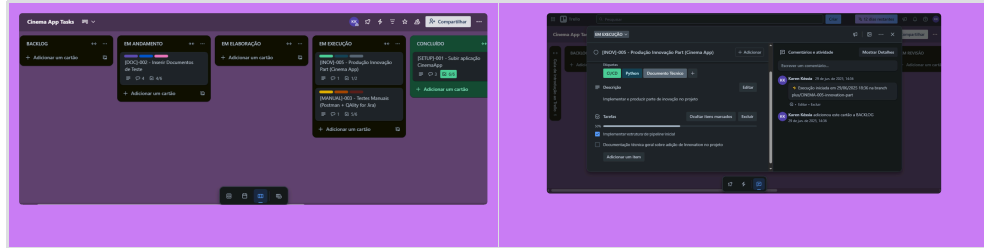


# Documentação Geral - Innovation > Git + Trello + Bitbucket



## Sobre [🔗](#)

"Automação auxiliar no processo de CI/CD voltada para a rastreabilidade de QA, integrando versionamento (Bitbucket) e gestão visual (Trello) para garantir visibilidade contínua do progresso técnico com mínimo esforço manual." [🔗](#)

Este documento descreve a camada de **Inovação Técnica** implementada no projeto **Cinema App**, com foco na **automação da rastreabilidade das atividades executadas pelo time de QA**. Essa automação foi desenvolvida integrando o **Bitbucket Pipelines** ao **Trello**, promovendo um fluxo de trabalho mais eficiente, auditável e conectado ao versionamento do código.

A proposta visa **eliminar atualizações manuais no board**, reduzir falhas humanas, e garantir uma **visão em tempo real do andamento das tarefas de qualidade**, desde a preparação dos testes até a finalização das entregas. Essa abordagem contribui diretamente para a **transparência, produtividade e alinhamento entre QA e desenvolvimento**.

## Arquitetura da Inovação Aplicada

A implementação se baseia em dois componentes fundamentais: [🔗](#)

- `bitbucket-pipelines.yml`
  - Executa ações automatizadas conforme o tipo de branch.
  - Detecta **Pull Requests** e move os cards para `EM REVISÃO`.
  - Realiza o update automático do card para `CONCLUÍDO` em merges na main/master.
- `scripts/trello_automation.py`
  - Gera **cards inteligentes**, com títulos, descrições e checklists organizados.
  - Utiliza variáveis seguras para IDs e tokens, promovendo segurança e escalabilidade.
  - Possui lógica inteligente que movimenta os cards conforme o contexto do Git.

### Mapeamento Inteligente entre Branches e Cards

Um dos pontos centrais dessa automação é o **mecanismo de mapeamento direto entre branches Git e tarefas do Trello**, implementado por meio de um dicionário estruturado ( `branch_to_card` ) no script Python.

Cada branch segue uma convenção ( `tipo/CINEMA-XXX-descricao` ) e está ligada a um `CARD_ID` e `CARD_TITLE` específicos, o que possibilita:

- **Identificação automática** da tarefa vinculada à branch ativa;
- **Criação de cards** no Trello caso ausentes, com checklist e descrição contextualizados;
- **Movimentação automatizada** entre listas conforme o estágio da entrega;
- **Registro de ações** com comentários detalhados, reforçando a auditabilidade.

Funcionalidades Automatizadas

Ação no Git	Atualização no Trello
Commits em branches comuns	Move o card para EM ANDAMENTO ou EM EXECUÇÃO
Pull Request detectado	Move para EM REVISÃO
Merge na branch principal	Move para CONCLUÍDO
Card não localizado	Criação automática em BACKLOG
Comentários e Logs	Adição de timestamp e contexto técnico
Checklists automáticos	Geração de itens padrão pré-configurados

Impacto para o QA

Ao adotar essa automação, o processo de QA torna-se mais ágil, confiável e alinhado com as práticas modernas de engenharia de software. Seus principais benefícios incluem:

- **Eliminação de atualizações manuais** no quadro de tarefas.
- **Rastreamento automático** da execução das atividades QA a partir de eventos reais no repositório.
- **Maior visibilidade** para todo o time, desde o planejamento até a entrega.
- **Redução de erros humanos**, mantendo consistência no fluxo de trabalho.
- Integração entre **repositório Git e ferramentas de gestão visual**, promovendo uma abordagem sistêmica e conectada.

Essa solução eleva o papel do QA de executor manual para **agente estratégico na garantia de qualidade contínua**, consolidando sua atuação como elo entre código, produto e entrega com valor real

Ferramentas Utilizadas

1	Git
2	Trello
3	Bitbucket
4	VSCode

Implementação Técnica da Inovação

A seguir, detalham-se os passos técnicos realizados para estruturar a área de qualidade (QA) do projeto **Cinema App**, com foco na automação e rastreabilidade de testes utilizando **Bitbucket**, **Trello** e **Robot Framework**.

1. Configuração do Repositório (Bitbucket)

1.1 Criação do Repositório

- Plataforma: [Bitbucket.org](#)
- Repositório: `cinema-app`
- Descrição: Repositório para centralizar testes do Cinema App
- Visibilidade: conforme necessidade do time (Privado ou Público)
- Configurações iniciais:

- README: habilitado
- .gitignore: Python (compatível com Robot Framework)

## 1.2 Geração e Registro da Chave SSH

- Gerar chave no terminal:

```
ssh-keygen -t rsa -b 4096 -C "seu-email@empresa.com"
```

- Caminho padrão: `~/.ssh/id_rsa`
- Copiar chave pública:

```
cat ~/.ssh/id_rsa.pub
```

- Adicionar chave no Bitbucket:
  - Navegar em **Settings > SSH Keys**
  - Label: `QA-Machine-[SeuNome]`
  - Colar chave pública gerada

---

## 2. Organização de Pipeline Visual (Trello) [🔗](#)

### 2.1 Criação do Board

- Nome: `Cinema App Tasks`

Lista	Descrição
BACKLOG	Demandas planejadas ou pendentes de início
EM ANDAMENTO	Atribuição e execução inicial da atividade
EM ELABORAÇÃO	Escrita de cenários, testes e documentação
EM EXECUÇÃO	Execução prática dos testes
EM REVISÃO	Validação por pares ou revisão técnica
CORREÇÕES	Ajustes pós-revisão ou falhas identificadas
CONCLUÍDO	Testes finalizados e aprovados

---

## [🔗](#) Integração Trello + Bitbucket (Automação) [🔗](#)

Esta seção descreve os passos para configurar a automação entre o Trello (para rastreabilidade de tarefas) e o Bitbucket Pipelines (para execuções automatizadas).

---

## Etapa 1 — Credenciais Trello [🔗](#)

### 1.1 Obtenção da API Key

- Acesse: <https://trello.com/app-key>
- Realize login com sua conta
- Copie a chave exibida ( `TRELLO_API_KEY` ) e salve com segurança

### 1.2 Geração do Token de Acesso

- Na mesma página, clique em **"Token"**
- Autorize o acesso clicando em **"Allow"**
- Copie o token gerado ( `TRELLO_TOKEN` )

### 1.3 Coleta dos IDs das Listas do Board

- Acesse o board desejado
- Adicione `.json` ao final da URL (ex: `https://trello.com/b/abc123/board.json` )

- Localize o bloco "lists" e extraia os IDs de cada coluna:
  - Exemplo:

```
{"id": "64a1b2...", "name": "BACKLOG"} {"id": "64b2c3...", "name": "EM ANDAMENTO"}
```

**Importante:** Mantenha esses IDs salvos — serão usados nas variáveis do pipeline.

## Etapa 2 — Variáveis no Bitbucket

### 2.1 Acesso às Configurações

- Vá até o repositório no Bitbucket
- Menu lateral: **Repository Settings** → **Repository variables**

### 2.2 Registro das Variáveis

Adicione as seguintes variáveis com o botão "Add variable":

Nome da Variável	Valor (Exemplo)	Secured?
TRELLO_API_KEY	sua_key_gerada	Yes
TRELLO_TOKEN	seu_token_gerado	Yes
TRELLO_LIST_BACKLOG	ID da lista "BACKLOG"	No
TRELLO_LIST_EM_ANDAMENTO	ID da lista "EM ANDAMENTO"	No
TRELLO_LIST_EM_ELABORACAO	ID da lista "EM ELABORAÇÃO"	No
TRELLO_LIST_EM_EXECUCAO	ID da lista "EM EXECUÇÃO"	No
TRELLO_LIST_EM_REVISAO	ID da lista "EM REVISÃO"	No
TRELLO_LIST_CORRECOES	ID da lista "CORREÇÕES"	No
TRELLO_LIST_CONCLUIDO	ID da lista "CONCLUÍDO"	No
TRELLO_LIST_BLOQUEADO	ID da lista "BLOQUEADO"	No

## Etapa 3 — Definição do Pipeline

Estrutura necessária:

```
1 seu-repositorio/
2 |— bitbucket-pipelines.yml
3 |— scripts/
4   |— trello_automation.py
```

### 3.1 Criação do Arquivo de Automação

- No repositório local, crie o arquivo `bitbucket-pipelines.yml` na raiz do projeto
- Este arquivo conterá os comandos para automação da movimentação de cards com base no status dos testes, builds ou deploys.

A lógica de automação pode incluir chamadas à API do Trello via `curl`, integração com frameworks de testes e indicadores de sucesso ou falha.

### 3.2 Criar a Pasta Scripts

1. Crie uma pasta chamada `scripts` na raiz do seu projeto
2. Dentro desta pasta, crie o arquivo `trello_automation.py`

**OBS:** Os códigos dos arquivos se encontram em:

## cinematests/cinema\_app\_final\_challenge

 Criado por cinematests · Atualizado em há 7 horas  2

Test project developed as part of the AWS & AI for Software Quality Engineering internship. This repository contains the target application for executing manual and automated tests, exploring best practices, techniques, and integration with modern tools for software quality.

 Bitbucket

## ✓ Etapa 4 - Execução Final


### Ativando a Automação no Bitbucket

 Para que o pipeline funcione corretamente, é necessário **ativar o recurso de Pipelines** no repositório. Caso ainda não tenha feito isso, siga os passos abaixo:

#### Como Ativar o Pipeline (Primeira Execução)

1. Acesse seu repositório no Bitbucket.
2. Clique em **“Create your first pipeline”**, ou siga o caminho:
  - Repository → Settings (menu lateral)
  - Selecione Pipelines → Settings
  - Ative a opção **“Enable Pipelines”** (Toggle ON)
3. Salve a configuração.


### Execução Inicial da Automação

 Após ativar os Pipelines e adicionar os arquivos `bitbucket-pipelines.yml` e `scripts/trello_automation.py` ao repositório:

- Faça um **commit em qualquer branch mapeada** ( `feature/` , `docs/` , `test/` , `plus/` etc.);
- O script será executado automaticamente no pipeline;
- Se o card ainda não existir, ele será **criado automaticamente na lista BACKLOG**;
- Cada card será preenchido com:
  - **Título e descrição predefinidos**
  - **Checklist detalhado da atividade**
  - **Comentários automáticos com timestamp e contexto técnico**

Essa execução inicial já estabelece todo o ecossistema automatizado de rastreabilidade — pronta para ser escalada.

### Conclusão

 Esta automação marca um avanço significativo na atuação do time de QA, promovendo rastreabilidade inteligente e integração contínua entre versionamento e gestão de tarefas. Ao reduzir ações manuais e refletir em tempo real o progresso das atividades no Trello, ela reforça a eficiência, transparência e o alinhamento com práticas modernas de CI/CD.