução normativa), mas sim de aprendizagem da base de dados dos projetos já aprovados/finalizados. O objetivo é gerar recomendações para o proponente para que este faça um projeto de qualidade (com menor probabilidade de haver diligências), desde a concepção.

Voltando ao processo de admissibilidade, o Salic já utiliza um sistema automatizado para validação de preço dos itens cadastrados em uma proposta: `PROCEDURE dbo.spCalcularMedianaItemOrcamentario`.

Proposta: Sistema de recomendações de projetos durante admissão/execução:

- Sugestões de modificação da planilha orçamentária
- Sugestões de acompanhamento do projeto

Execução

Oportunidades de aprendizagem de máquina: Como na fase de admissibilidade, a grande oportunidade consiste em recomendar do proponente para que este faça um projeto de melhor qualidade.

Proposta: Acompanhar e recomendar a prestação de contas:

- Com base no tempo de execução típico para cada tipo de projeto, recomendar ao proponente o registro antecipado de comprovantes dos itens (valor gasto, nota fiscal etc).

Qualidade do Projeto

A proposta sobre o uso de aprendizagem de máquina para o apoiar a execução da lei Rouanet é baseada na ideia de encontrar padrões que indicam a qualidade de um projeto. Logo, grande parte dos algoritmos de aprendizagem de máquina serão desenvolvidos para inferir/aprender a partir dos projetos já finalizados/executados índices de qualidade do projeto, a partir de diversas perspectivas.

O valor imediato é oferecer um panorama geral de qualidade do projeto que pode ser utilizado para orientar decisões de distribuição das análises e pontos mais críticos de cada projeto.

O valor de médio prazo é, a partir da interação das hipóteses do ML com os técnicos da SEFIC (construção conjunta Lappis+Minc) lapidar um índice de qualidade de projeto que atue preventivamente nas etapas de admissibilidade e acompanhamento, aumentando a qualidade dos projetos que chegam para as etapas de análise, diminuindo o número de diligências, reprovações e tomadas de conta.

Usar o aprendizado de máquina para detectar, inferir e exibir informações em tempo real que ajudem na qualificação e otimização do trabalho de análise de objeto e financeira dos projetos.

Essa ação também terá efeitos de médio e longo prazo no aumento da qualidade dos projetos propostos e na antecipação de eventuais erros durante a fase de execução, diminuindo o número de projetos que necessitam ser diligenciados ou reprovados.

Evolução Chatbot

Após os testes, junto a SEFIC, na entrega anterior, foram analisados os resultados. Estes resultados mostram que houveram muitas falhas de usabilidade, de conteúdo e de tecnologia. As falhas de usabilidade dizem respeito sobre como o usuário reage as interações da bot, a maior parte dos usuários não conseguiram manter uma conversa fluida e não chegaram no objetivo deles. As falhas de conteúdo dizem respeito sobre os textos utilizados pela assistente, eles se mostraram muito longos e técnicos, não contribuindo para a fluidez da conversa. As falhas de tecnologia dizem respeito sobre a tecnologia hubot-natural que se mostrou incapaz de analisar algumas palavras e contextos em português do Brasil, pois ela é voltada para o português de Portugal, impossibilitando alguns fluxos de conversa.

Para atacar estes problemas, foram necessárias algumas ações. O problema de usabilidade foi trabalhado fazendo-se um estudo sobre a usabilidade em chatbots, principalmente na parte do chatbot guiar a conversa para achar a resposta correta para o usuário. Um relatório

Como resultados dessa etapa foi gerado uma primeira baseline do fluxo de conversa com a Rouana, no contexto de admissibilidade, no qual será implementado e validado na próxima etapa. A ultima versão do fluxo é mostrado abaixo. As versões testadas dos fluxos anteriores estão disponíveis no Anexo III.



O problema relativo a tecnologia foi resolvido com o uso de outra tecnologia, o Rasa. Esta tecnologia se mostrou mais capaz em resolver os problemas que estamos enfrentando com a tecnologia anterior. Porém esta tecnologia não se conecta a nossa plataforma de chat, RocketChat, a qual não foi trocada. Para resolver isso, tivemos que desenvolver um conector entre o Rasa e o RocketChat. Este connector será incluído, como contribuição, para a

comunidade do Rasa, possibilitando qualquer pessoa utilizar esta nossa solução assim que for utilizar o Rasa.

Além do conector, foram feitos testes de algumas funcionalidades do Rasa. Diferente da tecnologia antiga, o Rasa possui várias funcionalidades a mais para o auxílio de desenvolvimento de chats. Para saber o que era possível ou não adicionar na nossa assistente virtual, foram feitos testes de algumas funcionalidades. Estes testes foram feitos através da leitura da documentação e da interação direta com a comunidade do Rasa, que se mostrou mais solícita e populosa que a comunidade da tecnologia anterior.

Visualização de dados e criação de Dashboards

Devido à mudanças de escopo do pacote de trabalho "Catálogo de Software", o pacote "Visualização de dados e criação de Dashboards" foi adiantado no cronograma, e a equipe do catálogo foi realocada para essa frente de trabalho.

As seguintes atividades foram desenvolvidas nesse período:

- [x] Execução do método Design Sprint para elaborar propostas de visualização
- [x] Desenvolvimento de protótipos em código usando a prática de desenvolvimento em power trios

Como apresentar projetos de incentivo à cultura de forma que as pessoas queiram contar para alguém?

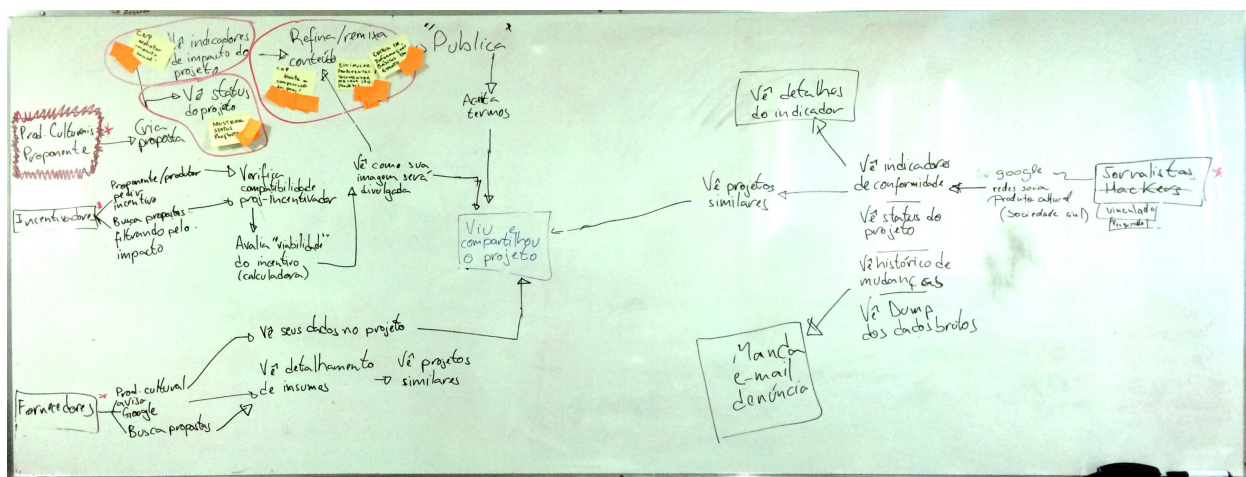
Informações de embasamento

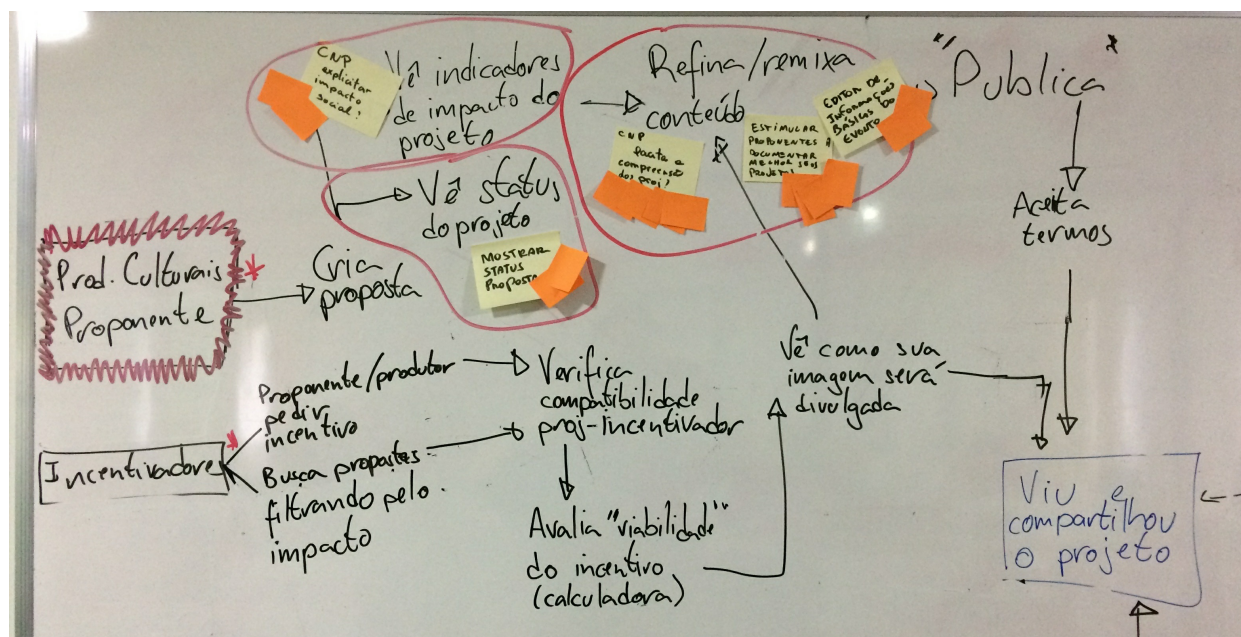
"Dores"

1. Versalica mostra pouco o impacto - valor positivo
2. Estimula exclusivamente caça às bruxas
3. Falta vitrine de divulgação
4. Muitos dados e pouca informação
5. CARETA
6. Falta visão transversal
7. Produtor cultural não é tratado como stakeholder

Perguntas e dúvidas

1. Quem são as pessoas / usuários?
 - R: Os principais são Produtores culturais, Incentivadores, Jornalistas e Fornecedores





Esboços de solução

Ver material nas épicas: [Visualização dos Projetos](#), [Indicadores culturais](#) e [Timeline](#).

Design Sprint panorama

Criar uma visualização de informações relevantes (da cultura) que sempre quis saber e nunca soube como conseguir

Informações de embasamento

"Dores"

1. Visão transversal da Cultura
2. Muitos dados, pouca informação
3. Relacionamento político (áreas do MinC)
4. Saber o que a lei faz e o que não faz (projeto submetido vs aprovado)
5. Saber as áreas da Cultura mais carentes
6. Ausência de mapeamentos por dimensões
7. Como melhorar o processo de incentivo
8. Falta de visibilidade dos projetos
9. Falta de visibilidade à cauda longa da lei -> impacto social, simbólico e econômico

Perguntas e dúvidas

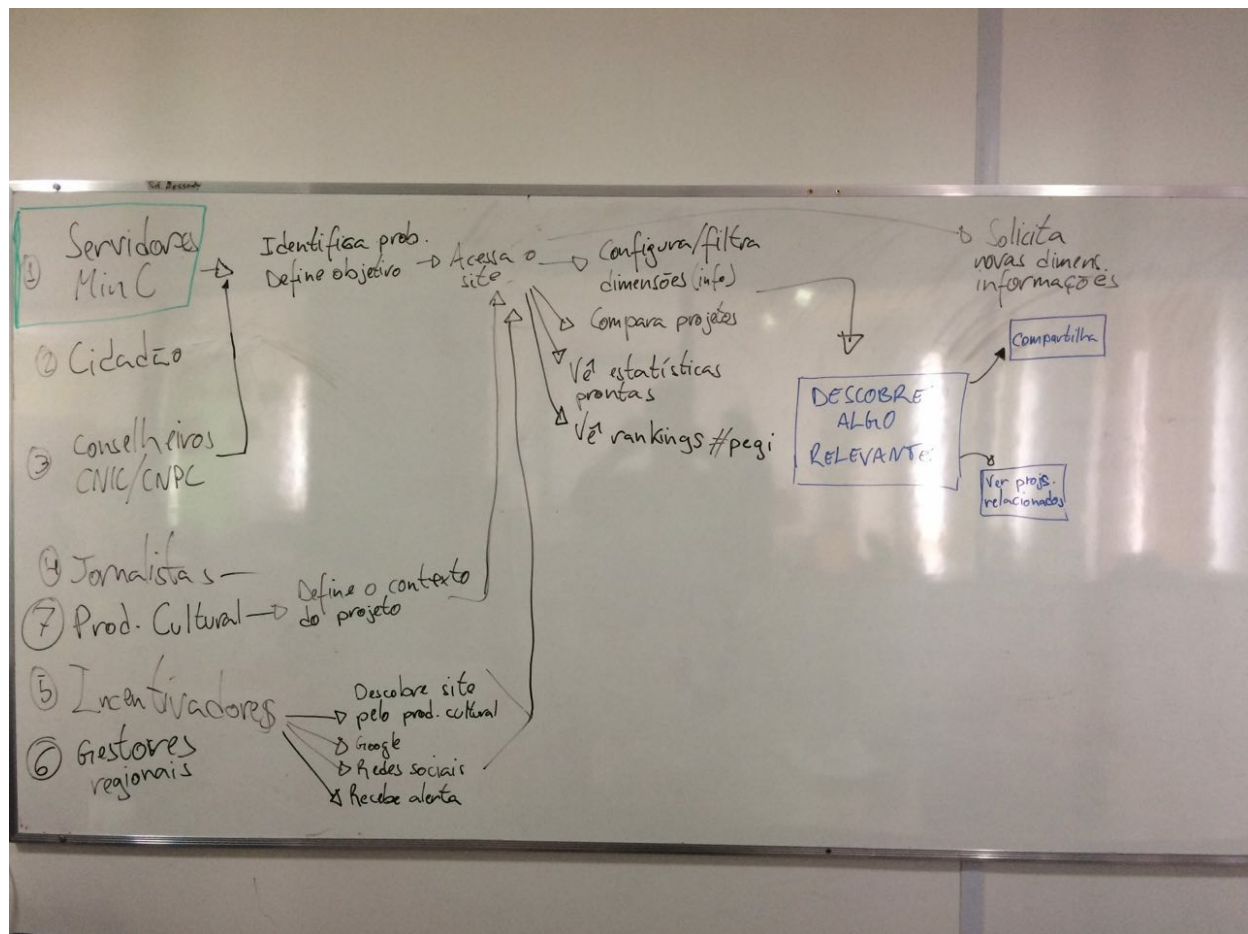
1. Quais dimensões vamos avaliar?

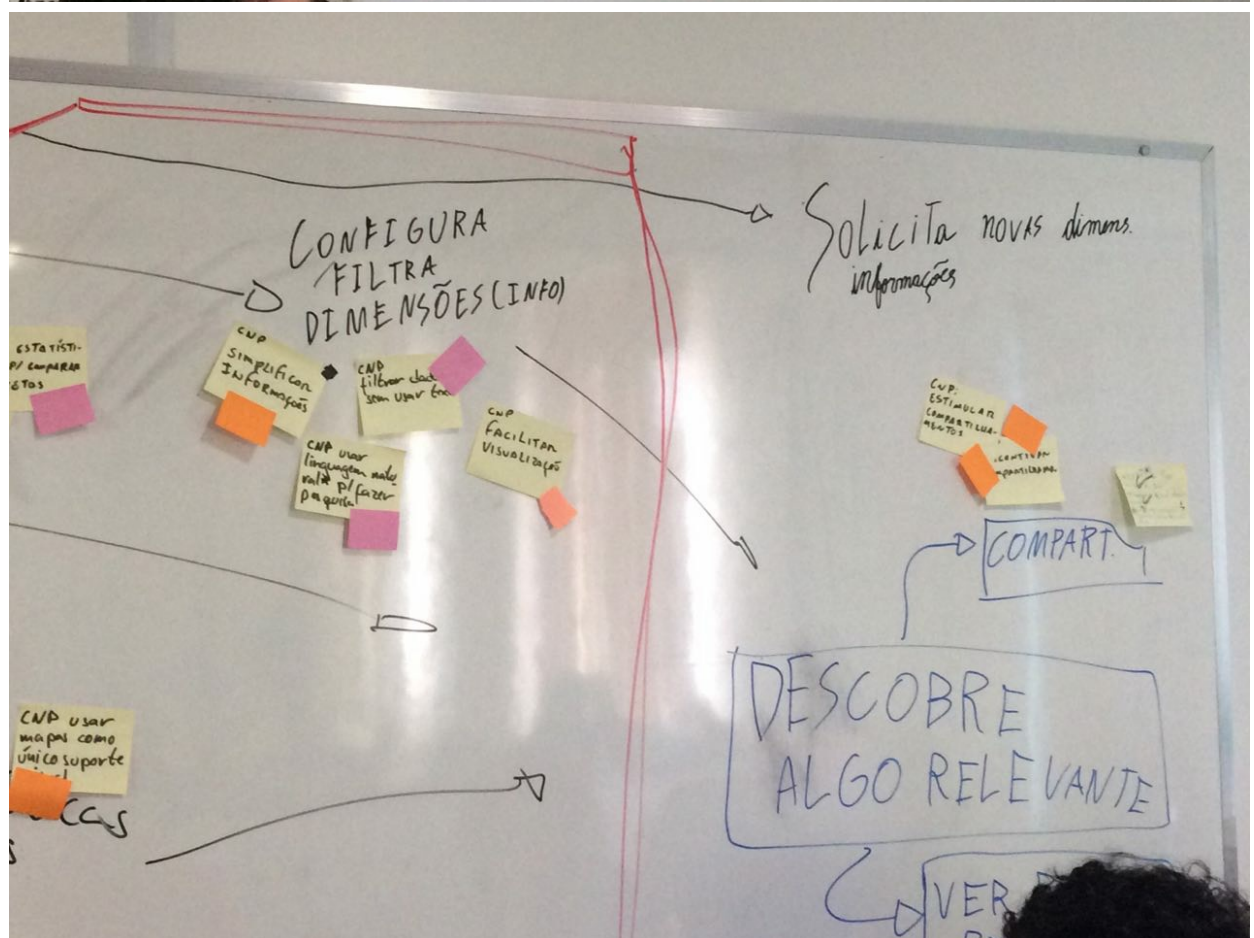
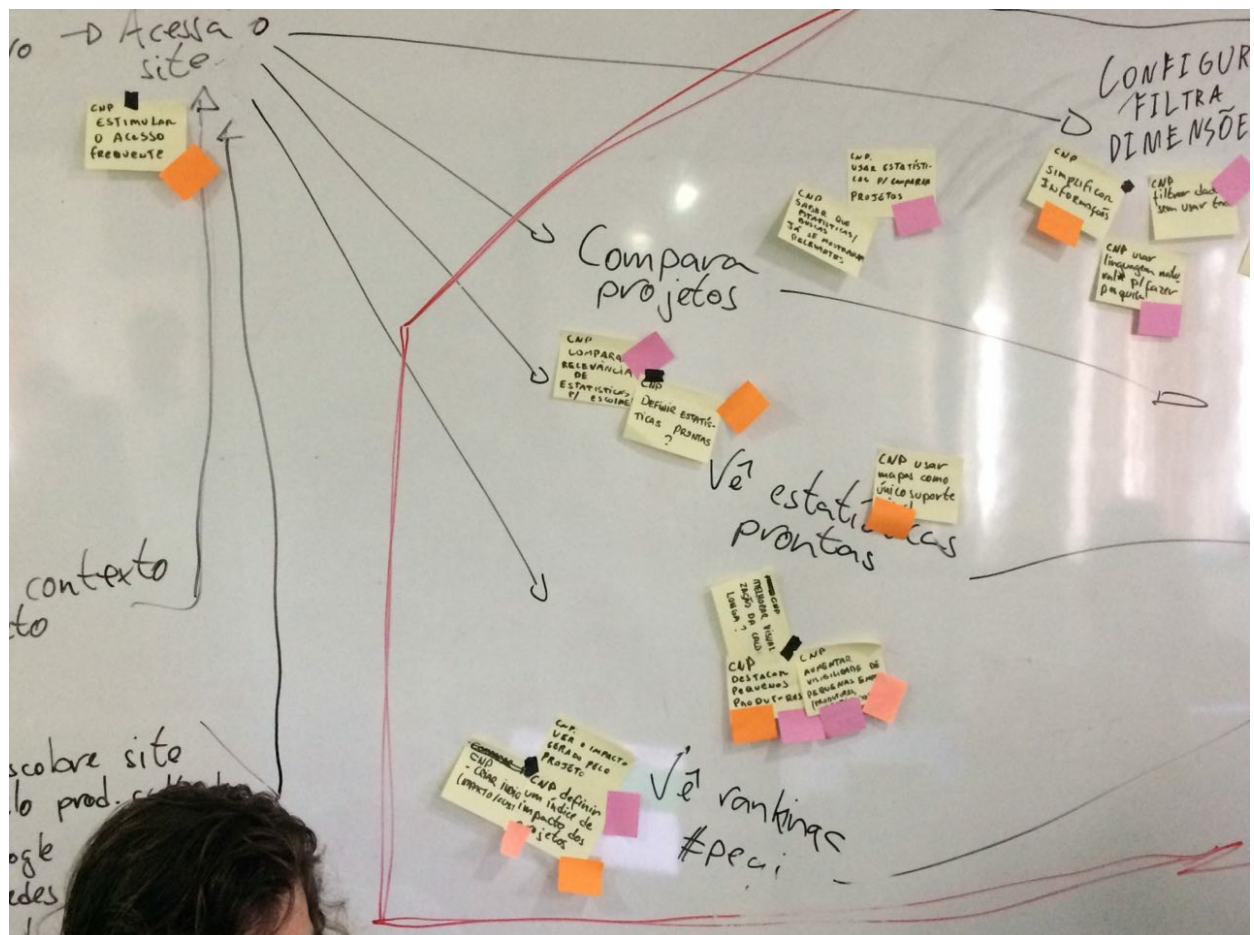
2. Como medir o impacto?
3. Quem são os inimigos do sucesso?
4. Quão versáteis temos que ser? (Somente estatísticas fixas ou possibilidade de variar dimensões)
5. Escopo (muita informação)

Principais atores

- Sefic
- Cidadãos e cidadãos
- Conselheiros CNIC e CNPC
- Jornalistas

Mapa de uso





Esboços de solução

Ver material nas épicas: [Home](#), [Mapa](#) e [Índice de impacto](#).

Prototipagem em Código

As demais sprints da etapa foram dedicadas à implementação das propostas validadas nos designs sprints. Todos os protótipos foram desenvolvidos utilizando o modelo proposto pelo laboratório chamado power trio, que, ao invés do tradicional prática ágil pareamento, usamos um par de engenheiros de softwares desenvolvendo junto com um designer, tomando decisões sobre código/interface/uso de forma colaborativa e contínua.

Todos os protótipos em códigos desenvolvidos estão disponíveis no gitpage do projeto <https://lappis-unb.github.io/PromovaCultura/>

Aferição e aceitação de produtos de software

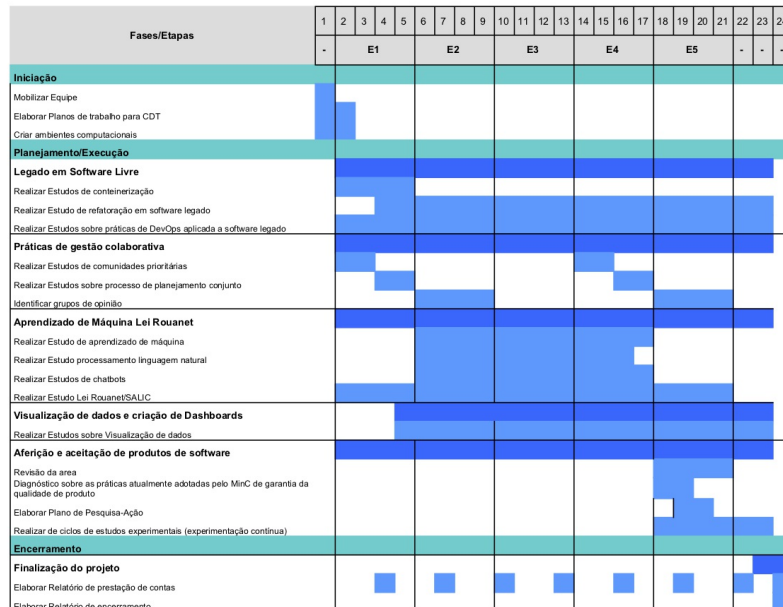
O objetivo geral desta frente de pesquisa é auxiliar os times de desenvolvimento e gestores de TI do MinC a aprimorarem sua capacidade em tomar decisões acerca da qualidade das versões dos produtos de software entregues por seus fornecedores.

As atividades referentes a essa frente foram replanejadas para iniciar a execução dos ciclos de estudos experimentais a partir das entregas contínuas, previstas para serem amadurecidas no final da próxima etapa. A alteração de planejamento está evidenciada na nova proposta de cronograma para o projeto.

Mudanças no cronograma

As seguintes mudanças são propostas no cronograma:

- Cancelamento do pacote de trabalho "Catálogo de Software" - foi cancelado na entrega passada, com comum acordo com os gestores do ministério.
- Adiantar o pacote de trabalho "Visualização de dados" - essa frente prevista pra se iniciar no mês 10 do projeto vai ser iniciado no mês 7 do projeto. A equipe do catálogo vai ser em sua maioria realocada para esse pacote de trabalho.
- Adiar as atividades relacionadas ao pacote de trabalho "Aferição e aceitação de produtos de software" - previsto desde o início do projeto, algumas análises qualitativas já foram realizadas/entregues, porém, as demais entregas foram adiadas para iniciar no mês 18 do projeto.



Acompanhamento Financeiro

O valor do repasse referente à Etapa III foi de R\$202.600,00. Todo esse repasse foi na rubrica 30.90.20, referente à auxílio Financeiro a Pesquisa (Bolsas). Desse repasse, um total de R\$190.635,90 foi executado na Etapa II, representando na prática que o orçamento foi consumido apenas na categoria mão-de-obra. Todo esse valor foi executado no pagamento das bolsas do time, e o valor gasto por frente do projeto pode ser visto na Figura 2 abaixo.

!Neste gráfico é possível observar a representação do percentual do custo da mão-de-obra incidido em cada equipe do projeto. A maior alocação de recursos encontram-se nas equipes do Catálogo de Softwares Culturais(representado pela cor azul), uma vez que grande parte das funcionalidades desenvolvidas são providas através desta frente, e a equipe do Aprendizado de máquina(representado pela cor verde), que desenvolveu o chatbot.] (figs/Planilha_de_Custos_MinC_2017-2019 - Entrega_3.png){width=400px}

Assinatura

Responsável pela Execução:

Nome: Carla Silva Rocha Aguiar
(Coordenadora do Projeto)

Assinatura:



Data: 06/04/2018

Anexo I - Resultado da homologação com a Rouana

Processo de Homologação da Rouana

Disponível em : <https://github.com/lappis-unb/EcosystemasSWLivre/wiki/Rouana-Homologação>

O que é a Rouana?

A **Rouana** é a assistente virtual de incentivo à Cultura em desenvolvimento pelo MinC, em parceria com o LAPPIS (UnB - FGA), para interagir com os cidadãos (produtores culturais e outros interessados) de forma simplificada com objetivo de apoiá-los no entendimento da Lei Rouanet e no uso da plataforma SALIC.

Versão 0.1

Esta **versão 0.1 (beta)** se utiliza de técnicas simples de Aprendizagem de Máquina alimentada (treinada) com as perguntas e respostas fornecidas pela ouvidoria da SEFIC (Secretaria de Fomento e Incentivo a Cultura).

Testes para homologação

A **assistente virtual Rouana** está prestes a ser disponibilizada em produção no portal da Lei de Incentivo à Cultura (Lei Rouanet) e para cumprir este objetivo sua ajuda é necessária.

Assim, com objetivo de homologar a **assistente virtual Rouana** versão 0.1 (beta) inicie um diálogo com o assistente virtual através do endereço abaixo e tente compreender e aprender os objetivos de aprendizagem que seguem.

- <https://rouana.lappis.rocks/>

É importante que durante a interação com a **Rouana** você assuma o papel de um cidadão sem conhecimento prévio sobre a Lei Rouanet ou sobre o sistema SALIC e seus procedimentos internos. Nesta fase de homologação a **assistente virtual Rouana** irá solicitar sua identificação (**Nome** e **Email**), preencha seus dados corretamente para podermos analisar com qualidade os dados dos testes de homologação, esta identificação não será solicitada ao cidadão quando a **Rouana** entrar em produção.

Ao final dos testes solicitamos que você responda a pesquisa no link abaixo se identificando com o mesmo **Nome** e **Email** informados anteriormente, os resultados desta pesquisa irão guiar o desenvolvimento das próximas versões do assistente virtual.

- <https://rouana.lappis.rocks/pesquisa>

Objetivos de Aprendizagem

Selecione 02 objetivos de aprendizagem de cada um dos 04 grupos abaixo e dialogue com o assistente virtual com o objetivo de aprender tais objetivos.

Objetivos de Aprendizagem - Grupo 01

1. Aprenda o que é a Lei Rouanet.
2. Descubra quantos projetos já participaram da Lei Rouanet.
3. Entenda o papel da SEFIC e MinC na Lei Rouanet.

Objetivos de Aprendizagem - Grupo 02

1. Imagine que você fez sua proposta de projeto, e quer saber quando você pode começar a captar os recursos para ele, descubra como e quando você estará autorizado para fazer isso.
2. Seu projeto está na situação D52, descubra o que isso significa.
3. Seu projeto já foi enquadrado, aprenda como desistir do prazo recursal.

4. Descubra como recuperar sua senha no SALIC.
5. Descubra como fazer o cadastramento de proponentes no SALIC.

Objetivos de Aprendizagem - Grupo 03

1. Descubra como você pode vincular o seu CPF ao CNPJ.
2. Aprenda como fazer sua proposta de projeto.
3. O valor do ingresso do seu projeto é maior que 250 reais, descubra como é possível cadastrar um valor maior.
4. Você passou a manhã de hoje preenchendo a proposta no SALIC, porém ao retornar ao sistema você notou que ela não tinha sido salva, descobrir como resolver isso.

Objetivos de Aprendizagem - Grupo 04

1. Descubra a diferença entre contrato de patrocínio e carta de intenção.
2. Descubra em qual banco você pode abrir a conta do seu projeto.
3. Aprenda se você pode utilizar o dinheiro captado em aplicações financeiras.
4. Seu projeto já começou a captar recursos, descubra quando você pode movimentar o recurso captado.
5. Você conseguiu captar mais do que o requisitado para o seu projeto, descubra o que fazer nesta situação.

Análise dos resultados

Análise dos [chats](#) por usuário.

@guest-25

Análise

Fez várias perguntas repetidas com a mesma estrutura, sem obter a resposta. Muitas perguntas eram muito parecidas, ou iguais.

Perguntas para análise

Perguntas Feitas:

- como fazer o cadastramento de proponentes no SALIC?
- como posso vincular o meu CPF ao CNPJ?

O contexto foi encontrado apenas na segunda pergunta, não sei se são similares.

Perguntas Feitas:

- É possível cadastrar um valor maior que 250 reais?
- O valor do ingresso do meu projeto é maior que 250 reais, é possível cadastrar um valor maior?

Não achou a pergunta certa, a pergunta certa cadastrada é:

O sistema informa **que** o custo por beneficiário não pode ultrapassar o valor **de** 250,00, mas não irei comercializar ingressos **ou** o ingresso não ultrapassa o valor citado. Como resolver?

Perguntas Feitas:

- quando posso movimentar o recurso captado?
- quando posso movimentar o recurso captado?

Estava querendo se referir a uma pergunta da fase de execução, porém estava no contexto geral da lei rouanet. A pergunta que a Rouana achou mais similar naquele contexto foi `quanto dinheiro a lei movimenta?`, e o que ele queria era `A partir de quando a conta pode ser movimentada?` no contexto de execução. A princípio não houve a menção de troca de contexto.

@guest-26

Análise

Várias perguntas sem resposta pela falta do devido contexto.

Perguntas para análise

Perguntas Feitas:

- o que é a situação d52

Pergunta foi feita sem contexto, a Rouana confundiu com `O que é uma assistente virtual?`. Temos que ver como a biblioteca de NLP trata esses casos.

Perguntas Feitas:

- como recuperar a senha do salic
- perdi minha senha do salic

Rouana não achou o contexto de admissibilidade, onde a pergunta sobre a senha está. Para mim esta pergunta deveria estar em uma categoria própria do salic.

Interações Interessantes

@guest-26 9:54 AM
quantos projetos já participaram da Lei Rouanet

rouana 9:54 AM
Que bom! A lei Rouanet é meu assunto **preferido** :)
O que você gostaria de saber sobre a lei Rouanet?

@guest-26 9:54 AM
quantos projetos já participaram da Lei Rouanet

O usuário interpretou, creio eu, que foi um erro do sistema, e repetiu a pergunta após a Rouana ter detectado um contexto. Então ele obteve a resposta correta.

@guest-26 9:55 AM
situação d52

rouana 9:55 AM
Acho que ainda não estou treinada para responder esse tipo de **assunto** :(
tente outro assunto
Além da Lei Rouanet, eu sei falar **sobre**:
- Preenchimento de proposta no SALIC
- A fase admissibilidade do seu projeto
- A fase execução do seu projeto

@guest-26 9:55 AM
admissibilidade

rouana 9:55 AM
Então você está na fase de admissibilidade
Qual sua dúvida sobre ela?

@guest-26 9:55 AM
d52

rouana 9:55 AM
Um projeto homologado (estado D52) recebeu uma posição definitiva

Este usuário entendeu como o fluxo funciona após 14 iterações, creio que este sucesso foi devido ao "menu" que a Rouana incluiu na mensagem de erro.

@guest-26 9:56 AM

execução do projeto

rouana 9:56 AM

Desculpa, sou uma assistente virtual em fase experimental
às vezes não entendo certas mensagens
use outras palavras na sua mensagem

@guest-26 9:56 AM

admissibilidade

rouana 9:56 AM

Nossa, desculpa, não sei mesmo como responder isso...
mas a equipe da Ouvidoria do MinC pode te ajudar, você pode entrar em contato por
Telefone: +55 (61) 2024-2245
Site: <http://ouvidoria.cultura.gov.br/ouvidoria/login.jsp>
ouvidoria.cultura.gov.br
Sistema de Ouvidoria

@guest-26 9:56 AM

admissibilidade

rouana 9:56 AM

Acho não entendi... Me passa mais detalhes por favor

Duas coisas interessantes, o usuário tentou trocar para o contexto de execução com a frase `execução do projeto`. Aparentemente, a Rouana não está identificando a palavra `execução`, creio que seja por causa do stemmer que é limitado a português de Portugal. A outra coisa interessante é que o usuário já estava no contexto de admissibilidade, e tentou ir para admissibilidade de novo, por algum motivo. Creio que uma resposta, vinda da Rouana, como `Já estamos falando sobre ela =)` causaria menos confusão no fluxo geral.

@guest-26 9:57 AM

perdi minha senha **do** salic

rouana 9:57 AM

Ah isso acontece com todo mundo ;)
você pode gerar uma nova senha aqui: <http://salic.cultura.gov.br/autenticacao/index>
salic.cultura.gov.br
Sistema **de** Apoio **À** s Leis **de** Incentivo **À** Cultura
ou se isso não funcionar, mande uma solicitação para: senhasalic@cultura.gov.br

@guest-26 11:53 AM

olá, bom dia

rouana 11:53 AM

Acho não entendi... Me passa mais detalhes por favor

O usuário tentou retomar a conversa com a Rouana horas depois, e ela lembrou do contexto anterior, não respondendo o bom dia dele.

@guest-27

Análise

Após algumas interações falhas, obtive apenas uma resposta, e ainda foi a errada. Desistiu de continuar conversando.

@guest-32

Análise

Fez muitas perguntas novas, e muitas caíram em 0 que é modal? , por causa do 0 que é .

Perguntas para análise

Perguntas Feitas:

- Como alterar o proponente?

A Rouana entendeu que ele estava se referindo a pergunta Perdi minhas alterações na proposta

Perguntas Feitas:

- Qual é o limite de arquivo para anexar no SALIC?

A Rouana entendeu Qual é o novo fluxo do Salic? , creio que o motivo foi o "Qual é" e o "SALIC".

Perguntas Feitas:

- Qual é o telefone de atendimento da Admissibilidade

A Rouana não soube responder, seria uma boa retornar exatamente o telefone da ouvidoria nesse caso.

Perguntas Feitas:

- Quando será a próxima CNIC?

- Quando será a próxima reunião da CNIC?
- O que é CNIC?
- O que significa CNIC?
- Como definir cultura?
- Qual é o percentual de direito autoral?
- Qual é o percentual de remuneração para captação de recursos?
- Qual é a porcentagem permitida para os custos de divulgação?
- Qual é o teto para os custos de divulgação?
- Qual é o prazo para homologar o recurso?
- Como tratar depósito equivocado?
- Como elaborar uma proposta de dança?
- Prefeitura pode ser proponente?
- Como arquivar uma proposta?
- O que é o ver salic?
- Qual é o prazo para a análise de ad referendum
- Quais são as mudanças da nova Instrução Normativa?
- Qual é o CEP da SEFIC?
- Como calcular o ecad?
- Como registrar a autoria de uma obra?
- Qual é o telefone do gabinete da SEFIC?
- O que é IBRAM?
- Para onde vai o dinheiro de projeto não executado?

Perguntas não cadastradas na nossa base

Perguntas Feitas:

- Que horas são?

Não é o foco, mas seria interessante responder, só para dar um ar de inteligencia a mais.

Frases erradas na Rouana

- Acho não entendi... Me passa mais detalhes por favor

Avaliação da Homologação da Rouana

Documento disponível em: <https://github.com/lappis-unb/EcosystemasSWLivre/issues/60>

Critério de aceitação de homologação de chatbot: 30% dos usuários satisfeitos, ou seja, conseguiram resolver seus problemas.

Pergunta: Qual foi a taxa de acerto das respostas dadas pela Rouana?

Creio que as respostas tão numa escala muito pequena, para o usuário calcular se o bot acertou de 70 à 89 ou de 90 a 100 por exemplo, são intervalos pequenos de apenas 10%. Acho pouco intuitivo e talvez os usuários não saibam responder isso. Creio que melhor seria perguntar algo numa escala mais subjetiva e nós calcularmos o percentual depois olhando os logs, nessa escala subjetiva imagino algo como: o bot acertou pelo menos metade das minhas perguntas ou o bot errou a maior parte das minhas perguntas.

No mais achei que está bom em quantidade de questões e na descrição delas, vamos fazer uma rodada interna após bater o martelo sobre essa primeira versão do roteiro e do survey? Acho que uma primeira rodada entre nós e o resto da equipe envolvida no chatbot nos dará feedback para pequenos ajustes antes de enviar pro MinC.

Principais problemas apontados nas respostas:

- textos longos, fonte pequena
- dificuldade em entender e chavear entre contextos, usuários não compreendem bem o chaveamento entre contextos
- não consegue responder muitas das dúvidas dos usuários (causa possível: base de conhecimento pequena)
- paleta de cores não segue a identidade do SALIC/Rouanet
- mais de 50% dos participantes ficaram insatisfeitos ao avaliar a conversa geral com o bot
- repetição da apresentação do bot "oi eu sou a assistente virtual ..."
- repetição de despedida, o bot entende que o usuário está terminando a conversa quando está apenas agradecendo pela informação
- apenas 1 pessoa obteve a resposta certa com 1 pergunta

Anexo II - Estudo de Usabilidade de Chatbots

Início da Ideia

Ao iniciar o desenvolvimento de um bot, assim como de qualquer outra ideia, é necessário fazer questionamentos sobre sua ideia, de modo a criar algo que cumpra com seu objetivo e atenda da melhor forma o usuário final.

Abaixo encontram-se 5 questões que devem ser respondidas antes de desenvolver um bot:

1. Que resultados eu quero para o meu produto ou usuários?

Como citado no [\[2\]](#)

"Não só construa uma cadeira, construa uma cadeira que as pessoas possam sentar."

Devemos sempre saber o porquê, o que aquilo que geramos terá um propósito.

2. Seu chatbot alcança um pedaço novo do mercado?

Do mesmo jeito que outros produtos, é importante certificar-se de que existem outros produtos que resolvem o mesmo problema, e se sim, no que o seu produto diferencia.

3. O Chatbot é a solução para o seu objetivo?

Muitas das ideias podem ter suas soluções contempladas por um chatbot, mas às vezes métodos tradicionais são saídas mais rápidas e que atendem um grupo maior de pessoas. Exemplos disso são bots para dizer sobre o clima, este se torna mais trabalhoso de se acionar quando se quer a resposta do que simplesmente pesquisar na internet.

4. Sua ideia tenta substituir tarefas que as pessoas gostam de fazer?

Se sim, é fundamental ter em mente que uma hora ou outra terá um possível usuário que irá criticar e ficará receoso em usar o chatbot, então para isso deve-se contruir estratégias para convencê-lo.

5. Os custos do bot valerá a pena?

Pensando no bot como produto de sua empresa, é recomendado levar em consideração todo esforço dispendido para o desenvolvimento do bot e se isto valerá a pena, não somente no retorno monetário, mas também de conhecimento agregado.

Categorias de AI

Não existe a necessidade de categorizar o bot em desenvolvimento, mas fazê-lo ajuda na compreensão da abrangência do bot, tanto quanto o que e quanto este bot pode expandir.

As categorias de um bot segundo o artigo [\[3\]](#) são:

Assistants

Exemplos de *Assistants* são Alexa, Siri e Cortana. Estes possuem capacidade atender as tarefas complexas exigidas pelo usuário, por comando de voz ou por chatbot, e sua capacidade de responder a estas tarefas aumenta a medida que ele se torna mais treinado.

Guide

Nesta categoria, o bot pergunta algumas tarefas, guiando o usuário a realiza-las. Ele mostra aos usuários qual deve ser os próximos passos, e caso ocorra algum erro, sugerir outros caminhos para o sucesso.

Resumindo, os *Guides* são dedicados a levar os usuários para seu destino final.

Um exemplo de *Guide* é o Wase.

Consultant

Um pouco além do *Guide*, o *Consultant* abrange o *user case* onde os usuários precisam de uma solução ou argumento para sua problemática de maneira "*just in time*", e o bot age como um ser humano mostrando o informações e recomendações. A **Rouana** se encaixa nesta área.

Colleague

A *Colleague* é como o *Consultant*, mas é movida a análises de dados e de compreensão da atual situação, e seu domínio de conhecimento é a organização. Este deve acessar documentos relevantes como planilhas de custos, agendas, planejamento, tudo isso de modo a ajudar gerentes e trabalhadores a tornar o trabalho mais eficiente.

Boss

O *Boss*, como próprio nome diz, delega e manda o ser humano realizar as tarefas, isso ainda no ambiente restrito da organização. Exemplos de ações que o *Boss* pode mandar fazer são: Começar ou terminar um trabalho, mudar cronograma, mandar memorandos ao time, etc.

Personalidade do Bot

Atribuir uma personalidade ao chatbot em desenvolvimento cria um melhor percepção do objetivo do bot e também como ele irá se comunicar, o tom, humor, estilo de conversa. Claro

que assumir que um bot tem vida é um erro, mas saber que uma pessoa irá atribuir uma personalidade a ele querendo ou não, ajuda a construir algo que se relacionará com o usuário de maneira mais natural.[\[4\]](#)

Um bom ponto de partida para gerar uma personalidade é espelhar a do usuário e desenrolar a conversa como se este estivesse falando com alguém próximo, como um amigo, causando um maior engajamento do usuário.

Um exemplo, citado pelo artigo, desta estratégia aplicada são lojas de roupas de jovens, que tendem a contratar outros jovens que se vestem e falam de maneira similar ao consumidor alvo, acarretando na empatia deste.

Quem cria as personalidades

As pessoas aptas a criar personalidade são aquelas que entendem a importância de *personas*.

Escritores e *designers* normalmente são criativos e conseguem transmitir a mensagem, normalmente curta, de maneira agradável.

Construindo a personalidade

Escolha o público-alvo

É sempre importante definir o público-alvo, e mesmo possuindo alguém na equipe capaz de desenvolver a personalidade do bot, é necessário procurar *feedback* de pessoas quem seriam possíveis usuários para saber este bot fala da mesma forma que ele.

Um exemplo do artigo [\[4\]](#) é que um gerente de *marketing* de 47 anos não seria de todo o mais adequado a avaliar se um bot voltado para adolescente está apropriado.

Qual é a tarefa do bot

Novamente, é necessário destacar que o objetivo do bot é algo que deve ser claro. Saber se ele será genérico ou especialista é relevante de maneira a deixar claro o quão profundo será o seu vocabulário para o seu domínio.

Tipos de Personalidade

O artigo descreve que associar as *personas* do bot com o tipo de personalidade Myers-Briggs[\[5\]](#) funciona bem para determinar os pontos fortes e as fraquezas, assim como seus

interesses, desagrados e habilidades. Consequentemente, definir como o bot reagirá em certas situações, o tom de voz, se ele terá ações mais introvertidas ou extrovertidas se torna mais fácil.

Diversidade e Gênero

A discussão sobre gênero e representatividade também atinge a área de chatbots. De acordo com uma pesquisa feita em 2016 por Maxus Survey[6], apesar de 56% dos bots que possuem gênero serem mulheres, 100% dos bot analisados na área de Direito e Finanças são homens, enquanto aqueles bots que são *virtual assistants* são comumente mulheres.

Estes que bots que são identificados como mulheres possuem uma personalidade atraente, agem de maneira sedutora e mais aberta, enquanto bots que agem em áreas mais "sérias" são predominantemente masculinos.

Uma tendência para alterar essas estatísticas, seria desenvolver o bot sem gênero, cuidando do tom de voz e sotaques, além de suas atitudes. Então, mesmo que o bot tenha voz feminina, suas atitudes não devem remeter aos estereótipos.

Definindo a Persona

No livro *Designing Bots*[7] de Amir Shevat, existe um *template* para começar a rascunhar a *persona* do bot. Este possui os seguintes tópicos:

- Nome: Nome para o bot.
- Ambiente: Em que ambiente o bot será implementado. (Ex: trabalho, lar, ...).
- Audiência/ Público Alvo: Deve-se definir também o intervalo de idade do público alvo. (Ex: adultos, entre 25-40 anos).
- Tarefa a ser realizada: O que o bot irá fazer. Isso afeta a construção da personalidade, pois não seria adequado utilizar a mesma personalidade de um bot para vendas de guitarras para um destinado a venda de medicamentos.
- Variações em *runtime*: Ações que deverão ser feitas caso algo saia do planejado. (Ex: o bot irá reportar casos de xingamento e ofensas; transferência para assistência 24 horas).
- Ambiente social/ Culturas aceitas: Deve-se levar em consideração a cultura vigente onde o bot irá atuar. Em ambientes de trabalho, caso o bot chame o usuário de "Querido/a", isto pode não ser visto com bons olhos.

- Marca registrada: As principais características que devem ser associadas ao bot. Aquilo que deve ser o destaque.
- Valores: Nada mais é que os valores associados ao bot, tais como: entreter as pessoas, facilitar sua comunicação...
- Personalidade derivada: O que viria a ser algo a mais ao bot, como ele ser divertido, informal, carinhoso, ou coisas assim que tornariam ele uma pouco mais atraente para o público alvo.

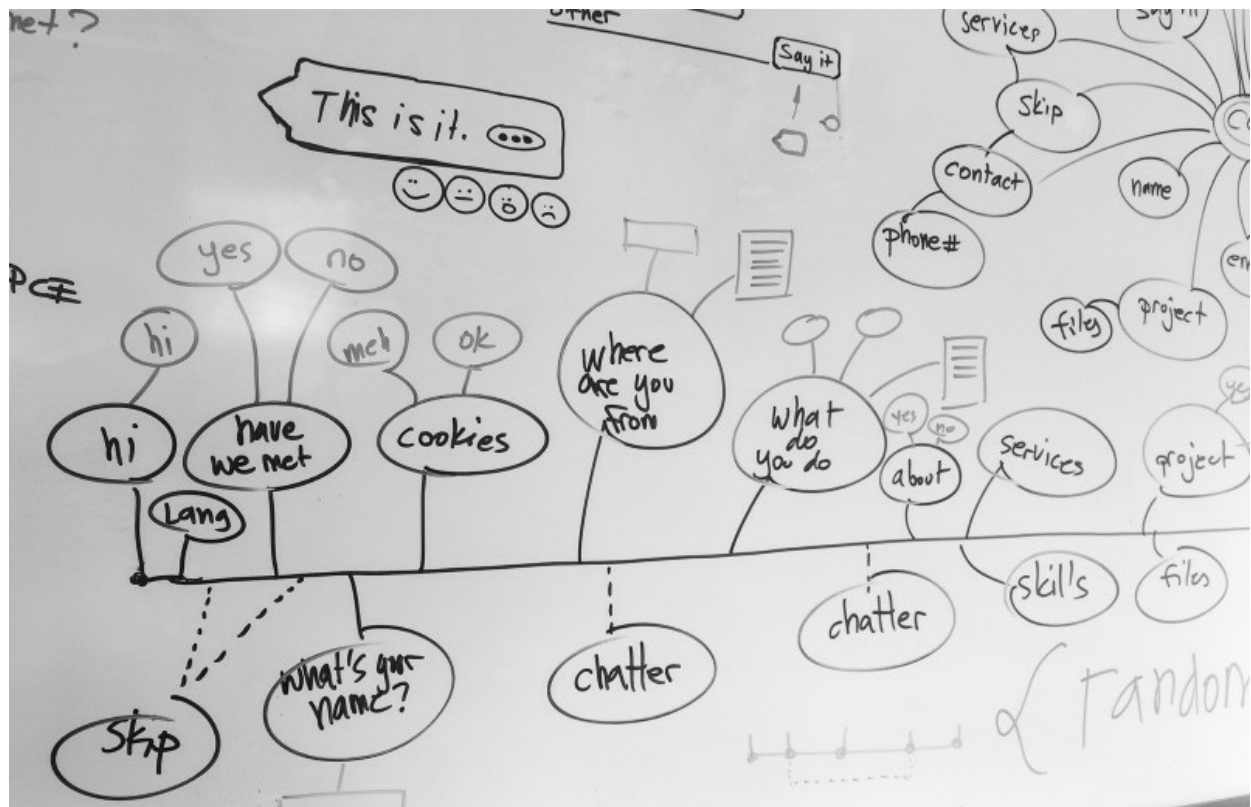
Fluxo de Conversa

A conversa é o cerne da interação entre o bot e o usuário. Para construí-la, existem várias maneiras, e fica a critério do desenvolvedor.

Neste tópico, é descrito algumas práticas que não é consenso, mas já foram experimentadas por outros desenvolvedores.

Desenhando o Fluxo

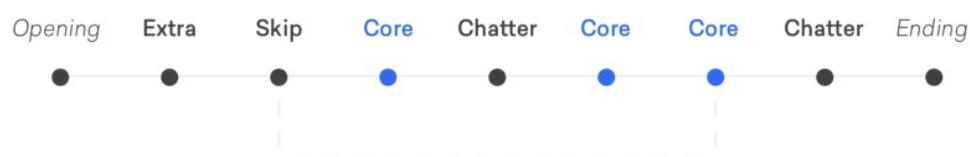
No estudo de caso de [\[8\]](#), eles primeiramente desenharam em um quadro branco um mapa mental relacionando os objetivos do bot.



Estágio inicial do fluxo

Feito isso, eles dividiram as partes funcionais e que possuíam algum padrão ou que era muito atrelado ao objetivo, aqueles que eram para caracterizar um diálogo normal, e os que possuíam alguma informação extra. Ao final, esta divisão ficou da seguinte maneira:

- Opening
- Extra(s)
- Skip(s)
- Core(s)
- Chatter(s)
- Ending

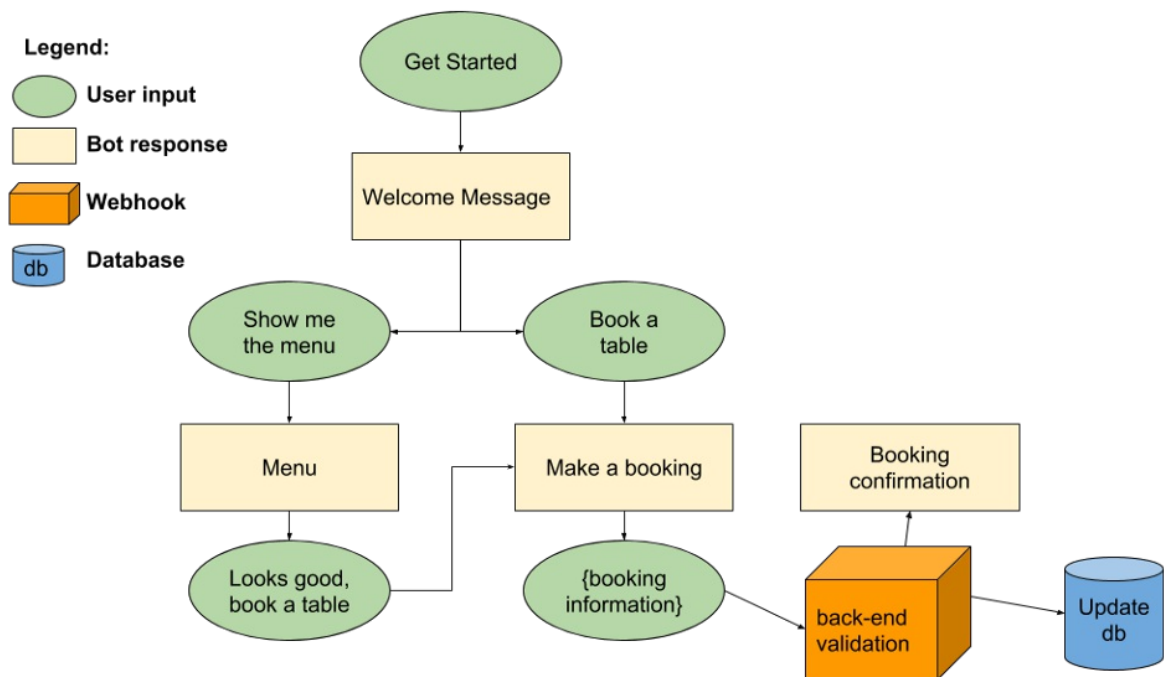


E a partir destes blocos, criaram-se os scripts de conversação.

Agora, os artigos [9] e [10] falam que diagrama é uma forma de se mostrar como se imagina a interação. Segundo o artigo [9], uma conversa não deve possuir mais de 3 interações, e modelá-los traz uma visão melhor para a escrita do script.

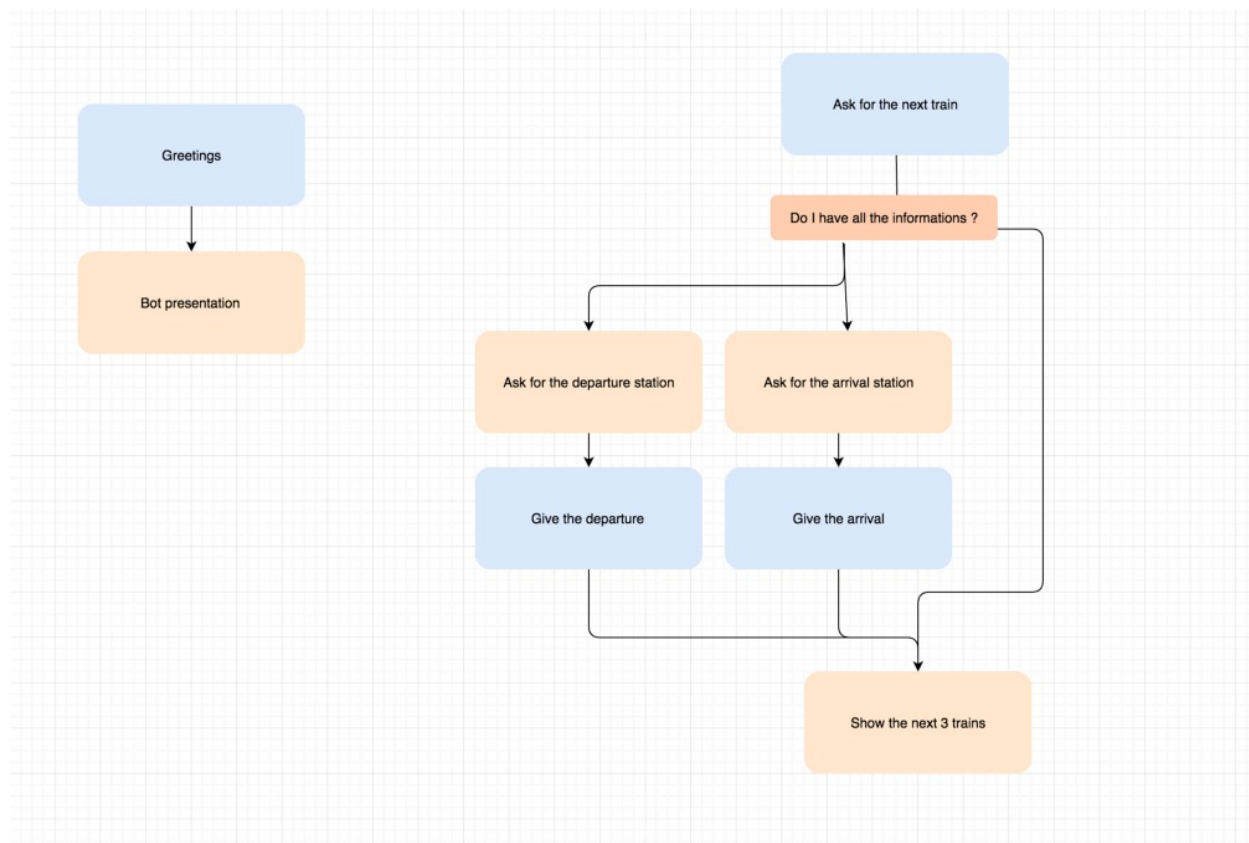
Os diagramas não possuem padrão para escreve-los, mas um coisa em comum é a diferenciação entre o usuário e o bot.

Neste diagrama, ele utiliza a diferenciação tanto por cor, quanto por formato, e mostra quando existe validação e armazenamento de dados.



Fluxo do RestaurantBot

Já nesse caso, não foi definido as validações e armazenamento, mas continuou-se com as ramificações para cada ação que deve ser tomada.



Fluxo Simples

Scripts

Algumas práticas adotadas para a construção do scripts são:

O bot deve iniciar a conversa

Sempre que houver uma nova conversa, o bot deve iniciá-la, deixando claro suas capacidades e seus objetivos. Não se deve esperar que o usuário inicie a conversa. Use a primeira mensagem também para sugerir o que o usuário deve fazer primeiro. [\[12\]](#)

As mensagens de boas vindas/ introdução não devem ter mais de 160 caracteres. [\[11\]](#)

Parágrafos ou Frases?

As pessoas não costumam conversar em parágrafos, ao invés utiliza-se frases simples. Normalmente os usuários não ficam entretidos com o bot quando se deparam com mensagens grandes. Assim, é preferível separar a fala do bot em diversas sentenças. [\[8\]](#)

Algo importante a se considerar é o intervalo de tempo entre as mensagens. Este deve ser proporcional ao tempo de leitura da mensagem anterior, para evitar que as mensagens cheguem ao mesmo tempo, ou que demore muito para aparecer. [\[14\]](#)

Direcione as respostas

Direcionar as respostas é uma forma de evitar que o usuário escreva algo que o bot não suporta. Ao invés de perguntar algo como "Sobre o que você gostaria de falar?", pergunte "Você gostaria de falar sobre o clima de hoje ou de amanhã?", sempre algo que leve para algum dos fluxos pré-estabelecidos. [\[8\]](#)

Porém, deve-se ter cuidado de não construir frases que sejam ambíguas.

Continue e Controle a Interação

Após fornecer o conteúdo que o usuário deseja, forneça o próximo passo, aquilo que pode estar relacionado e que possivelmente seria um incremento para o usuário. [\[12\]](#)

Feedbacks e Erros

Coletar feedbacks tanto positivos quanto negativos é uma parte importante para a evolução bot, então deve-se criar um script dedicado a esta parte.

Quanto aos erros, em [\[7\]](#), existem 4 categorias de erros e como lidar:

- *Chitchat*: são entradas aceitas, que não são relevantes para a conversa, mas são boas para reconhecer.
- *Entity extration issues*: entradas inválidas que aconteceram enquanto era analisadas outras informações fornecidas pelo usuário.
- Abuso: entradas como ofensas, palavrões e discursoss de ódio.
- Genéricos: mensagens de erros que devem ser mostradas que não se encaixam em outra categoria.

Referências Bibliográficas

1. [Storytelling and Bot Making](#)
2. [To Build a successful Chatbot Ask These 5 Questions](#)
3. [AI won't change companies without great UX](#)
4. [A guide to developing bot personalities](#)
5. [Myers-Briggs - 16 Personalities](#)
6. [Siri, Cortana, Alexa, Marcus. Do bots really need a gender?](#)
7. [Designing Bots - Creating Conversational Experiences, Amir Shevat](#)
8. [conversational UI Principles Complete process of Designing a Website Chatbot](#)
9. [Building Great Bots an Enterprise Chatbot Methodology](#)
10. [Dialogflow Restaurant bot Tutorial](#)
11. [Conversational UX Design. All Facebook Messenger Bots Interactions](#)
12. [The Bot Playbook](#)
13. [Designing chatbots - A step by step guide with example](#)
14. [6 User Experience Tips for Designing Your Best Chatbot](#)

Anexo III - Estratégias para disponibilizar grandes datasets para a comunidade de aprendizagem de máquina

Ao trabalhar com aprendizado de máquina, é inevitável a utilização de grandes quantidades de dados. Neste documento serão apresentadas as estratégias estudadas e a adotada para disponibilizar para a comunidade os dados utilizados durante as pesquisas realizadas neste repositório.

O Salic é o sistema de acompanhamento de projetos culturais do Ministério da Cultura. Atualmente, tal sistema conta com mais de 1.5TB de dados. Muitos desses dados são vídeos e imagens e não serão utilizados nas pesquisas do salic-ml. A expectativa é a utilização de algumas dezenas de GB de dados, retirados do banco de dados do Salic. Para encontrar a melhor estratégia para armazenar tais dados, foram estudadas as seguintes estratégias:

- Utilização do Git-LFS
- Repositório Git para dados como submódulo do salic-ml
- Repositório Git-LFS para dados como submódulo do salic-ml
- Google Drive, Dropbox e similares
- Servidor FTP local com acesso externo

A estratégia adotada foi a utilização de um servidor FTP na infraestrutura local com acesso externo, devido a sua maior capacidade de armazenamento e facilidade de download dos dados. Abaixo são detalhadas as estratégias utilizadas.

Utilização do Git-LFS

O [Git-LFS](#) (Large Files Storage) é uma extensão do Git para versionamento de grandes arquivos de dados em um repositório. Além de facilitar o versionamento de grandes arquivos, o Git-LFS permite o download de tais arquivos individualmente e separados do `git clone`. Contudo, o GitHub permite o armazenamento de, no máximo, 1GB por repositório [\[link\]](#), o que não é suficiente para armazenar os dados utilizados. A primeira conclusão é que é preciso armazenar os dados fora do GitHub.

Repositório Git para dados como submódulo do salic-ml

Estudou-se a possibilidade de criação de um repositório Git para armazenamento dos dados

em uma outra plataforma e utilização de tal repositório como um submódulo do salic-ml. A primeira plataforma estudada foi o [GitLab](#), contudo, ele permite o armazenamento de, no máximo, 10GB por repositório [[link](#)], o que também não é suficiente para armazenar os dados utilizados. Uma alternativa foi a utilização de uma instância do [Gitea](#) na infraestrutura local e com acesso externo. Desta forma, seria possível armazenar todos os dados necessários sem restrição de espaço e ainda utilizar o repositório como submódulo do salic-ml. Uma desvantagem desta abordagem é a impossibilidade de baixar arquivos individuais do repositório: o comando `git clone` baixa todo o repositório em uma única vez.

Repositório Git-LFS para dados como submódulo do salic-ml

Para contornar a impossibilidade de baixar arquivos individuais do repositório de dados, estudou-se a possibilidade de utilização do repositório em uma instância local do Gitea com a extensão Git-LFS, desta forma, não haveria limitação de capacidade de armazenamento de dados e seria possível baixar arquivos individuais do repositório. A desvantagem desta abordagem é a complexidade de sua utilização, não seria prático ter que utilizar os comandos específicos do `git-submodules` e do `git-lfs` sempre que fosse necessário baixar um arquivo de dados.

Google Drive, Dropbox e similares

Para simplificar a utilização dos arquivos de dados, foram estudadas plataformas de armazenamento em nuvem, como [Google Drive](#), [Dropbox](#) e [OneDrive](#), contudo, todas as ferramentas têm limitações de capacidade de armazenamento de dados e, assim, não são viáveis.

Servidor FTP

Por fim, foi estudada e adotada a utilização de um servidor FTP na infraestrutura local com acesso externo. O servidor FTP conta com acesso de leitura para usuários anônimos e escrita para um usuário administrador. É possível baixar todos os dados ou arquivos individuais, além de não ter limitação para armazenamento. Para baixar todos os dados ou apenas dados específicos, basta acessar o servidor FTP, uma maneira para baixar os dados facilmente é utilizando o comando `wget`, conforme mostrado no README deste repositório. Os arquivos devem ser salvos na pasta `data` do repositório local, pasta ignorada pelo Git.

Acesso aos dados

Todos os datasets (em `.csv`) usados neste projeto são obtidos a partir do banco de dados do Salic. Este documento explica brevemente o processo de obtenção dos datasets.

1. Conecte-se com o banco de dados do Salic via VPN: [Guia de acesso ao banco de dados via VPN](#)
2. Identifique os bancos de dados, tabelas e colunas relevantes ao seu objetivo.
3. Execute comandos/scripts SQL para exibir os dados de interesse:

```
SELECT * FROM Sac_db.tbPlanilha;
```

4. No DBeaver os resultados dos comandos executados são exibidos na aba/view "Results"
5. Clicando com o botão direito em qualquer linha do resultado da aba "Results", selecione a opção "Export resultset"
6. Escolha a opção CSV
7. Configure a saída de acordo com seu interesse, em geral as opções padrão são suficiente.
8. O CSV será baixado será salvo no local escolhido.

Anexo IV - Template de notebooks para algoritmos de aprendizagem de máquina usando dados da Lei Rouanet

A frente tem como meta para a primeira entrega a implementação de um sistema de indicadores de complexidade de projetos culturais. Ao fim das primeiras 4 entregas, a meta do grupo é entregar todo o sistema de informações de projetos na etapa de análise de resultados, e a meta de toda a Fase 2 é entregar os dois sistemas listados acima (sistema de informações e sistema de recomendações).

As entregas ainda foram divididas em *sprints* de duas semanas e a frente se organizou em três equipes:

1. Time de pesquisa e geração: responsável pelo estudo dos dados do Salic e geração de métricas e indicadores que possam ajudar os técnicos da Sefic (Secretaria de Fomento e Incentivo à Cultura) a analisar projetos culturais, por exemplo: métricas de nível de complexidade de projetos culturais, semelhança entre projetos culturais, entre outras.
Repositório da equipe: [salic-ml](#)
2. Time de disponibilização de dados: responsável por disponibilizar as métricas e

indicadores gerados pelo time de pesquisa e extração para que os mesmos possam ser utilizados de alguma forma na plataforma Salic ou em algum outro sistema de interesse.

Repositório da equipe: [salic-ml-web](#)

3. Time de DataViz: responsável pela integração das métricas e indicadores gerados na plataforma Salic. Repositório da equipe: [salic-ml-web](#)

Todas as *sprints* contam com uma reunião inicial de planejamento com duração de aproximadamente 3 horas, reuniões não formais ao longo do período de desenvolvimento (conforme a necessidade) e, ao final, uma reunião de revisão e retrospectiva. As reuniões inicial e final são realizadas, idealmente, com todos os membros dos três times da frente.

Como todo o produto será software livre e disponibilizado na plataforma [GitHub](#), o grupo decidiu por utilizar o GitHub com a extensão [ZenHub](#) para gerenciar o backlog de tarefas.

Ao início de cada entrega, as tarefas de todas as equipes são cadastradas no repositório `salic-ml` do GitHub como *issues* e adicionadas à *milestone* referente àquela entrega. À cada entrega é dado o nome de uma ponte famosa (Golden Bridge, JK Bridge, entre outras) para facilitar a associação da mesma às suas tarefas.

Todas as *issues* são adicionadas também ao *board* "Backlog" do ZenHub. Durante a reunião de planejamento, as tarefas da *sprint* são priorizadas, atribuídas aos membros do time e movidas para a aba "Sprint Backlog". Durante a *sprint*, as *issues* em andamento são movidas para a aba "Doing". Após finalizadas, é aberto um *pull request* e, assim que ele é revisado e aprovado, a *issue* é movida para a aba "Done". Durante a reunião de revisão e retrospectiva, as *issues* finalizadas são movidas para a aba "Closed", o que encerra o ciclo de vida de uma tarefa.

Todas as tarefas devem ter critérios de aceitação bem definidos em sua descrição. É comum para todas as tarefas que a documentação e a criação de testes automatizados sejam escritos como critério de aceitação.

Por fim, aqui na *Wiki* são documentadas todas as reuniões da frente e eventuais estudos sobre temas relacionados ao projeto.

Os notebooks são disponibilizados em <https://github.com/lappis-unb/salic-ml/tree/master/notebooks>.

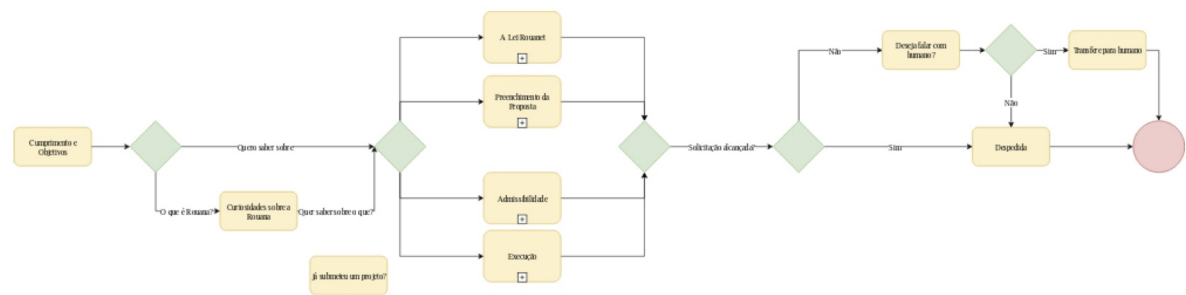
Exemplo de um notebook:

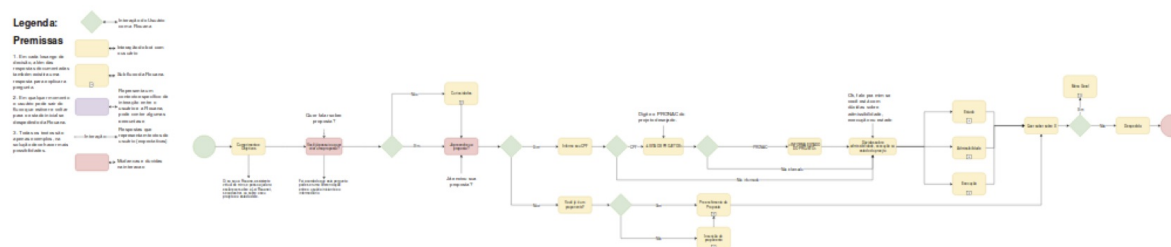
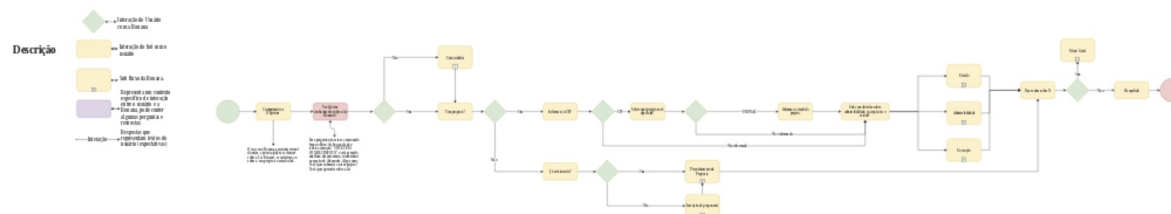
https://github.com/lappis-unb/salic-ml/blob/master/notebooks/exploratory/project_historic_situation-v0.1.ipynb

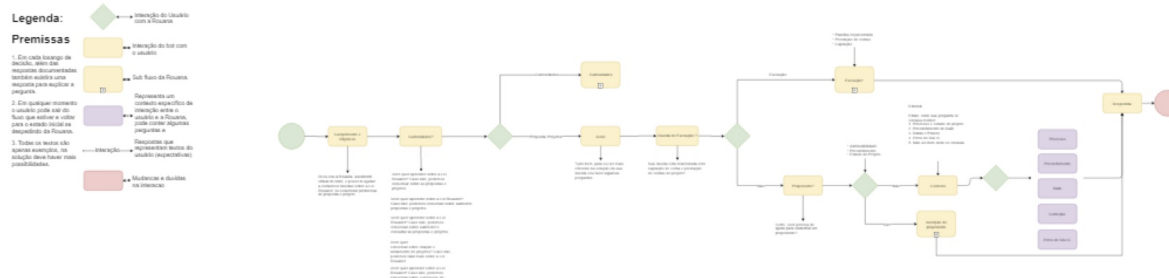
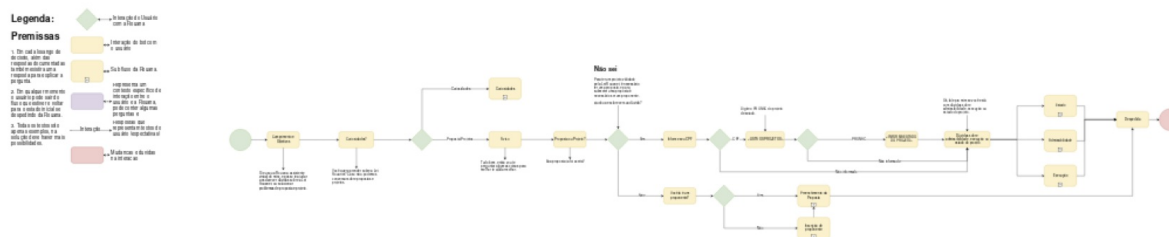
Versões Fluxo Rouana

v0_fluxo_principal_rouana

25/08/2018 08:46







Análise da prototipação em código utilizando framework javascript de componetização

Com base na nova metodologia de trabalho inspirada no design sprint. Foi notado a necessidade da utilização de ferramentas que possibilitem a extração do melhor da interação entre desenvolvedores e designers de forma a proverem maior velocidade e organização para as validações das ideias desenvolvidas. A partir disso foi questionado se a utilização de frameworks de componentização se encaixaria bem nesta fase de prototipação de ideias. Motivados por este desafio, foi proposta a análise de um framework javascript tomando como base esse cenário.

A análise se baseará em 3 principais etapas:

- **Componetização dos elementos desenvolvidos:** componetizar todas ideias prototipadas até o momento.
- **Capacitação básica dos membros da equipe:** explicar como será o workflow de desenvolvimento.
- **Análise dos resultados:** após duas sprints, será analisado o resultado obtido verificando se com base nas métricas definidas essa escolha atrapalhou ou ajudou e em quais aspectos.

Framework escolhido

Para a escolha do framework javascript não foi feita nenhuma análise profunda a respeito de qual seria a melhor ferramenta. Visto que o foco desta análise é a respeito da utilização de componentes no desenvolvimento de prototipações em código. A escolha foi baseada na facilidade de implementação de componentes e na sintaxe limpa que o framework oferece. Sendo assim o framework escolhido para a análise foi o: Vuejs v2.0.

Execução do projeto

Na raiz do projeto devemos instalar todas as dependências do projeto.

```
// Instalação via yarn
yarn install
// Execução via yarn
yarn run dev
// Instalação via npm
```

```
npm install
// Execução via npm
npm run dev
```

Métricas para indicar desempenho

Os parâmetros que estão sendo analisados são:

- Esta proposta possibilitou a melhor organização dos protótipos desenvolvidos?
- Esta proposta facilitou a prototipação em código do posicionamento de elementos na tela tanto para a versão desktop quanto para a versão para mobile?
- Esta proposta diminuiu esforços quanto a duplicação de código?
- Esta proposta possibilitou a prototipação em código de várias versões da mesma ideia de forma ágil?

Organização dos protótipos desenvolvidos.

Para garantir a organização dos protótipos desenvolvidos foi adotado a seguinte estrutura:

- **~/src/pages:** contém todas as páginas do projeto, cada página representa uma ou mais ideias de prototipação.
- **~/src/components:** contém todos os componentes desenvolvidos que compõem as páginas.
- **~/src/routes/index.js:** contém todas as rotas da aplicação.
- **~/src/App.vue:** representa o principal componente da aplicação, na qual todas as outras páginas serão renderizadas nesta. É também utilizada como o template básico das páginas.

Criação de componentes

Vamos criar um simples componente. Primeiramente, deve se acessar o diretório

~/src/components__ na qual, devemos criar todos os componentes que irão compor as páginas. Dentro deste diretório, criaremos o arquivo *_HelloWorld.vue*. Desta forma, notamos 3 principais informações, todos os arquivos relacionados aos componentes devem estar com a primeira letra maiúscula obedecendo o padrão de Camel Case e a extensão de um componente é *.vue_*.

Após o arquivo **~/src/components/HelloWorld.vue** ser criado iremos escrever no arquivo as estrutura básica de como o *Vuejs* define um componente.

```

<template>
  <!-- Código HTML -->
</template>

<script>
  // Código Javascript
</script>

<style>
  /* Código css */
</style>

```

Agora vamos inserir um texto contendo o conteúdo do nosso componente:

```

<template>
  <!--Note que todos os elementos HTML devem estar dentro de uma tag HTML, mas é
  <div class="Hello-world">
    <h1>Hello World</h1>
    <p>Este é um componente no Vuejs</p>
  </div>
</template>

<script>
  // Por convenção devemos exportar o nome do componente.
  export default {
    name: 'HelloWorld',
  }
</script>

<style>
  /* Código css */
</style>

```

Agora devemos importar o nosso componente na página que queremos que o mesmo seja exibido.

```

<template>
  <div id="app">
    <!--Tag representando o componente HelloWorld-->
    <hello-world/>
    <router-view/>
  </div>
</template>

<script>
  // É assim que importamos o componente, o '@' representa a raiz do projeto.

```

```
import HelloWorld from '@components/HelloWorld'

export default {
  name: 'App',
  components: {
    // hello-world é o nome da tag que utilizaremos dentro do nosso app.
    // HelloWorld é o componente que importamos
    'hello-world': HelloWorld
  }
}
</script>

<style>
</style>
```

Pronto! assim criamos o nosso primeiro componente e importamos ele dentro de um outro componente.

Criação de uma página

Uma página nada mais é do que um componente que é composto por outros componentes. Desta forma o tutorial de criação de um componente se aplica da mesma forma a criação da página. Havendo somente duas diferenças: haverá vários componentes dentro da página e devemos importar a página no `~/src/router/index.js` e **criar uma nova rota para conseguirmos acessa la. Então vamos criar uma rota para o nosso componente `_HelloWorld.vue`, vamos considerar agora que este componente é uma página portanto a partir da agora este componente se encontra dentro do diretório `__~/src/pages/`.**

Vamos importar o componente `HelloWorld.vue` no arquivo `__~/src/router/index.js`.

```
`` javascript
import Vue from 'vue'
import Router from 'vue-router'
// Importando a página HelloWorld
import HelloWorld from '@pages/HelloWorld'
```

```
Vue.use(Router)
```

```
export default new Router({
  routes: [
    {
      //Caminho que acessaremos pela url
      path: '/',
      name: 'HelloWorld',
```

```
component: HelloWorld
}
]
})
""
```

Desta forma, é só executar o projeto. E acessar a raiz do projeto.

Análise de biblioteca de mapas

Nome	Licença	Compatibilidade com GPL
Chartkick	MIT	Sim
Google Charts	Apache 2	Sim
jQuery Mapel	MIT	Sim
D3js	3-clause BSD	Sim
Kartograph	LGPL / MIT	Sim
KnightLab (Timeline/Mapa)	MPL 2	Sim
AMCharts	Pago	Não
Leaflet	Copyright (All rights reserved)	A confirmar
Polymaps	Copyright (All rights reserved)	A confirmar
Código de mapa	Sem licença	-
Maujor	Sem licença	A confirmar
ChartJS		
JsMaps	MIT	Sim

JsMaps

Essa biblioteca foi utilizada na [issue de implementar a ilustração do mapa do Brasil](#). Como pontos positivos dessa experiência, pode-se destacar:

- Boa documentação ([fórum](#), por exemplo, contém vários tópicos interessantes)
- Bom uso de atributos que facilitam:
 - Criar regiões incluindo os estados
 - Colorir estados ou regiões individualmente
 - Colorir de forma diferenciada quando houver hover
 - *Hooks* que facilitam associar ações ao clique no mapa
 - Inserir ícones no mapa com tamanhos diferenciados

Todas essas características já foram o suficiente para suprir o escopo da *issue*. Não foram observados pontos negativos notáveis sobre o uso dessa biblioteca.