CAVC – Automação de Reservas de Salas – Manual de Implementação

# Introdução

Este documento serve como um guia para a implementação do sistema de automação de reservas na FEA. O objetivo é fornecer uma solução eficiente que agilize o processo de solicitação de reservas, substituindo fluxos de trabalho manuais.

## Dos Pré-Requisitos

Para implementação correta deste projeto, além de uma **Conta Google** ativa, é necessário criar uma conta no **GitHub** para gerenciar o código-fonte do projeto e uma conta no **Render** (plano gratuito) para hospedar o serviço de automação.

# Implementando o código de automação

Aqui é necessário publicar a API em Python que automatiza as interações com o site de reservas. Usaremos o **Render**, uma plataforma que hospeda sua aplicação web.

## Obtendo uma Cópia do Código (Fork)

A primeira etapa é fazer um “fork” do repositório no GitHub..

* Acesse o repositório do projeto no GitHub:

[Igor0Pires/CAVC-AutomationRoom](https://github.com/Igor0Pires/CAVC-AutomationRoom)

* No canto superior direito da página, clique no botão **Fork** e na página que abrir clique em **Create Fork**.

A screenshot of a computer

AI-generated content may be incorrect.

**A screenshot of a computer

AI-generated content may be incorrect.**

Depois de fazer o fork, o repositório será copiado para sua conta do GitHub. É esta cópia que o Render irá usar para construir a sua aplicação.

A screenshot of a computer

AI-generated content may be incorrect.

## Conectando ao Render

Agora, criaremos o serviço de hospedagem para a sua API.

* Faça o login na sua conta do Render.
* No painel principal ([Render Dashboard](https://dashboard.render.com/)), clique em New Web Service.

A screenshot of a computer

AI-generated content may be incorrect.

Imagem 2.2.1: Botão “New Web Service” no painel do Render

* O Render irá pedir para você conectar sua conta do GitHub. Autorize a conexão e siga os passos, em seguida, selecione o repositório que você acabou de fazer fork.

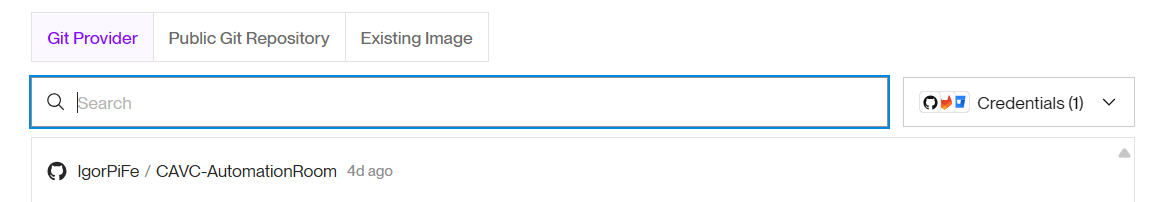
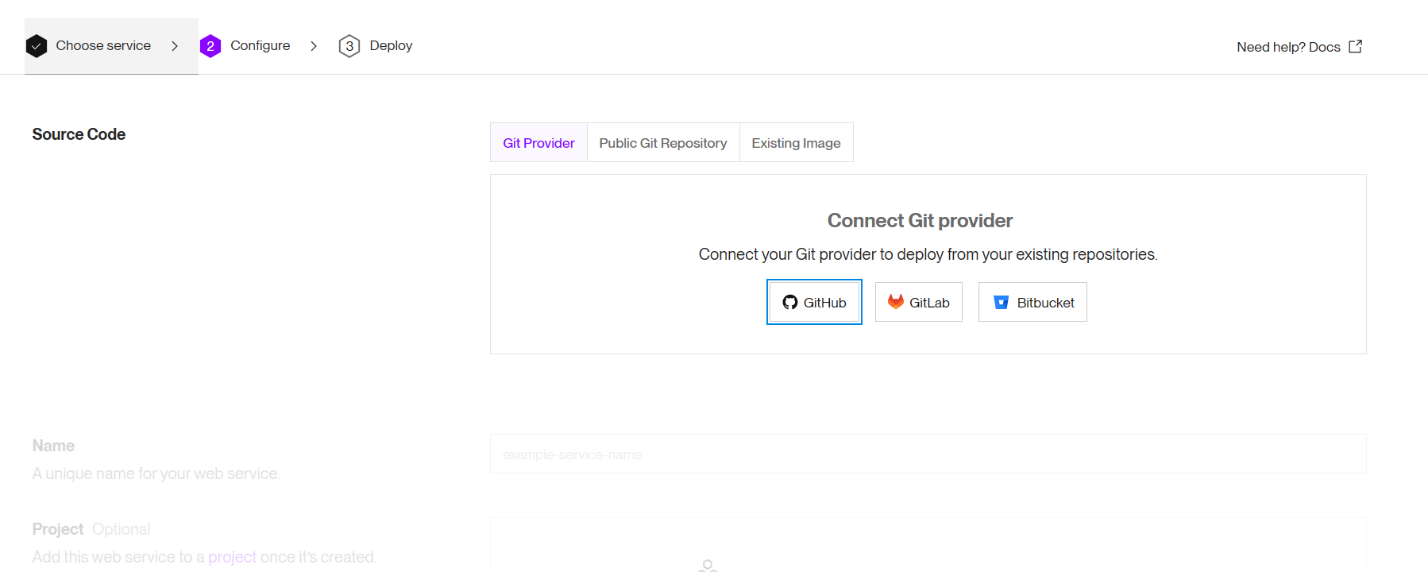


Imagem 2.2.2: Opção do GitHub de conexão e seleção do repositório

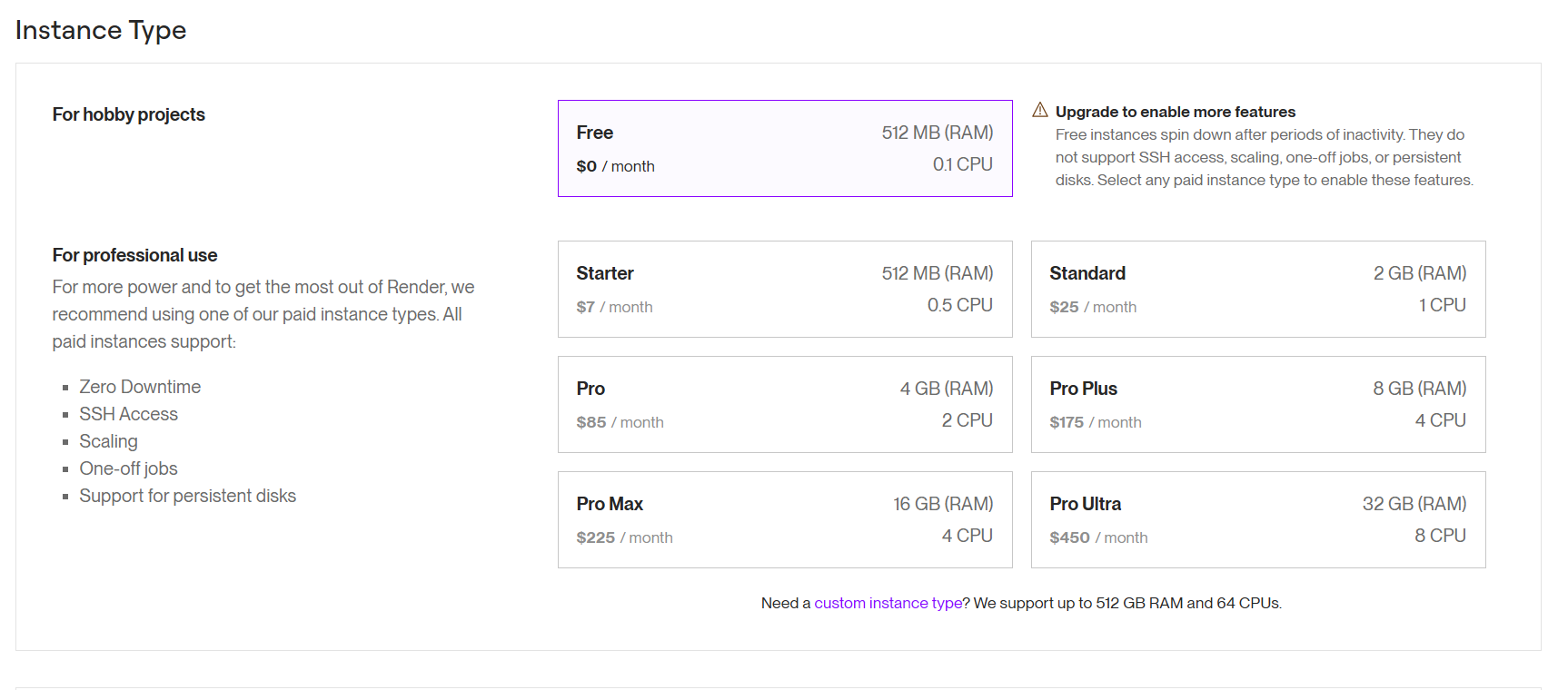
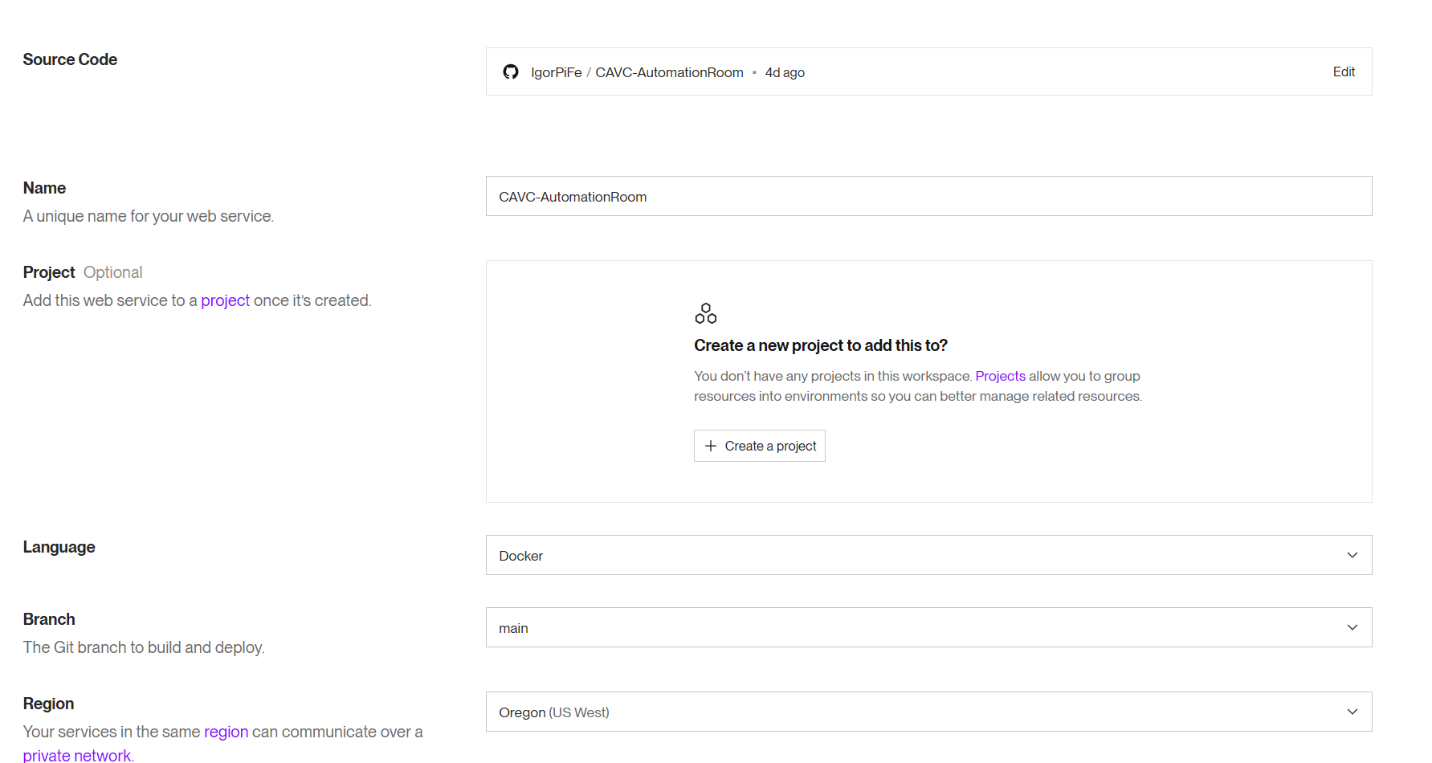
## Defina as Configurações da API

Nesta etapa, informaremos ao Render como ele deve construir e executar sua API.

* **Repository:** Selecione o repositório que você copiou
* **Name:** Dê um nome ao ser serviço, por exemplo: **CAVC-AutomationRoom**
* **Branch:** Mantenha o padrão, que geralmente é **main**

## 2.4. Configure as Variáveis de Ambiente

Imagem 2.3.1: Configurações do serviço no Render



Informações sensíveis como senhas e chaves de API nunca são armazenadas diretamente no código. Por isso, usamos “Variáveis de ambiente”:

* Localize a seção **Envirionment Variables** e clique em **Add Enviromenment Variable**
* Adicione as variáveis, uma por uma, com os valores correspondentes, para mais informações contate o responsável pela automação

A screenshot of a computer

AI-generated content may be incorrect.

Imagem 2.4.1: Adicionando variáveis de ambiente no Render (exemplo)

## Publique!

Depois de preencher todas as informações, role para o final da página e clique em **Deploy Web Service.**

* O Render começará o processo de implantação. Ele irá baixar o código, construir e iniciar o serviço (pode demorar alguns minutos).
* Você pode acompanhar o progresso no painel de logs
* A screenshot of a computer program

  AI-generated content may be incorrect.Quando a implantação for concluída com sucesso, o status do serviço mudará para **Live** e você verá uma **URL pública** na parte superior da página.

Imagem 2.5.2: Tela de implantação em progresso e o status “Live”

**Atenção:** Guarde a URL da sua API. Ela será a chave da conexão com o Apps Script no próximo passo!

# Configurando Apps Script

Nesta fase, iremos configurar o lado do Google, que irá receber as solicitações, armazená-las e enviar os comandos para a API que você acabou de implantar no Render.

## Formulário e a Planilha de Controle

* Abra o formulário fornecido pelo responsável.
* No menu do formulário, clique no botão de três pontos e selecione **Fazer uma cópia** e dê um nome.
* Quando abrir na primeira vez poderá aparecer uma mensagem, isso acontece porque a pasta onde ficam salvos os arquivos de upload não existe no seu drive, clique em **Restaurar** e verifique se a pergunta “fazer upload de Ofício” está obtendo os arquivos, caso não esteja siga os passos para fazê-lo.
* Depois de ter a sua cópia, clique no ícone de Respostas e selecione **Link para as planilhas.** E crie uma nova planilha, esta será a sua planilha de controle, onde o status das solicitações será atualizado, dê um nome a ela.

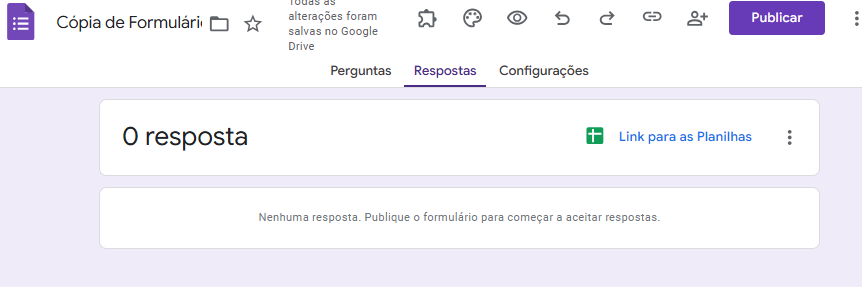
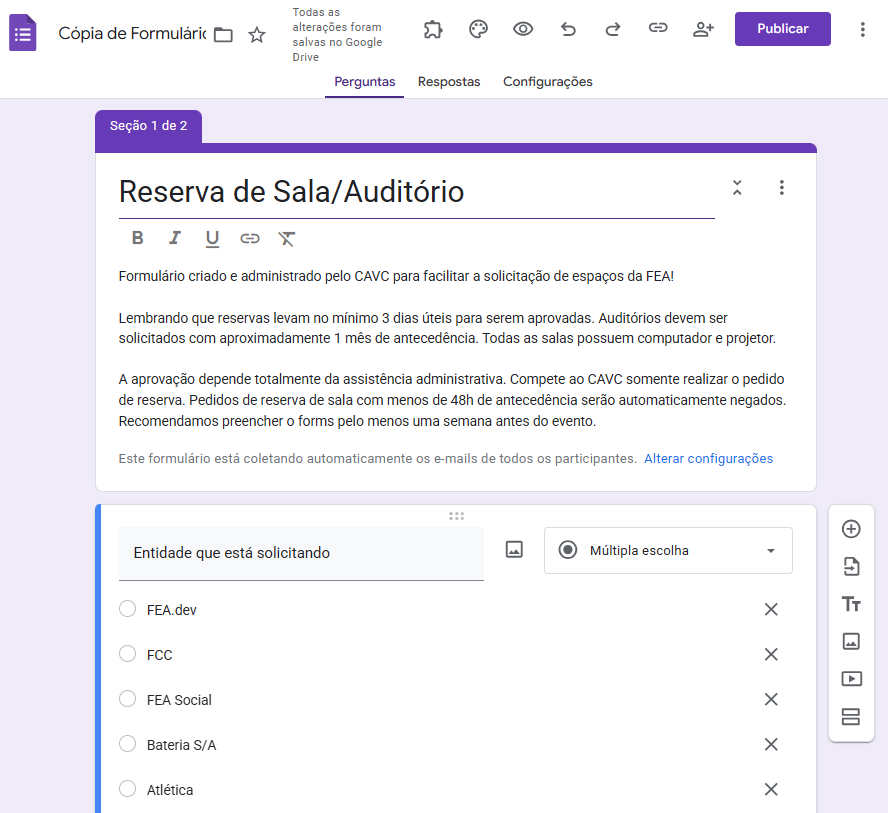


Imagem 3.1.1: Fluxo para criação da planilha

* **Importante:** Após a criação da planilha, você verá as colunas, cada coluna se referindo a uma pergunta, adicione três novas colunas no início, exatamente com os nomes (Status, Link do formulário, Tentativas).

A screenshot of a computer

AI-generated content may be incorrect.

## Configurando o código

A screenshot of a computer

AI-generated content may be incorrect.Por sorte, ao copiar o formulário o Google Forms copia o código de forma automática, para verificar, clique no botão de três pontos e selecione **Apps Script**, se estiver correto, deve abrir uma página como está na imagem a seguir (caso não esteja contate o responsável).

## Implantando o App da Web

A funcionalidade do botão de confirmação do e-mail só funciona se o seu script for publicado como um Web App.

* No editor do Apps Script, clique em **Implantar > Nova Implantação**
* Clique no ícone de engrenagem ao lado na janela que abrir de “Selecione tipo” e marque **App da Web** (as vezes vem marcada por padrão)
* Configure a implantação:
  + **Executar como:** Eu (email),
  + **Quem pode acessar:** Qualquer pessoa
* Clique em **Implantar**
* O Google pedirá uma permissão inicial. Autorize o script
* Ao final, você receberá a URL do App da Web. Guarde-a

A screenshot of a computer

AI-generated content may be incorrect.

## Configurando as variáveis ambiente

Para facilitar, todas as chaves e URLs de configuração serão armazenadas em um único lugar em **Propriedades do Script.**

* No editor do Apps Script, vá em **Configurações do projeto** (ícone de engrenagem) no menu lateral.
* Role até a seção **Propriedades do script** e clique em **Adicionar propriedade do script**
* Adicione as variáveis, uma por uma.

A screenshot of a computer

AI-generated content may be incorrect.

Imagem 3.4.1 Adicionando as propriedades do script

# Execução e Gatilhos

Agora, ativaremos os gatilhos que farão com que toda a automação funcione de forma autônoma. Um “gatilho” no Google Apps Script é um evento que dispara a execução de uma função automaticamente, sem a necessidade de intervenção manual.

## Visão geral dos arquivos de código

Antes de configurar os gatilhos, é importante entender a função de cada parte do código, volte para o editor (ícone [**<>**]):

* **1.CreateTrigger.gs:** Este arquivo contém uma função *createFormSubmitTrigger()* que criará um gatilho para toda vez que um formulário seja respondido, **Execute-o** agora, clicando em Executar em cima do código, essa deve ser a resposta esperada:

A screenshot of a computer

AI-generated content may be incorrect.

* **Utils.gs:** Este arquivo contém a função *onFormSubmit(),* que é o ponto de partida do fluxo. Ele é acionado imediatamente após um novo formulário de reserva ser submetido. Sua principal responsabilidade é enviar um e-mail ao responsável com o link de confirmação.
* **WebApp.gs:** Este arquivo hospeda a função *doGet()*, que é ativada quando o responsável clica no link de confirmação do e-mail. A função *doGet()* atualiza o status na planilha para “Confirmado”.
* **Request.gs:** Este é o coração da automação. Um gatilho de tempo rodará esta função periodicamente. Ela verifica a planilha por solicitações com status “Confirmado” e faz a requisição à sua API no Render para iniciar a automação no site da FEA
* **GetStatus.gs:** Este arquivo é dedicado a automação secundário. Um segundo gatilho de tempo o executará para verificar o status o das reservas e atualizá-lo na planilha de status.

## Configurando o Gatilhos de Tempo

### 4.2.1 Gatilho 1: *request()*

1. Na tela de gatilhos (ícone de relógio), clique em **Adicionar gatilho**

2. Configure as opções:

* **Escolha a função que será executada:** escolha a função *request*
* **Escolha qual implantação deve ser executada:** escolha a última versão implementada
* **Selecione a origem do evento:** escolha “Baseado no tempo”
* **Selecione o tipo de acionador com base no tempo:** escolha “Contador de minutos”
* **Selecione o intervalo de minutos:** A cada 10 minutos
* A screenshot of a computer

  AI-generated content may be incorrect.3. Clique em **Salvar**

### 4.2.1 Gatilho 2: *getStatus()*

Espere 5 minutos para adicionar este gatilho:

1. Na tela de gatilhos, clique novamente em **Adicionar gatilho**

2. Configure as opções:

* **Escolha a função que será executada:** escolha a função *getStatus*
* **Escolha qual implantação deve ser executada:** escolha a última versão implementada
* **Selecione a origem do evento:** escolha “Baseado no tempo”
* **Selecione o tipo de acionador com base no tempo:** escolha “Contador de minutos”
* **Selecione o intervalo de minutos:** A cada 10 minutos

3. Clique em **Salvar**

# 5. Testes e Verificação

Ótimo, neste passo verificaremos todas as partes do sistema para garantir que esteja funcionando como esperado.

**Importante:** Para fins de teste e segurança, o endpoint da API no Render está configurado para não enviar o formulário de solicitação, ele apenas preenche e faz a cópia da página e o último passo de submissão será ignorado.

## 5.1. Testando o Fluxo Completo

1. **Inicie uma Nova** **Solicitação:**

* Abra o formulário do Google Forms e preencha uma solicitação de reserva completa.
* Verifique sua planilha de controle. Uma nova linha deve aparecer com o status **Pendente.**

2. **Confirme a Solicitação:**

* Acesse o e-mail do responsável e de quem enviou o formulário pela solicitação. Você deve ter recebido um e-mail com os links de confirmação e o solicitante recebido um e-mail confirmando a solicitação.
* Clique no botão de **Confirmação**. A página do seu Web App se abrirá com uma mensagem de sucesso, e o status da sua planilha mudará para **confirmado.**

3. **Aguarde a Automação:**

* Aguarde o gatilho de tempo do *request()* ser executado (10 min.)
* No editor do Apps Script, vá em **Execuções** (o ícone de lista) para verificar se a função *request* foi concluída com sucesso

4. **Verifique os Resultados:**

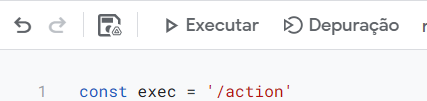
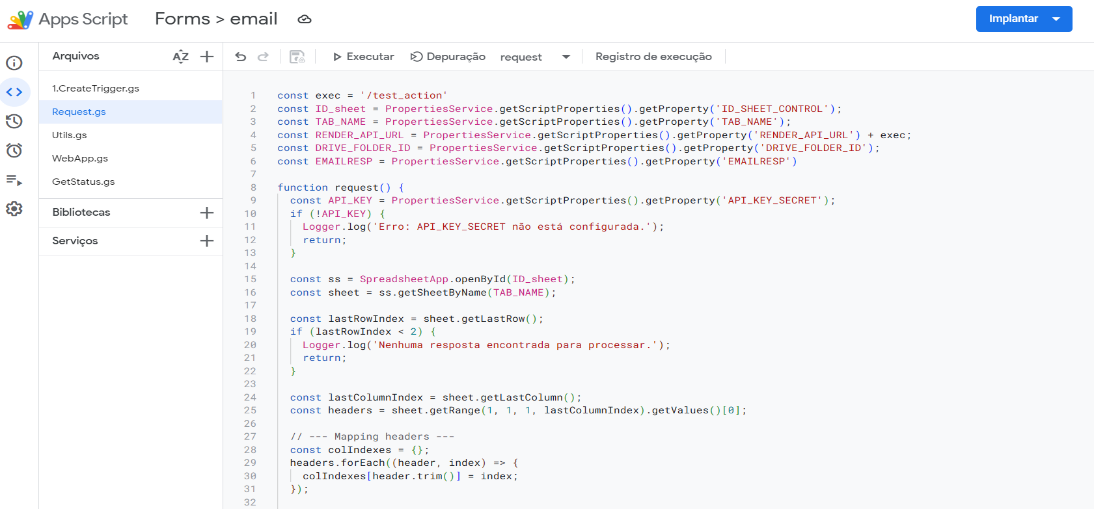
* **Planilha:** O status da sua solicitação na planilha deve mudar para **Sucesso.**
* **Google Drive:** Um PDF de teste será salvo na sua pasta que você configurou
* **E-mail:** O responsável e o solicitante devem ter recebido um e-mail de confirmação final, incluindo o link para o PDF de teste

Caso aconteça qualquer erro, chame o responsável.

## 5.2. Habilitando a submissão

Após a conclusão dos testes, você pode habilitar a automação para enviar a solicitação para a FEA.

* No editor do Apps Script, abra o arquivo *Request.gs*
* Localize a variável *exec*, na primeira linha
* Mude *‘/test\_action’* para *‘/action’*
* **Importante:** Salve o arquivo clicando em salvar projeto no drive, e agora, qualquer solicitação confirmada será enviada para a FEA.
* Teste os passos do **5.1.** preenchendo um novo formulário
* Ao final, o PDF de confirmação será a confirmação real gerada pelo site da FEA, que será salva na sua pasta do Google Drive



# 6. Considerações finais

6.1. O sistema é construído com base em ferramentas de custo zero, o que garante a viabilidade a longo prazo sem despesas operacionais. No entanto essa abordagem traz algumas limitações inerentes, como o “repouso” do serviço de automação após períodos de inatividade e quotas de uso diário, que não deve ser problemas para a demanda de usos do centro acadêmico

6.2. O sistema foi projetado com uma lógica de retentativas. Quando a automação de reserva falha, o status na planilha muda para **“Falha”**, e o gatilho de tempo tentará reprocessar essa solicitação por um máximo de **três vezes**. Se, após a terceira tentativa, a automação continuar falhando, s solicitação será permanentemente marcada como “Falha”, exigindo uma intervenção manual. Implementei essa lógica para que problemas temporários como serviço fora do ar não impeçam a conclusão da tarefa, e ao mesmo tempo que impede uma falha persistente consuma os recursos do sistema.

6.3. Sobre a personalização e manutenção do código:

6.3.1. Antes de habilitar o envio final para o site da FEA, realize todas as suas alterações e testes utilizando o modo de teste (*/test\_action*). **Lembre-se de salvar sempre.**

6.3.2. Qualquer alteração que afete o fluxo da automação, especialmente seletor de páginas ou ordem dos campos, deve ser comunicada e validade com o responsável do projeto

6.3.3. Considere criar uma cópia do projeto para experimentação. Em um ambiente de teste, você pode simular falhas e testar novos recursos sem afetar a operação principal.