Documentação da API

ÍNDICE

1. Documentação da API

1.1. Autenticação

Esta página descreve como funciona a autenticação no serviço Latch.

Todas as requisições à API do Latch devem ser assinadas. O processo de assinatura é uma versão simplificada do protocolo Oauth de duas vias.

Cada Solicitação HTTP à API deve ser acompanhada de dois cabeçalhos de autenticação: **Authorization** e **Data**.

1.1.1. O Cabeçalho Authorization

O cabeçalho **Authorization** deve ter o seguinte formato:

```
11PATHS requestId requestSignature
```

- **requestId** pode ser um applicationId ou um userId, dependendo da API que está sendo acessada pela Solicitação.
- **11PATHS** é uma constante que determina o método de autenticação.
- **applicationId** é um identificador alfanumérico obtido ao registrar um novo aplicativo.
- **requestSignature** é uma assinatura derivada da URL, parâmetros, cabeçalhos personalizados e data da Solicitação atual, todos com hash usando um Secreto que também é obtido com o applicationId ao registrar a aplicação.

A assinatura da Solicitação é uma string codificada em Base64 e assinada com HMAC-SHA1.

A string deve ser criada seguindo este processo:

1. Começar com uma string vazia.

2. Anexar o nome do método em letras maiúsculas. Atualmente, somente **GET, POST, PUT e DELETE** é suportado.

3. Anexar um separador de linha, "\n" (código Unicode U+000A).

4. Anexar uma string com a data atual no formato exato **aaaa-MM-dd HH:mm:ss**. Esse formato deve ser autoexplicativo, mas lembre-se de que tudo é numérico e deve ser preenchido com zeros, se necessário, para garantir que todos os números tenham dois dígitos (exceto o ano, que tem quatro dígitos). Você deve manter esse valor para usá-lo no cabeçalho **Data**. A verificação da assinatura falhará se eles não corresponderem.

5. Anexar um separador de linha, "\n" (código Unicode U+000A).

6. Serializar todos os cabeçalhos específicos desta aplicação (não todos os cabeçalhos HTTP da Solicitação). Todos os nomes desses cabeçalhos começam com X-11paths-.
    1. Converter todos os nomes de cabeçalho para letras minúsculas.
    2. Ordenar os cabeçalhos por nome em ordem alfabética crescente.
    3. Para cada valor de cabeçalho, converter os cabeçalhos de várias linhas em uma única linha, substituindo quebras de linha "\n" por espaços simples " ".
    4. Criar uma string vazia. Em seguida, para cada cabeçalho após a ordenação e transformações, adicionar a essa nova string o nome do cabeçalho seguido de dois pontos ":" e o valor do cabeçalho. Cada nome:valor deve ser separado do próximo por um espaço simples " ".
    5. Remover espaços em branco no início ou no final dessa string.

7. Anexar um separador de linha, "\n" (código Unicode U+000A).

8. Anexar a string de consulta em formato URL-encoded, que consiste no caminho (iniciando pela primeira barra) e nos parâmetros de consulta. A string de consulta não deve conter o nome do host ou a porta e não deve conter espaços em branco no início ou no final.

9. Somente para pedidos POST ou PUT, anexar uma quebra de linha " " (Ponto Unicode U+000A).

10. Somente para pedidos POST ou PUT, seriar os parâmetros do pedido da seguinte forma, sendo o nome do parâmetro e seu valor, a representação em UTF-8 de sua codificação URL.

1. Ordenar os parâmetros pelo nome em ordem alfabética crescente e depois pelo valor do parâmetro.
2. Criar uma string vazia. Em seguida, para cada parâmetro após a ordenação, adicionar a essa nova string o nome do parâmetro seguido de um sinal de igual "=" e o valor do parâmetro. Cada par nome=valor deve ser separado do próximo por um “e comercial” "&".
3. Remover espaços em branco no início ou no final dessa string.

Depois de criar essa string seguindo o processo descrito, ela deve ser assinada usando o algoritmo HMAC-SHA1 e o **Secreto** obtido ao registrar a aplicação. Após a assinatura, os dados binários brutos devem ser codificados em Base64. A string resultante é a requestSignature que deve ser adicionada ao cabeçalho **Authorization**.

1.1.2. O Cabeçalho X-11Paths-Date

O cabeçalho **X-11Paths-Date** contém o valor da data UTC atual e deve ter o seguinte formato:

yyyy-MM-dd HH:mm:ss

onde aaaa é o ano, MM o mês em forma de número, dd o dia em forma de número, HH a hora em formato de 24 h, mm o minuto dentro da hora e ss o segundo dentro do minuto. Todos os valores têm que ser completados com zeros, de forma que todos eles sejam valores de 2 dígitos, exceto o ano, que tem 4.

Todos os valores devem ser preenchidos com zeros para garantir dois dígitos, exceto para o ano, que tem quatro dígitos.

É muito importante que o valor e o formato deste cabeçalho sejam exatamente os mesmos usados no processo de criação do requestSignature para o cabeçalho **Authorization**, conforme explicado acima.

Note que você pode continuar usando o cabeçalho HTTP **Data** padrão em qualquer formato desejado (por exemplo, RFC 1123). Apenas certifique-se de não confundir ambos e sempre usar o valor que você utiliza em **X-11Paths-Date** no processo de assinatura. A API ignorará o cabeçalho **Data** padrão.

Erros

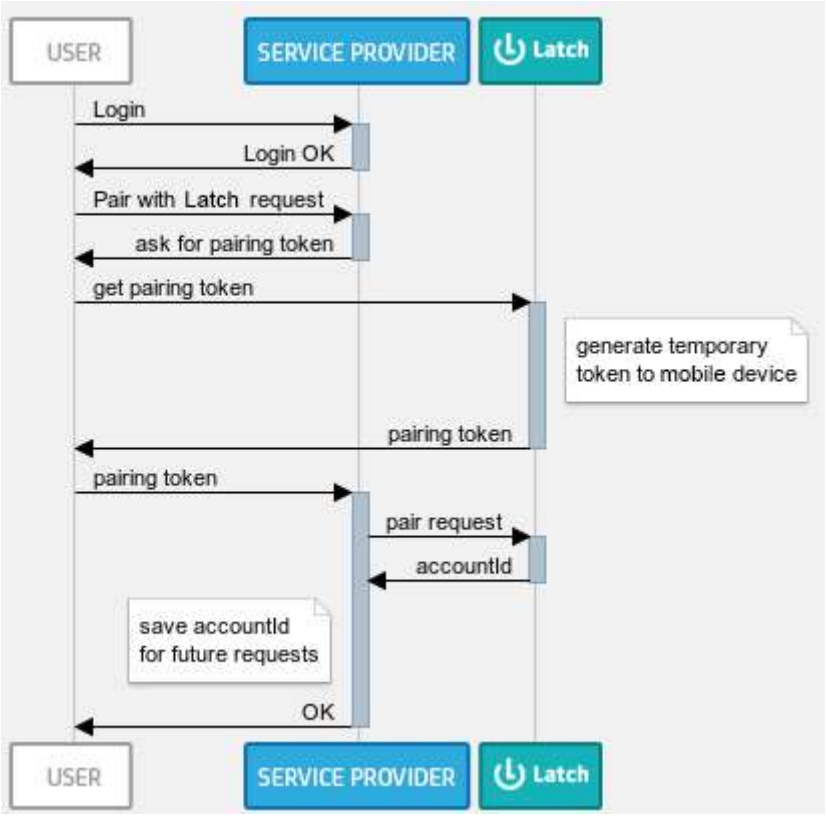
A seguir está uma lista de condições de erro possíveis que podem ocorrer durante a autenticação:

- **Erro 101:** Invalid Authorization header format
- **Erro 102:** Invalid application signatura
- **Erro 103:** Authorization header missing
- **Erro 104:** Date header missing
- **Erro 108:** Invalid date format
- **Erro 109:** Request expired, date is too old
- **Erro 111:** User not authorized
- **Erro 112:** Invalid user signatura
- **Erro 113:** Secret signing this request is not authorized to perform this operation

1.2. API de Aplicação

1.2.1. Pareamento de uma conta

Esta página descreve o método preferencial para parear contas usando um token de pareamento que os usuários podem obter pelos aplicativos móveis.



Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Para parear com a API de token, faça uma Solicitação HTTP GET ao seguinte endpoint:

```
https://latch.tu.com/api/2.0/pair/{token}
```

onde token é um token alfanumérico de seis caracteres de comprimento, que deve ser obtido do usuário, que por sua vez o obteve pelo aplicativo móvel.

Adicionalmente, e de forma totalmente opcional, você pode incluir um parâmetro commonName onde insira um identificador do usuário no seu sistema. Esse identificador será exibido na ferramenta de suporte (LST) e o ajudará a identificar o usuário de uma forma mais simples.

```
https://latch.tu.com/api/2.0/pair/{token}?commonName=YOUR_COMMON_NAME
```

Resposta

Se tudo correr bem, a API retornará uma resposta como esta:

```
{"data":{"accountId":"..."}}
```

onde accountId é um token alfanumérico de 64 caracteres de comprimento que pode ser usado para identificar de forma única a conta que está sendo pareada com este aplicativo. Esse valor deve ser armazenado para uso posterior em chamadas subsequentes, como **status** ou **anular pareamento**.

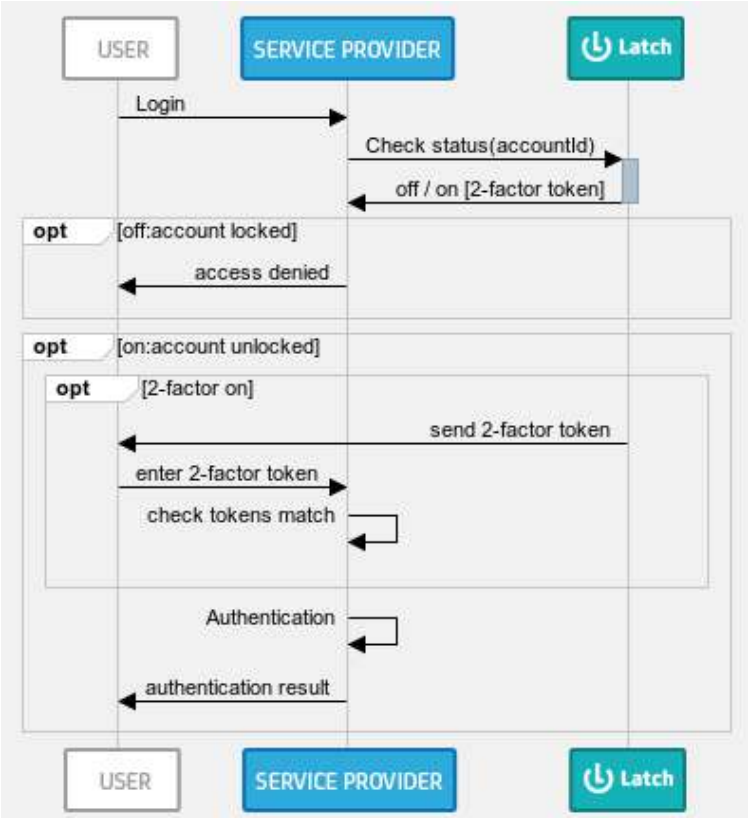
Erros

A seguir está uma lista de possíveis condições de erro que podem ocorrer ao tentar usar esta API:

- **Error 205:** Account and application already paired
  Este erro será retornado quando um usuário cuja conta já esteja pareada com este aplicativo tiver gerado o token especificado.
- **Error 206:** Pairing token not found or expired
  A causa mais provável deste Error é o uso de um token expirado. Os tokens devem ser usados para realizar o pareamento até 60 segundos após sua criação; depois desse período, expiram. Outra razão óbvia para este Error é que a Solicitação inclua um token inventado.
- **Error 401:** Missing parameter in API call
  A causa deste Error é que algum parâmetro obrigatório está faltando ou está vazio.
- **Error 406:** Invalid parameter length
  Este Error será retornado quando o parâmetro commonName tiver um comprimento maior que 100 caracteres.

1.2.2. Status de uma conta

Esta página descreve como verificar o status de uma conta e as diferentes respostas possíveis.



Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Para usar a API de status, faça uma solicitação HTTP GET ao seguinte endpoint:

```
https://latch.tu.com/api/2.0/status/{accountId}
```

onde accountId é um token alfanumérico de 64 caracteres de comprimento, que pode ser usado para identificar de forma única a conta cujo status está sendo consultado, obtido pelo método de pareamento.

Para obter informações relativas a uma operação específica, contrária a todas as operações definidas pelo seu aplicativo:

```
https://latch.tu.com/api/2.0/status/{accountId}/op/{operationId}
```

IMPORTANTE: Se você criou operações para um aplicativo, NÃO deve precisar consultar o status usando o ID do aplicativo. O mesmo acontece se você tiver operações aninhadas. Assim que uma operação tiver operações filhas, o ID do aplicativo da operação pai não deve ser usado para chamadas de status.

Você pode fazer isto, e irá funcionar. No entanto, nos aplicativos móveis, os aplicativos que tiverem operações ou as operações com filhos não terão as opções de OTP, opção de Programar ou Bloqueio automático. Em vez disso, eles terão a função de latch mestre sobre todas as operações filhas, permitindo que sejam bloqueadas ou desbloqueadas na mesma ação.

Opcionalmente, se você quiser consultar o status:

- Eliminando qualquer senha descartável (OTP) na resposta, é possível fazê-lo adicionando o sufixo /nootp a qualquer um dos endpoints.
- Sem enviar notificações push para os aplicativos móveis, é possível fazê-lo adicionando o sufixo /silent a qualquer um dos endpoints.

Exemplos:

```
https://latch.tu.com/api/2.0/status/{accountId}/nootp

https://latch.tu.com/api/2.0/status/{accountId}/silent

https://latch.tu.com/api/2.0/status/{accountId}/nootp/silent

https://latch.tu.com/api/2.0/status/{accountId}/op/{operationId}
/nootp/silent
```

Resposta

Se tudo correr bem, a API retornará uma resposta JSON que se parece com isto:

```
{"data":{"operations":{"applicationId":{"status":"...",
"two_factor":{...}, "operations":{...}}}}}
```

Existem dois valores possíveis para o campo de status: on e off.

O campo two_factor é opcional e, quando presente, tem o formato:

```
"two_factor":{"token":"...","generated":...}
```

onde token é um valor alfanumérico de seis caracteres de comprimento que também será enviado para o dispositivo móvel do usuário e deve ser verificado para concluir a operação. O campo generated é a marca de tempo em que o token foi gerado, e pode ser usado para controlar a janela de tempo na qual o token pode ser validado.

O objeto operations, se presente, contém as operações filhas e é definido recursivamente pelos mesmos campos: status, two_factor e operations.

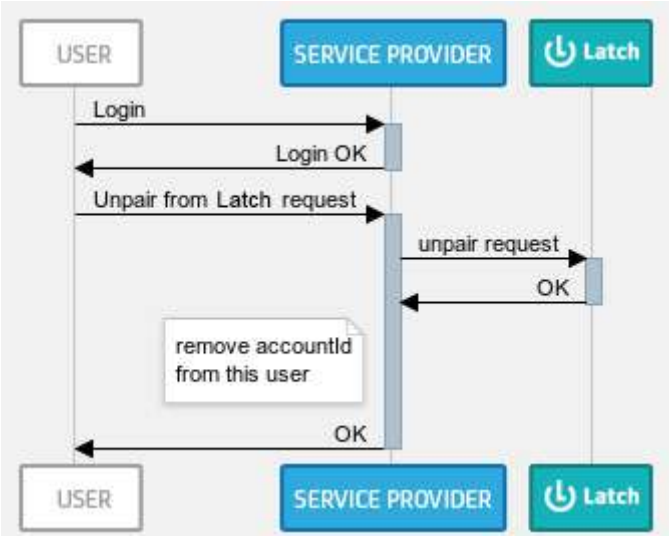
Erros

A seguir está uma lista de possíveis condições de Error que podem ocorrer ao tentar usar esta API:

- **Error 201:** Account not paired
  Este Error ocorre quando o identificador da conta está incorreto ou não está pareado atualmente com o aplicativo que solicita a verificação de status.
- **Error 401:** Missing parameter in API call
- **Errorr 704:** Silent notification is not applied due to subscription limits

1.2.3. Despareamento de uma conta

Esta página descreve como desparear uma conta do Latch.



Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Para desparear, faça uma solicitação GET de HTTP no seguinte ponto de extremidade:

```
https://latch.tu.com/api/2.0/unpair/{accountId}
```

onde accountId é um token alfanumérico de 64 caracteres de comprimento, que pode ser usado para identificar de forma única a conta cujo pareamento está sendo anulado, obtido pelo método de pareamento.

Resposta

Se tudo correr bem, a API retornará uma resposta JSON vazia, que se parece com isto:

```
{}
```

Erros

A listagem a seguir descreve os estados de erros possíveis que podem surgir ao tentar usar esta API:

- **Error 201:** Account not paired (API > 0.6)
- **Error 204:** Error unpairing account (API <= 0.6)
  Este Erro ocorre quando o identificador da conta está incorreto ou não está pareado atualmente com o aplicativo que solicita a anulação de pareamento.
- **Error 401:** Missing parameter in API call

1.2.4. Modificar Status

Esta página descreve como modificar o status de uma conta do Latch.

Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Para utilizar a API de modificações externas, realiza-se um pedido HTTP POST nos seguintes extremos, dependendo se trata-se de bloquear (lock) ou desbloquear (unlock) o estado de um Latch para um aplicativo ou operação:

```
https://latch.tu.com/api/2.0/lock/{accountId}
https://latch.tu.com/api/2.0/unlock/{accountId}
```

onde accountId é um token alfanumérico de 64 caracteres de comprimento que pode ser usado para identificar de forma única a conta cujo status está sendo modificado, obtido pelo método de pareamento.

Se você criou uma ou mais operações para serem controladas por Latches individuais, pode modificar o status de cada operação usando seu identificador de operação:

```
https://latch.tu.com/api/2.0/lock/{accountId}/op/{operationId}
https://latch.tu.com/api/2.0/unlock/{accountId}/op/{operationId}
```

Resposta

Se tudo correr bem, a API retornará uma resposta JSON vazia, que se parece com isto:

```
{}
```

Erros

A seguir está uma lista de possíveis condições de Error que podem ocorrer ao tentar usar esta API:

- **Error 111:** User not authorized
  Este erro é produzido quando o atual plano de assinatura do usuário não inclui as ferramentas de suporte.

- **Error 201:** Account not paired
  Este erro ocorre quando o identificador da conta é incorreto ou não está pareado atualmente ao aplicativo que solicitou a verificação do status.

- **Error 401:** Missing parameter in API call

1.2.5. Gerenciar Aplicações

Nesta página é descrito como gerenciar as operações de uma aplicação, permitindo adicionar, atualizar ou excluir operações.

Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Pedidos

Adicionar uma operação

Para adicionar uma operação, realiza-se um pedido HTTP PUT no seguinte extremo:

`https://latch.tu.com/api/2.0/operation`

Parâmetros da pedido:

- parentId: Identificador do aplicativo ou operação à qual desejamos adicionar a nova operação a criar.
- name: Nome para a nova operação.
- two_factor (Opcional): Valor para o segundo fator de autenticação. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.
- lock_on_request (Opcional): Valor para o bloqueio após consulta. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.

Resposta

Se tudo correr bem, a API retornará uma resposta JSON que se parece com isto, sendo operationId o identificador da operação recém-criada: {data:{"operationId":"..."}}

Modificar uma operação

Para modificar uma operação, realiza-se um pedido HTTP POST no seguinte extremo:

`https://latch.tu.com/api/2.0/operation/{operationId}`

onde operationId é o identificador dessa operação.

Parâmetros da pedido:

- name: Novo nome para a operação.

- two_factor (Opcional): Valor para o segundo fator de autenticação. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.

- lock_on_request (Opcional): Valor para o bloqueio após consulta. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.

Resposta

Se tudo sair bem, será recebida uma resposta json vazia da API, que se parece a esta:

```
{}
```

Eliminar uma operação

Para excluir uma operação realiza-se um pedido HTTP DELETE ao seguinte endpoint:
https://latch.tu.com/api/2.0/operation/{operationId}

onde operationId é o identificador dessa operação.

Resposta

Se tudo correr bem, a API retornará uma resposta JSON vazia, que se parece com isto:

```
{}
```

Retornar operações

Para ver suas operações, faça uma Solicitação HTTP GET ao seguinte endpoint:

```
https://latch.tu.com/api/2.0/operation
https://latch.tu.com/api/2.0/operation/{operationId}
```

Resposta

Se tudo correr bem, a API retornará uma resposta como esta:

```
{"data":{"operations":{"operationId":{"name":"...","two_factor":
"...","lock_on_request":"...","operations":{...}}}}}
```

Erros

A seguir está uma lista de possíveis condições de Erros que podem ocorrer ao tentar usar esta API:

- **Error 301:** Application or Operation not found
  Este Error ocorre quando o identificador de operação utilizado não corresponde a um identificador de operação válido.
- **Error 401:** Missing parameter in API call
- **Error 703:** Application or Operation not created due to subscription limits
  Este Error ocorre quando se tenta criar uma operação fora dos limites de assinatura.

1.2.6. Histórico do Usuário

Nesta página é descrito como consultar o histórico de uma conta.

Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Para utilizar a API de histórico, realiza-se um pedido HTTP GET no seguinte extremo:

```
https://latch.tu.com/api/2.0/history/{accountId}/{from}/{to}
```

onde accountId é um token alfanumérico de 64 caracteres de comprimento que pode ser usado para identificar de forma única a conta cujo histórico está sendo consultado, obtido pelo método de pareamento.

Os parâmetros from e to, que são opcionais, permitem filtrar o histórico entre duas datas. Ambos os parâmetros são numéricos e correspondem à quantidade de milissegundos decorridos desde 1º de janeiro de 1970, 00:00:00 GMT (Epoch Time).

Resposta

Se tudo correr bem, a API retornará uma resposta JSON que se parece com isto:

```
{ "data": { "applicationId": { ... }, "count": ...,
"clientVersion": { ... } }, "lastSeen": ..., "history": [ { "t":
..., "action": "...", "what": "...", "value": "...", "was":
"...", "name": "...", "userAgent": "...", "ip": "..." } ] } }
```

onde as informações contidas são:

- **applicationId**: informações relativas ao aplicativo e suas operações.
- **count**: quantidade de elementos contidos no array history.
- **clientVersion**: informações relativas aos clientes móveis utilizados pelo usuário.
- **lastSeen**: horário em que foi registrada a última atividade do usuário.
- **history**: array que contém as ações realizadas na conta, seja pelo usuário ou pelo desenvolvedor.
    - **t**: tempo em que a ação foi realizada.
    - **action**: se foi uma ação realizada pelo usuário (USER_UPDATE), pelo desenvolvedor (DEVELOPER_UPDATE) ou uma consulta de status (get).
    - **what**: o parâmetro sobre o qual a ação foi realizada (status, two_factor, from etc.).
    - **was**: o valor anterior da ação se foi uma modificação.
    - **value**: o valor atual da ação.

- **name**: nome do aplicativo ou operação sobre a qual a ação foi realizada.
- **ip / userAgent**: IP e User-Agent de quem realizou a ação.

Erros

A seguir está uma lista de possíveis condições de erros que podem ocorrer ao tentar usar esta API:

- **Error 111: User not authorized**
  Este erro ocorre quando o plano de assinatura atual do usuário não inclui ferramentas de suporte.
- **Error 201: Account not paired**
  Este erro ocorre quando o identificador de conta está incorreto ou não está pareado atualmente com o aplicativo que solicita a verificação de status.
- **Error 401: Missing parameter in API call**
- **Error 405: History response is limited to 1000 entries for the selected date range**
  Este erro não fatal ocorre quando a resposta deveria conter mais elementos do que o máximo (1000) que são retornados.

1.2.7. Administrar instâncias

Nesta página é descrito como gerenciar as instâncias de uma aplicação ou operação associadas a um accountId específico, bem como a forma de interagir com elas.

Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Pedidos

Adicionar uma instância

Para adicionar uma instância a uma aplicação, realiza-se um pedido HTTP PUT ao seguinte extremo:

```
https://latch.tu.com/api/2.0/instance/{accountId}
```

Para adicionar instâncias a uma operação, realiza-se um pedido HTTP PUT no seguinte extremo:

```
https://latch.tu.com/api/2.0/instance/{accountId}/op/{operationId}
```

onde accountId é um token alfanumérico de 64 caracteres de comprimento, que pode ser usado para identificar de forma única a conta cujas instâncias estão sendo modificadas. Ele terá sido obtido anteriormente pelo método de pareamento.

Parâmetros da pedido:

- instances: O nome das instâncias a serem criadas no formato instances=nome, podendo criar várias simultaneamente separadas por &.

Exemplo: instances=Name_1&instances=Name_2&instances=Name_N

Resposta

Se tudo correr bem, a API retornará uma resposta JSON que se parece com isto, sendo os vários instancesId os identificadores das instâncias recém-criadas e que devem ser guardados para interagir com elas:

```
{"data":{"instances":{"instanceId_1":"Name_1","instanceId_2":"Name_2","instanceId_N":"Name_N", ...}}}
```

Eliminar uma instância

Para eliminar uma instância de uma aplicação, realiza-se um pedido HTTP DELETE ao seguinte extremo:

```
https://latch.tu.com/api/2.0/instance/{accountId}/i/{instanceId}
```

Para eliminar uma instância de uma operação, realiza-se um pedido HTTP DELETE ao seguinte extremo:

```
https://latch.tu.com/api/2.0/instance/{accountId}/op/{operationId}/i/{instanceId}
```

onde instanceId é o identificador dessa instância e operationId é o identificador da operação à qual ela pertence.

Resposta

Se tudo correr bem, a API retornará uma resposta JSON vazia, que se parece com isto:

```
{}
```

Retornar instâncias

Para retonar as instâncias de uma aplicação, realiza-se um pedido HTTP GET ao seguinte extermo:

```
https://latch.tu.com/api/2.0/instance/{accountId}
```

Para obter as instâncias de uma operação, realiza-se um pedido HTTP GET ao seguinte endpoint:

```
https://latch.tu.com/api/2.0/instance/{accountId}/op/{operationId}
```

onde instanceId é o identificador dessa instância e operationId é o identificador da operação à qual ela pertence.

Resposta

Se tudo correr bem, a API retornará uma resposta como esta:

```
{"data": {"instanceId_1": {"name": "MyInstance1", "two_factor":
"MANDATORY|DISABLED|OPT_IN", "lock_on_request":
"MANDATORY|DISABLED|OPT_IN"}, "instanceId_2": {"name": "...",
"two_factor": "...", "lock_on_request": "..."}}}
```

Editar uma instância

Para editar uma instância de uma aplicação, realiza-se um pedido HTTP POST ao seguinte extremo:

```
https://latch.tu.com/api/2.0/instance/{accountId}/i/{instanceId}
```

Para editar uma instância de uma operação, realiza-se pedido HTTP POST ao seguinte extremo:

```
https://latch.tu.com/api/2.0/instance/{accountId}/op/{operationI
d}/i/{instanceId}
```

onde instanceId é o identificador dessa instância e operationId é o identificador da operação à qual ela pertence.

Parâmetros da pedido:

- name (Opcional): Novo nome para a instância.
- two_factor (Opcional): Valor para o segundo fator de autenticação. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.
- lock_on_request (Opcional): Valor para bloqueio após consulta. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.

Resposta

Se tudo correr bem, a API retornará uma resposta JSON vazia, que se parece com isto:

```
{}
```

Verificar o estado de uma instância

Para consultar o status de uma instancia, realiza-se uma solicitação HTTP GET em um dos seguintes endpoints (para instâncias de uma aplicação ou instâncias de uma operação):

```
https://latch.tu.com/api/2.0/status/{accountId}/i/{instanceId}
```

```
https://latch.tu.com/api/2.0/status/{accountId}/op/{operationId}/i/{instanceId}
```

Tanto a resposta quanto as demais opções adicionais como silent ou nootp estão detalhadas na seção Status da Conta.

Modificar o estado de uma instância

Para modificar o estado de uma instância, realiza-se uma solicitação HTTP POST em um dos seguintes extremos (para instâncias de um aplicativo ou instâncias de uma operação):

```
https://latch.tu.com/api/2.0/lock/{accountId}/i/{instanceId}
```

```
https://latch.tu.com/api/2.0/lock/{accountId}/op/{operationId}/i/{instanceId}
```

```
https://latch.tu.com/api/2.0/unlock/{accountId}/i/{instanceId}
```

```
https://latch.tu.com/api/2.0/unlock/{accountId}/op/{operationId}/i/{instanceId}
```

Tanto a resposta quanto as informações adicionais podem ser consultadas na seção Modificar estado.

Erros

A seguir está uma lista de possíveis condições de Error que podem ocorrer ao tentar usar esta API:

- **Error 301: Application or Operation not found**
  Este erro ocorre quando o identificador de operação utilizado não corresponde a um identificador de operação válido.

- **Error 302: Instance not found**
  Este erro ocorre quando o identificador de instância utilizado não corresponde a um identificador de instância válido.

- **Error 401: Missing parameter in API call**

- **Error 703: Application or Operation not created due to subscription limits**
  Este erro ocorre quando se tenta criar uma instância fora dos limites de assinatura.

1.2.8. Editar CommonName

Esta página descreve como editar um nome comum para um usuário emparelhado.

Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Definir um nome comum

Para definir um nome comum, execute uma solicitação HTTP POST ao seguinte endpoint:

```
https://latch.tu.com/api/3.0/commonName/{accountId}/
```

Parâmetros da pedido:

- commonName: Identificador do provedor de serviço para o usuário pareado

Resposta

Se tudo sair bem, será recebida uma resposta json vazia da API, que se parece a esta:

```
{}
```

Erros

A seguir está uma lista de possíveis condições de erroque podem ocorrer ao tentar usar esta API:

- **Error 201: Account not paired**
  Este erro ocorre quando o identificador da conta está incorreto ou não está pareado atualmente com o aplicativo que solicita a verificação de status.

- **Error 401: Missing parameter in API call**
  A causa deste erro é que algum parâmetro obrigatório está faltando ou está vazio.

- **Error 406: Invalid parameter length**
  Este erro será retornado quando o parâmetro commonName tiver um comprimento maior que 100 caracteres.

1.3. API de Usuário

1.3.1. Gerenciar aplicativos

Esta página descreve como gerenciar os aplicativos de um desenvolvedor, permitindo-lhe adicionar, atualizar ou remover aplicativos.

Autenticação

Para usar esta solicitação, você deve autenticar seu usuário utilizando sua userId e fazer o logon como explicado na seção de autenticação.

Pedidos

Adicionar uma aplicativo

Para adicionar um aplicativo, execute uma solicitação HTTP PUT no seguinte caso:

```
https://latch.tu.com/api/2.0/application
```

Parâmetros da solicitação:

- name: O novo nome para a aplicativo.
- contactEmail: Email de contato para a aplicativo
- contactPhone: Telefone de contato para a aplicativo.
- two_factor (Opcional): Valor para o segundo fator de autenticação. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.
- lock_on_request (Opcional): Valor para bloqueio após consulta. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.

Resposta

Se tudo correr bem, a API retornará uma resposta JSON que se parece com isto, sendo applicationId o identificador da aplicativo recém-criada:

```
{data:{"applicationId":"...", "secret":"..."}}
```

Modificar um aplicativo

Para modificar um aplicativo, execute uma solicitação HTTP POST no seguinte caso:

```
https://latch.tu.com/api/2.0/application/{applicationId}
```

'applicationId' será o identificador desse aplicativo.

Parâmetros do pedido:

- name: O novo nome para a aplicativo.
- contactEmail: Email de contato para a aplicativo.
- contactPhone: Telefone de contato para a aplicativo.
- two_factor (Opcional): Valor para o segundo fator de autenticação. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.
- lock_on_request (Opcional): Valor para bloqueio após consulta. Valores possíveis: MANDATORY, OPT_IN, DISABLED. Valor padrão é DISABLED.

Resposta

Se tudo correr bem, a API retornará uma resposta JSON vazia, que se parece com isto:

```
{}
```

Remover um aplicativo

Para remover um aplicativo, execute uma solicitação HTTP DELETE no seguinte caso:

```
https://latch.tu.com/api/2.0/application/{applicationId}
```

onde applicationId é o identificador desse aplicativo.

Resposta

Se tudo correr bem, a API retornará uma resposta JSON vazia, que se parece com isto:

```
{}
```

Obter aplicativo

Para obter um aplicativo existente, execute uma solicitação HTTP GET no seguinte caso:

```
https://latch.tu.com/api/2.0/application
```

Resposta

Se tudo correr bem, a API retornará uma resposta como esta:

```
{"data":{"operations":{"applicationId":{"name":"...","two_factor
":"...","lock_on_request":"...","operations":{...}}}}}
```

Erros

A seguir está uma lista de possíveis condições de Error que podem ocorrer ao tentar usar esta API:

- **Error 111: User not authorized**
  Este error ocorre quando o plano de assinatura atual do usuário não inclui a API de usuário.
- **Error 301: Application or Operation not found**
  Este erro ocorre quando o identificador de aplicativo usado não corresponde a um identificador de aplicativo válido.
- **Error 401: Missing parameter in API call**
- **Error 402: Invalid parameter value**
  Este Error ocorre quando alguns parâmetros têm formato incorreto, como endereços de email, números de telefone ou nomes.
- **Error 703: Application or Operation not created due to subscription limits**
  Este Error ocorre quando se tenta criar uma aplicação fora dos limites de assinatura.

Esta página descreve como obter informações da assinatura do usuário.

IMPORTANTE: A API aqui descrita, somente estará disponível para aqueles usuários cujo plano de assinatura permita funções de suporte via API. (Assinaturas do tipo **GOLD** ou **PLATINUM**).

1.3.2. Subscrição desenvolvedor

Autenticação

Para usar esta solicitação, você deve autenticar seu usuário utilizando sua userId e fazer o logon como explicado na seção de autenticação.

Pedidos

Obter Assinatura

Para obter informações sobre a assinatura, execute uma solicitação HTTP GET no seguinte caso:

```
https://latch.tu.com/api/2.0/subscription
```

Resposta

Se tudo sair bem, será recebida uma resposta json da API, que se parece a esta:

```
{"data":{"subscription":{"id":"subcriptionName","applications":{
"inUse":X,"limit":X},"operations":{"appName1":{"inUse":X,"limit"
:X},"users":{"inUse":X,"limit":X}}}}
```

onde a informação contida é:

- **subscriptionName**: O nome da sua assinatura atual.
- **inUse**: Quantidade de aplicativos, usuários ou operações em utilização no momento.

- **limit**: Quantidade máxima permitida para a sua assinatura atual. -1 equivale a Ilimitado.

Erros

A listagem a seguir descreve os estados de erros possíveis que podem surgir ao tentar usar esta API:

Error 111: User not authorized
Este erro ocorre quando o plano de assinatura atual do usuário não inclui os recursos da API do Usuário.

1.4. Servidor de TOTP

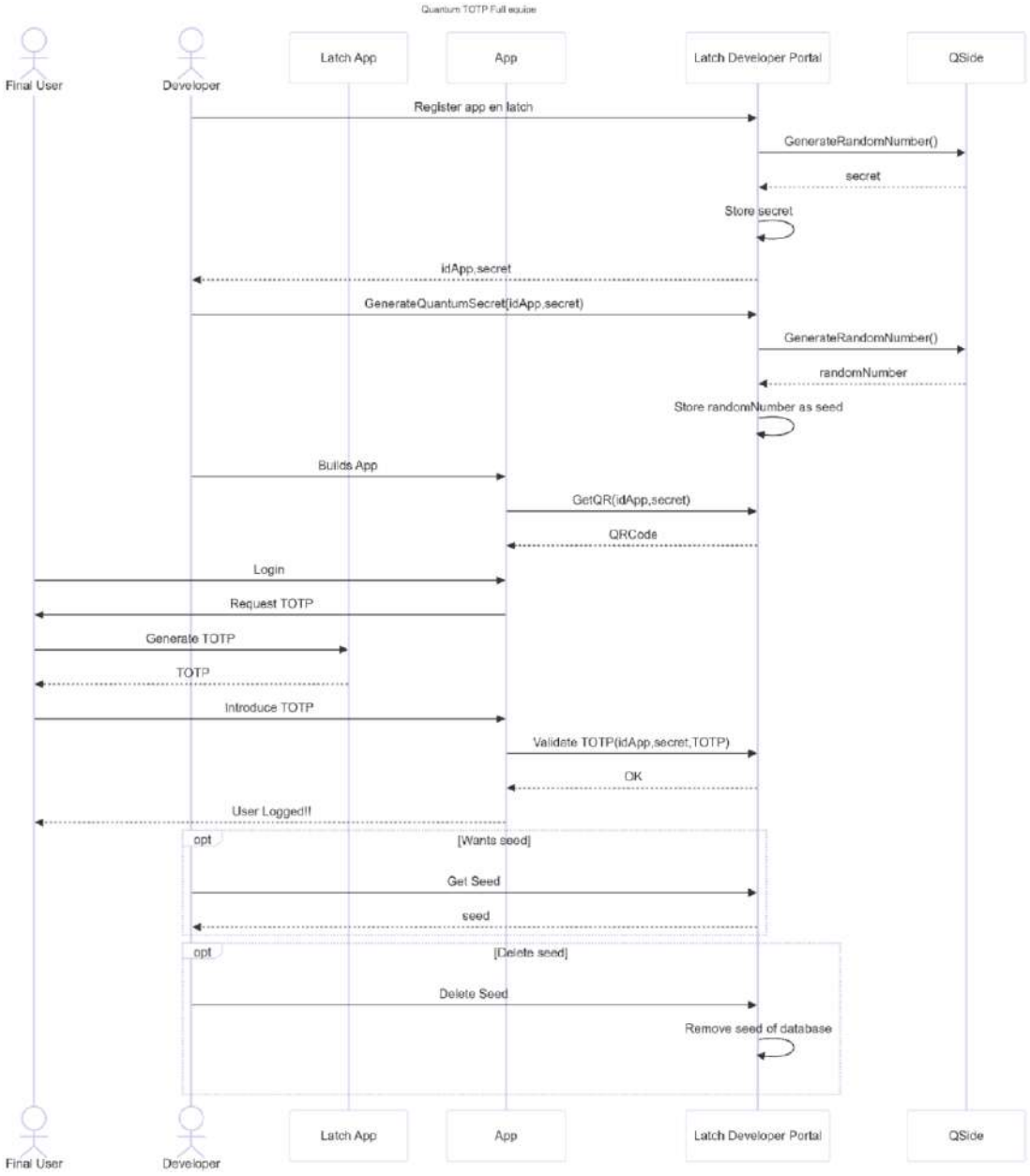
1.4.1. Criar TOTP

Este serviço funciona como um servidor TOTP (Time-based One-Time Password) que permite criar, gerenciar e validar senhas descartáveis baseadas em tempo.

Ele fornece uma API para criar e gerenciar esses TOTPs e segredos associados a diferentes usuários. As funcionalidades principais incluem:

- **Criar um TOTP**: Permite que os usuários criem um TOTP para sua autenticação.
- **Excluir um TOTP**: Os TOTPs criados podem ser excluídos quando não forem mais necessários.
- **Obter informações de um TOTP**: Você pode consultar as informações de um TOTP específico.
- **Validar um TOTP**: Permite validar um código TOTP fornecido por um usuário.

A sequência das operações e seu uso é representada no diagrama a seguir:



Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Cria um novo TOTP para um determinado usuário. É necessário userId (identificador do usuário) e commonName (nome comum do TOTP).

```
POST https://latch.tu.com/api/3.0/totps
```

Parâmetros:

- userId: Identificador que será associado ao usuário (dependente do provedor).
- commonName: Nome a ser associado ao usuário (dependente do provedor).

Resposta

Se bem-sucedido, a API responderá com as seguintes informações do TOTP:

```
{
    "data": {
        "totpId": "identificador del TOTP",
        "secret": "secreto del TOTP",
        "appId": "identificador de la app de Latch",
        "identity": {
            "id": "identificador del usuario",
            "name": "nombre del usuario"
        },
        "issuer": "nombre de la app",
        "algorithm": "algoritmo del TOTP",
        "digits": "dígitos del TOTP",
        "period": "período del TOTP",
        "createdAt": "fecha de creación",
        "qr": "imagen del código QR en base64",
        "uri": "URI del TOTP, formato otpauth"
    }
}
```

Erros

A seguir está uma lista de possíveis condições de Error que podem ocorrer ao tentar usar esta API:

- **Error 301:** Application or Operation not found
  Este erro ocorre quando o identificador do aplicativo utilizado não corresponde a um identificador válido.
- **Error 304:** No totp server configured
  Não existe um TOTP Server configurado para este aplicativo.
- **Error 401:** Missing parameter in API call
  Parâmetros inválidos.
- **Error 307:** Unable to generate totp
  Não foi possível gerar o segredo para o TOTP. Erro interno.

1.4.2. Excluir TOTP

Autenticação

Para usar esta solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Exclua um TOTP específico usando seu identificador totpId.

```
DELETE https://latch.tu.com/api/3.0/totps/{totpId}
```

Parâmetros:

- totpId: Identificador do TOTP que se deseja excluir.

Resposta

Se tudo correr bem, será retornado um HTTP RESPONSE 204.

Erros

A seguir está uma lista de possíveis condições de Error que podem ocorrer ao tentar usar esta API:

- **Error 304:** No totp server configured
  Não existe um TOTP Server configurado para este aplicativo.
- **Error 305:** App totp not found
  TOTP não encontrado.

1.4.3. Obter TOTP

Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Retorna as informações de um TOTP específico com base em seu `totpId`.

```
GET https://latch.tu.com/api/3.0/totps/{totpId}
```

Parâmetros:

- totpId: Identificador do TOTP que se deseja obter.

Resposta

Se tudo correr bem, a API responderá com as informações relativas ao TOTP:

```
{
  "data": {
```

```
    "totpId": "identificador del TOTP",
    "secret": "secreto del TOTP",
    "appId": "identificador de la app de Latch",
    "identity": {
      "id": "identificador del usuario",
      "name": "nombre del usuario"
    },
    "issuer": "nombre de la app",
    "algorithm": "algoritmo del TOTP",
    "digits": "dígitos del TOTP",
    "period": "período del TOTP",
    "createdAt": "fecha de creación",
    "qr": "imagen del código QR en base64",
    "uri": "URI del TOTP, formato otpauth"
  }
}
```


Erros

A seguir está uma lista de possíveis condições de Error que podem ocorrer ao tentar usar esta API:

- **Error 304:** No totp server configured
  Não existe um TOTP Server configurado para este aplicativo.
- **Error 305:** App totp not found
  TOTP não encontrado.

1.4.4. Validar TOTP

Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Valida um código TOTP específico fornecido pelo usuário. O totpId identifica o TOTP, e code é o código gerado que deve ser validado.

```
POST https://latch.tu.com/api/3.0/totps/{totpId}/validate
```

Parâmetros:

- totpId: Identificador do TOTP que será usado para validar o algoritmo contra o código enviado.
- code: Código enviado pelo usuário final, para validar segundo o algoritmo especificado pelo totpId e o timestamp atual.

Resposta

Em ambos os casos, será retornado um código HTTP 200, conforme:

Código correto:

```
{}
```

Código incorreto:

```
{"error": {
 "code": 306,
 "message": "Invalid totp code"
   }
}
```

Erros

A seguir está uma lista de possíveis condições de erro que podem ocorrer ao tentar usar esta API:

- **Error 304:** No totp server configured
  Não existe um TOTP Server configurado para este aplicativo.
- **Error 305:** App totp not found
  TOTP não encontrado.
- **Error 306:** Invalid totp code
  Código inválido de acordo com os parâmetros armazenados e o carimbo de data/hora atual.
- **Error 307:** Unable to generate totp
  Errorao obter os parâmetros de geração de TOTPs.
- **Error 401:** Missing parameter in API call
  Algum dos parâmetros de entrada não existe.
- **Error 402:** Invalid parameter value
  Parâmetros inválidos.
- **Error 708:** TOTP not validated due to subscription limits
  TOTP não é válido devido aos limites de assinatura do desenvolvedor.

1.5. Controles de Autorização

Introdução

A funcionalidade de Controles de Autorização no produto Latch permite estabelecer um mecanismo para gerenciar a aprovação de ações dentro de um sistema, com base na intervenção de um ou mais usuários. Esta funcionalidade é útil em cenários onde é necessário obter a aprovação explícita de certos usuários antes de permitir o avanço de um processo ou fluxo.

Criação de um Controle de Autorização

A criação de controles de autorização é feita através de uma ferramenta chamada LST. Nela, administradores ou usuários com permissões podem definir uma nova regra ou controle que gerencie autorizações para um grupo específico de usuários.

Passos para criar um controle de autorização

1. Acessar a ferramenta LST.
2. Selecionar a opção para criar um novo Controle de Autorização.
3. Especificar os usuários envolvidos no controle.
4. Definir se o controle deve se basear em:
    - **Aprovação Unânime (TODOS)**: O fluxo ou ação só será permitido se todos os usuários selecionados aprovarem a solicitação.
    - **Aprovação Parcial (ALGUM)**: O fluxo ou ação será permitido se ao menos um dos usuários selecionados aprovar a solicitação.
5. Salvar o controle de autorização.

Identificação do Controle de Autorização

Cada controle de autorização criado possui um identificador único chamado controlID. Esse controlID é fundamental para interagir com o recurso por meio da API do sistema, permitindo sua consulta ou modificação posterior.

Uso da API para os Controles de Autorização

Depois que um controle de autorização é criado, ele pode ser gerenciado ou consultado pelas APIs do Latch. O controlID será o principal identificador usado para executar qualquer operação sobre esse controle.

Operações disponíveis via API:

- **Consultar um controle de autorização**: Permite recuperar os detalhes do controle, incluindo os usuários associados e as regras de aprovação (TODOS ou ALGUNO).

- **Atualizar um controle de autorização**: Modificar os usuários ou as condições de aprovação.
- **Excluir um controle de autorização**: Remover um controle de autorização se não for mais necessário.

Exemplo de Cenário de Uso

Suponha que em um fluxo de negócios seja necessária a aprovação de três usuários antes de concluir uma transação. É possível configurar um controle de autorização no Latch onde:

- Os três usuários envolvidos são adicionados ao controle.
- Define-se que a aprovação deve ser unânime, ou seja, **todos** os usuários devem aprovar antes que a transação seja executada.

Em outro caso, se for necessário que apenas um dos três usuários possa aprovar para prosseguir, configuraria-se o controle com a opção ALGUM.

Conclusão

A funcionalidade de Controles de Autorização no Latch oferece uma maneira flexível e poderosa de gerenciar aprovações de usuários, permitindo configurações baseadas tanto na participação de todos os usuários envolvidos quanto na aprovação por parte de um subconjunto. A integração com a LST e a disponibilidade via API facilitam seu uso tanto em nível de interface gráfica quanto de automação em aplicativos externos.

Autenticação

Para usar esta Solicitação, você deve autenticar seu aplicativo usando seu applicationId e assiná-lo conforme descrito na seção de Autenticação.

Solicitação

Consulte o status de um controle de autorização específico.

```
GET https://latch.tu.com/api/3.0/control-status/{controlId}
```

Parâmetros:

- controlId: Identificador do controle de autorização a ser consultado.

Resposta

Se tudo correr bem, a API responderá com as informações relativas ao status do controle de autorização controlId:

```
{
    "data": {
        "status": "on" | "off"
    }
}
```

Erros

A seguir está uma lista de possíveis condições de erro que podem ocorrer ao tentar usar esta API:

- **Error 113:** Secret signing this request is not authorized to perform this operation
  A assinatura appId/secret não está autorizada a realizar esta operação.

- **Error 301:** Application or Operation not found
  Este erro ocorre quando o identificador de operação utilizado não corresponde a um identificador de operação válido.

- **Error 302:** Instance not found
  Este erro ocorre quando o identificador de instância utilizado não corresponde a um identificador de instância válido.

- **Error 1100:** Authorization control not found
  O identificador do controle de autorização não foi encontrado.

- **Error 1104:** Error checking authorization control status
  Erro ao consultar o controle de autorização.

2. Webhooks

Os WebHooks permitem que os desenvolvedores recebam notificações em tempo real quando seus usuários pareados realizarem ações através do app móvel.

Depois de registrar uma URI para receber WebHooks, Latch começará a enviar solicitações HTTP a essa URI cada vez que houver mudanças em algum usuário.

2.1. Adicionar um Webhook

Para adicionar um novo WebHook, acesse a página da sua aplicação em **'Minhas Aplicações'**.

Em seguida, adicione a URI completa do seu WebHook sem incluir parâmetros de consulta (ex.: https://www.example.com/hook) na seção **"WebHooks"**. Observe que essa URI precisa estar publicamente acessível na Internet; ou seja, URIs como 127.0.0.1 ou localhost não serão válidas, pois o Latch não poderá se comunicar com sua rede local.

Após introduzir uma URI para um WebHook, será enviada a essa URI uma solicitação de verificação. Esta verificação é uma solicitação HTTP GET que inclui um parâmetro 'challenge'. O seu WebHook deve responder um eco com tal parâmetro para poder ser verificado. Esta verificação é realizada para comprovar que o seu aplicativo realmente deseja receber notificações nessa URI. Se introduzir por engano uma URI incorreta (ou se alguém de forma maliciosa introduzir seu servidor como WebHook), o seu aplicativo não responderá corretamente ao desafio e o Latch não enviará nenhuma notificação a essa URI.

Exemplo:

```
GET https://www.example.com/hook?challenge=ABC123
```

Se seu WebHook responder corretamente ao desafio acima, o Latch começará a enviar notificações para a URI do seu WebHook com as alterações dos seus usuários (consulte a aba **Notificações de WebHooks** para mais informações). Caso contrário, você verá uma mensagem de Error indicando o problema encontrado.

NOTA: Sua URI tem apenas dez segundos para responder à notificação antes que ocorra um 'timeout'.

2.2. Notificações Webhook

Quando uma URI de WebHook é adicionada, você começa a receber **requisições de notificação** sempre que um dos seus usuários fizer alguma alteração no seu aplicativo.

Uma Solicitação de notificação é uma Solicitação HTTP POST que inclui um JSON em seu corpo. Observe que várias alterações simultâneas para vários usuários podem estar incluídas na mesma notificação. Você pode encontrar o formato JSON específico para cada tipo de notificação abaixo.

Assinatura das notificações

Cada Solicitação de notificação incluirá um cabeçalho HTTP X-11paths-Authorization formado pela assinatura HMAC-SHA1 do corpo da Solicitação (o **secret** do seu aplicativo é usado como chave de assinatura). Isso permitirá verificar se a notificação é proveniente do Latch. É recomendável verificar a validade da assinatura para cada notificação recebida.

Bloqueio / desbloqueio de um latch

Essa notificação é enviada sempre que um usuário modifica o latch da sua aplicação ou de qualquer uma de suas operações internas.

```
{
   "t":1457913600,
   "accounts":{
     "ACCOUNT_ID1":[
                {
                 "type":"UPDATE",
                 "id":"APP_OR_OPERATION_ID | GLOBAL_LATCH",
                 "source":"USER_UPDATE | DEVELOPER_UPDATE",
                 "new_status":"on | off | interval"
                },
                { ... }
     ],
     "ACCOUNT_ID2":[
       ...
     ]
   }
 }
```

Informações:

- **t**: O timestamp (UTC) quando o Latch envia esta notificação.
- **ACCOUNT_ID**: O accountId dos seus usuários, obtido durante a fase de pareamento.
- **id**: O identificador da aplicação ou operação do latch que o usuário acabou de modificar. Se o usuário modificar o 'latch mestre' de todos os aplicativos, o valor será GLOBAL_LATCH.
- **source**: Se a alteração foi feita pelo usuário (USER_UPDATE) ou pelo desenvolvedor usando a API de suporte ou a ferramenta LST (DEVELOPER_UPDATE).

- **new_status**: O novo estado do latch: on para desbloqueio, off para bloqueio, ou interval se o usuário tiver ativado a opção de "bloqueio programado" para esse latch.

3. Segredos Limitados

Cada aplicação do Latch fornece um **identificador de aplicação** (appId) e um **secret**. Esse par de credenciais permite que as requisições à API sejam assinadas para garantir que sejam feitas pelo dono legítimo da aplicação. Entretanto, existem diversos cenários em que esse segredo pode ser comprometido (aplicativos de desktop, aplicativos móveis, etc.).

Nesses cenários, um secret comprometido poderia permitir que um possível invasor realizasse operações indesejadas na API. Para evitar isso, é possível criar até **3 segredos limitados** para cada aplicação, cada um com escopo restrito. Esses segredos são usados da mesma forma que o secret principal, mas só podem ser utilizados para assinar as requisições para as quais foram configurados.

Um **segredo limitado** pode conter um ou mais dos seguintes permissões:

- **Status**: Permite realizar chamadas de status à API para conhecer o status de um Latch.
- **Pairings**: Permite usar as chamadas de parear e desparear um usuário.
- **Support**: Permite usar as chamadas de suporte da API para bloquear e desbloquear “latches”.
- **History**: Permite consultar o histórico de ações de um serviço.
- **Operations**: Concede acesso para realizar as chamadas de gerenciamento de operações. Esse gerenciamento possibilita criar, editar, consultar e excluir operações.
- **Instances**: Concede acesso para realizar as chamadas de gerenciamento de instâncias. Esse gerenciamento possibilita criar, editar, consultar e excluir instâncias. Para consultar o status de uma instância, usa-se a permissão STATUS, e para modificar seu status, a permissão SUPPORT.

- **Servidor de TOTPs**: Concede acesso para realizar as chamadas de gerenciamento dos TOTPs.
- **Nº segredos quânticos**: Concede acesso para realizar as chamadas de gerenciamento de segredos quânticos.
- **Controles de autorização**: Concede acesso para realizar as chamadas de gerenciamento de controles de autorização.

4. Ligação Directa

4.1. O que é Ligação Directa?

'Deep linking' ou Ligação Directa é uma técnica pela qual se permite iniciar aplicativos móveis nativos por meio de um link.

Ligação directa permite que você conecte um único url com uma ação particular de uma aplicação móvel; dependendo das plataformas móveis, Uris necessário para iniciar o aplicativo pode ser diferente.

Estas são as características de nossas aplicações podem agora ser feito com " profunda ligação ' ou link profundo. No momento, eles só estão disponíveis para Android e iOS.

4.2. Ligação directa no Latch

Você pode usar qualquer um desses recursos URI tanto de um navegador web a partir de uma aplicação móvel nativa:

Inicie o aplicativo Latch

Android

```
latch://launch
```

iOS

```
latch://launch
```