

## AWS Proton ワークショップ

2021/03/05

シニアエバンジェリスト  
亀田

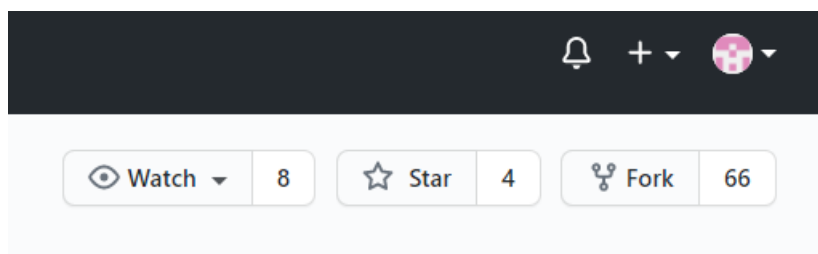
### 1. github アカウントの設定

1.1. ご利用のブラウザで github にログインを行いアクセスします

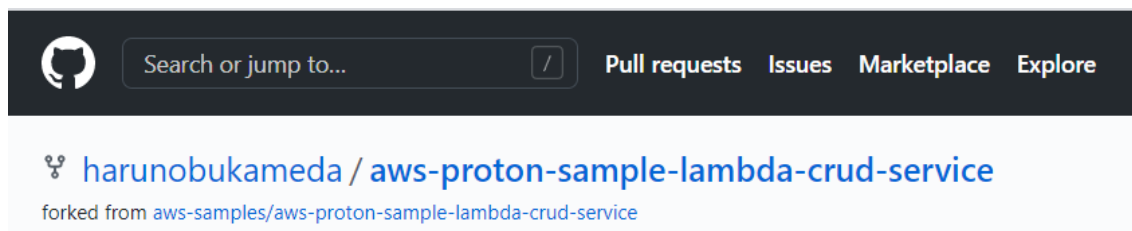
1.2. 別のタブで以下の URL にアクセスします

<https://github.com/aws-samples/aws-proton-sample-lambda-crud-service>

1.3. 画面右上[fork]のボタンをおします



1.4. 自分のアカウントにレポジトリがフォークされたことを確認します

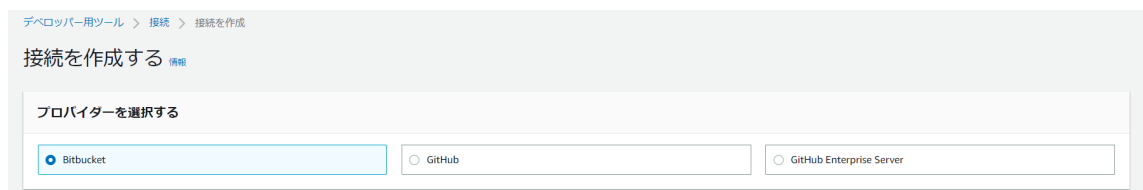


1.5. Proton のマネージメントコンソールから[ソース接続]をクリックします



1.6. [接続の作成]ボタンをおします

1.7. [GitHub]を選択し、適当な接続名をいれて、[GitHub に接続する]ボタンをおします



1.8. [新しいアプリをインストールする]ボタンをおします



1.9. 画面が github へ遷移しますので、[Save]ボタンをおします



## AWS Connector for GitHub

🕒 Installed 3 days ago

👤 Developed by aws

🔗 <https://docs.aws.amazon.com/dtconsole/latest/userguide/welcome-connections.html>

Enables you to connect GitHub with AWS

### Permissions

✓ Read access to issues and metadata

✓ Read and write access to administration, code, and pull requests

### Repository access

☒ All repositories

This applies to all current *and* future repositories.

☐ Only select repositories

Save

Cancel

1.10. AWS マネージメントコンソールに画面が戻ってきますので[接続]ボタンをおします

デベロッパー用ツール > 接続 > 接続を作成

## GitHub に接続する

### GitHub 接続設定 情報

接続名

GitHub アプリ

GitHub アプリは、GitHub との間に接続リンクを作成します。開始するには、新しいアプリをインストールし、この接続を保存します。

または

1.11. 以下のように、ステータスが利用可能になれば設定成功です



## 2. Proton 用 IAM ロールの作成

2.1. 左のペインから[アカウントロール]をクリックし[設定]のボタンを押します

### AWS Proton

環境

サービス

サービスインスタンス

#### ▼ テンプレート

環境テンプレート

サービステンプレート

#### ▼ 設定

ソース接続 

**アカウントロール**

2.2. [新しいサービスロール]を選択して、適当な名前をつけます。[アカウントに管理特....]の部分にチェックを付け、[変更を保存]をおします

## CI/CD パイプラインロールを設定

### CI/CD パイプラインロール

**パイプラインサービスロール**  
このロールにより、AWS Proton は CloudFormation API 呼び出しを行い、ユーザーに代わってパイプラインの CloudFormation スタックをデプロイできます。

☒ **新しいサービスロール**  
アカウントにサービスロールを作成します。

☐ **既存のサービスロール**  
アカウント内の既存のサービスロールを選択します。

**パイプラインサービスロール名**

protonrole20210305

文字、数字、ハイフンだけを使用します。最大長は 100 文字です。

☒ アカウントに管理特権を持つ Proton ロールを作成することに同意します。これにより、Proton とアカウントにテンプレートを登録できるすべてのユーザーに管理権限が与えられます。  
[Proton のアクセス制御について詳しくはこちらをご覧ください。](#)

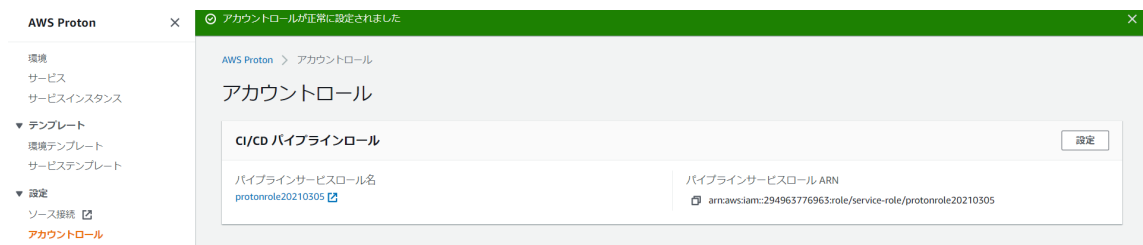
**テンプレートに必要なリソースとアクションのロールを制限することをお勧めします。**

► **Service role policy**

キャンセル

変更を保存

2.3. 以下の画面が表示されれば、アカウントロールの設定が完了です



### 3. 環境の構築

環境とは、いわゆるアプリケーションを実行させるインフラストラクチャなどを管理する箇所になります。

3.1. 左のテンプレートから環境テンプレートをクリックします

## AWS Proton

環境

サービス


サービスインスタンス

### ▼ テンプレート

環境テンプレート

サービステンプレート

### ▼ 設定

ソース接続 

アカウントロール

### 3.2. [環境テンプレートを作成]のボタンをおします



### 3.3. [Use one of our....]を選びます。

## 環境テンプレートを作成

環境テンプレートを作成すると、ベイクされたパラメータを使用して環境を簡単にデプロイできます。

### ▼ 環境テンプレートバンドルの仕組み



テンプレートバンドル



Proton テンプレート

テンプレートバンドルには、Proton が環境とサービスをデプロイするために必要なすべての情報が含まれています。組織のベストプラクティスに従って、独自のテンプレートバンドルを作成します。

[環境テンプレートバンドルの詳細について](#)

テンプレートバンドルは、それらをテンプレートとして作成して Proton に登録すると、再利用可能で、標準化かつ厳選されたテンプレートになります。

### テンプレートバンドルソース

- ☒ Use one of our sample template bundles  
Get started right away with a sample template bundle.
- ☐ Use your own template bundle  
Provide your own environment template bundle and upload to S3.

3.4. [サンプルテンプレートバンドル]から lambda-environment を選びます。

### サンプルテンプレートバンドル

#### テンプレートバンドルソース

このサンプルテンプレートバンドルのコレクションは、Proton の使用を開始して独自のテンプレートをすばやく構築するのに役立ちます。

lambda-environment ▲

fargate-environment

lambda-environment

3.5. テンプレート名に適切な名前を付け、[環境テンプレートを作成]をおします

3.6. しばらく待つと、ステータスが[Draft]になります。

lambdateemplate

環境テンプレートの詳細

環境テンプレートの表示名 lambdateemplate	環境テンプレートの説明 -	推奨バージョン -	作成済み 2021/3/5 13:10
環境テンプレートの名前 lambdateemplate	AWS KMS キー -		最終変更日 2021/3/5 13:10

発行することを忘れないでください  
マイナーバージョン 1.0 は「下書き」ステータスです。デベロッパーチームに表示するには、公開する必要があります。マイナーバージョンのみが公開されることに注意してください。

v1.0 を公開

テンプレートのバージョン (2)

最終フェッチ済み 10 秒前

キーワードで検索

	テンプレートのバージョン	説明	作成済み	最終変更日	ステータス	ステータスメッセージ
<input type="radio"/>	1.0	-	2021/3/5 13:10	2021/3/5 13:10	Draft	-
<input type="radio"/>	1	-	2021/3/5 13:10	2021/3/5 13:10	-	-

3.7. この時点では、まだ公開されておらず Draft 状態ですので、テンプレートを選んで [公開] ボタンをおします

3.8. 以下のようにステータスが変わればテンプレートが公開されています

テンプレートのバージョン (2)

最終フェッチ済み 40 秒前

キーワードで検索

	テンプレートのバージョン	説明	作成済み	最終変更日	ステータス	ステータスメッセージ
<input type="radio"/>	1.0 <b>推奨</b>	-	2021/3/5 13:10	2021/3/5 13:13	Published	-
<input type="radio"/>	1	-	2021/3/5 13:10	2021/3/5 13:10	-	-

3.9. [環境を作成する] ボタンをおします

このテンプレートから作成された環境

キーワードで検索

名前	テンプレートのバージョン	デプロイのステータス	最後に成功したデプロイ	作成済み
環境が見つかりません このアカウントには環境がありません。				

環境を作成する

3.10. テンプレートを選んで、[次へ]をおします

環境テンプレートを選択

Proton 環境テンプレートは、すべての共有リソースを定義します。

環境テンプレート (1)

キーワードで検索

lambdateemplate

最終更新日: 2021/3/5

バージョン: 1.0

キャンセル 次へ

3.11. 環境名に適当な名前を付け、先ほど作成した IAM ロールを [環境ロール] から選びます



## 環境を設定

環境のメタデータ設定。

### 環境設定

環境名

testenvironment

文字、数字、ハイフンのみを使用します。最大長は 100 文字です。

環境の説明 - オプション

最大長は 255 文字です。

### 環境ロール

Proton service role

デフォルトのパイプラインロールを使用した場合は、それを再利用できます。または、テンプレートで使用されるリソースの範囲を限定したロールを作成することもできます。 [詳細については、ドキュメントを参照してください。](#)

protonrole20210305

- 3.12. 次の画面はデフォルトのサンプルテンプレートがパラメータとして取得を必要としている、DynamoDB の TTL を設定する画面ですが、空欄で作業可能ですので、そのまま進めるため[次へ]をおします。さらにその次の画面で[作成]を押します

Environment detail

Environment name	Description
testenvironmentproton	-

Environment roles

Proton service role option
protonrole20210305

Tags

キー	値

ステップ 3: カスタム設定を構成

編集

Environment input

Ttl attribute
ttl

キャンセル

戻る

作成

3.13. 環境テンプレートから環境の作成が開始され、デプロイのステータスが[In Progress]となるので、少し待ちます。

AWS Proton > 環境 > testenvironmentproton			
testenvironmentproton		🔄	アクション ▼
環境の詳細			
環境名	環境 ARN	デプロイのステータス	作成済み
testenvironmentproton	arn:aws:proton:ap-northeast-1:294963776963:environment/testenvironmentproton	🔄 In progress	2021/3/5 13:31
環境の説明	環境サービスのロール	デプロイステータスメッセージ	最後に成功したデプロイ
-	arn:aws:iam::294963776963:role/service-role/protonrole20210305		-
		テンプレート	前回のデプロイの試行
		lambda template	2021/3/5 13:31
		テンプレートのバージョン	-
		-	

その間 CloudFormation の画面へ遷移すると、一つスタックが生成されていることがわかります。

AWSProton-testenvironment-cloudformation--MJNRSEEBURXAXPO

削除 更新する スタックアクション ▼ スタックの作成 ▼

スタックの情報 イベント **リソース** 出力 パラメータ テンプレート 変更セット

リソース (1)

リソースの検索

論理 ID	物理 ID	タイプ	ステータス	状況の理由	モジュール
AppTable	AWSProton-testenvironment-cloudformation-MJNRSEEBURXAXPO-AppTable-1O5Q40GOHXU08	AWS::DynamoDB::Table	CREATE_COMPLETE	-	-

このスタックでは DynamoDB が生成されています。念のため DynamoDB の画面でテーブルを確認してください。

テーブルの作成 テーブルの削除

テーブル名によるフィルター X

テーブルグループ... ▼ アクション ▼

名前

● AWSProton-testenvironment-cloudfor

AWSProton-testenvironment-cloudformation-MJNRSEEBURXAXPO-AppTable-1O5Q40GOHXU08 開じる

概要 項目 メトリクス アラーム キャパシティ インデックス グローバルテーブル バックアップ 投稿者のインサイト トリガー さらに ▼

最近のアラート

このテーブルでトリガーされている CloudWatch アラームがありません。

Kinesis データストリームの詳細

Amazon Kinesis Data Stream for DynamoDB を使用して、テーブルの項目レベルの変更を Kinesis データストリームにストリーミングできます。 [詳細はこちら](#)

このワークショップでは、環境は Dynamo を管理し、それとは別チームがサービスで Dynamo を利用する Lambda などを構築する、という流れになります。そして、環境、サービスともに Proton から起動される構築スクリプトは全て CloudFormation スクリプトとなります。つまり、Proton とはサーバレスに特化した CloudFormation の管理インターフェースを提供するサービス、といえます

3.14. 環境テンプレートの画面から、環境が正しく作成されたことを確認します。

環境テンプレートの詳細

環境テンプレートの表示名 lambdateemplate	環境テンプレートの説明 -	推奨バージョン 推奨: 1.0	作成済み 2021/3/5 13:10
環境テンプレートの名前 lambdateemplate	AWS KMS キー -		最終変更日 2021/3/5 13:10

テンプレートのバージョン (2) 最終フェッチ済み 今 削除 公開 新しいバージョンを作成

キーワードで検索

テンプレートのバージョン	説明	作成済み	最終変更日	ステータス	ステータスメッセージ
1.0 推奨	-	2021/3/5 13:10	2021/3/5 13:13	Published	-
1	-	2021/3/5 13:10	2021/3/5 13:10	-	-

このテンプレートから作成された環境 (1)

キーワードで検索

名前	テンプレートのバージョン	デプロイのステータス	最後に成功したデプロイ	作成済み
testenvironmentproton	1.0	Succeeded	2021/3/5 13:31	2021/3/5 13:31

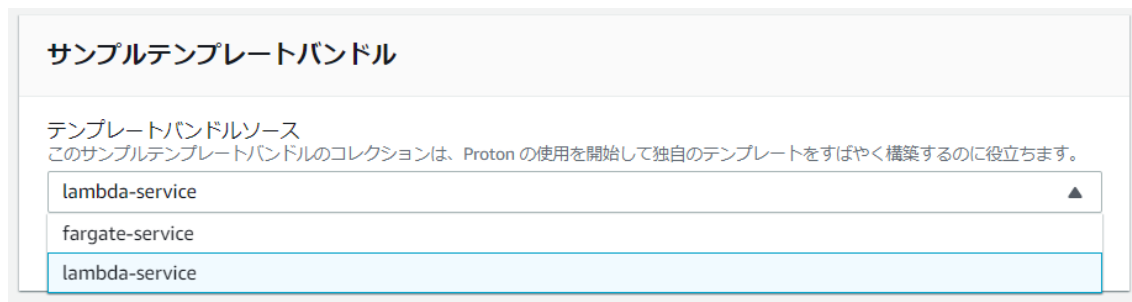
#### 4. サービスの構築

サービスとは、環境の上に構築されるアプリケーション実行環境そのものになります。環境を AWS アカウント全体管理者が管理し、そのうえで開発者がサービスを構築する、という利用用途を想定しています。

##### 4.1. 左のペインから[サービステンプレート]をクリックします



##### 4.2. [Use one of …]を選択し、[lambda service]をドロップダウンから選びます



##### 4.3. 適当な名前を付け、[互換性のある環境テンプレート]から先ほど作成した Lambda 用環境のテンプレートを選びます

### テンプレートの詳細

テンプレート名  
サービステンプレート名は、作成後に変更することはできません。

lambdateemplate

文字、数字、ハイフンのみを使用します。最大長は 100 文字です。

テンプレートの表示名 - オプション  
これは、デベロッパーに表示されるテンプレート名です。

My template name

最大長は 100 文字です。

テンプレートの説明 - オプション  
これは、デベロッパーに表示されるテンプレートの説明です。

最大長は 255 文字です。

### 互換性のある環境テンプレート

lambdateemplate

☐ - lambdateemplate バージョン 1

環境テンプレートを選択 ▲

#### ▼ 追加の構成 - オプション

パイプラインオプション  
このテンプレートにパイプラインを使用しない場合は、この選択を解除します

☒ このテンプレートには CI/CD パイプラインが含まれています

キャンセル

サービステンプレートを作成

4.4. [パイプラインオプション]をオンにして[サービステンプレートを作成]ボタンをおします

4.5. しばらく待つとステータスが[Draft]になります



## サービステンプレートを選択

Proton サービステンプレートは、すべてのインフラストラクチャリソース、CI/CD パイプライン、および可観測性ツールを定義します。

サービステンプレート (1/1)

🔍 キーワードで検索

< 1 > ⚙️

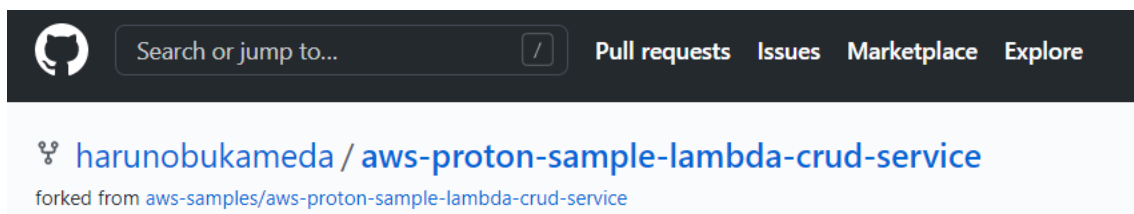
lambdateemplate

最終更新日: 2021/3/5

バージョン: 1.0

キャンセル 設定

- 4.10. 適当なサービス名を付け、[サービスレポジトリの設定]パートで[ブランチ名]に [main]と記載します。(異なる github ブランチを使用している方は別の名前を付けてください)。レポジトリ名は github の以下の部分を使用します。以下の例ですと [harunobukameda/aws-proton-sample-lambda-crud-service]になります。github からコピすると [ / ]の前後に半角スペースが入るので気を付けてください。



- 4.11. [リポジトリ接続]から先ほど作成した github 接続を選択します

サービスリポジトリの設定

ブランチ名

main

リポジトリブランチ

リポジトリ ID

ソースの変更が検出されるリポジトリの所有者と名前。

harunobukameda/aws-proton-sample-lambda-crud-service

リポジトリ接続

接続されたソースアカウントの既存のリポジトリ。 [新しいソース接続を追加](#)

github20210305

リポジトリ接続 ARN

- 4.12. [次へ]ボタンをおすと、CloudFormation テンプレートが要求している Parameter 入力画面が表示されます。適当な名前を入力し、[Environment]は先ほど作成した環境を指定し、残りはそのまま[次へ]ボタンをおします。さらに次の画面で[作成]ボタンをおします。

## カスタム設定を構成

これらは、管理者によって作成された設定です。

### Crud service input

#### testinstance

Name

testinstance

Environment

testenvironmentproton

- 4.13. サービスインスタンスが作成中となります。ここでいうインスタンスは EC2 インスタンスとは別物です。サービスのテンプレートで指定された CloudFormation スタックのことを意図しています。

protonservice

概要 バイブライン

### サービスの詳細

サービス名 protonservice	ARN arn:aws:proton:ap-northeast-1:294963776963:service/protonservice	サービスステータス Create started	作成済み 2021/3/5 13:48
サービスの説明 -	Service template lambdateemplate v.1.0	パイプラインのステータス In progress	

### サービスインスタンス (1)

最終フェッチ済み 20 秒前

サービスを作成

キーワードで検索

名前	Service name	Template name	デプロイのステータス	デプロイステータスメッセージ	最後に成功したデプロイ	作成済み
testinstance	protonservice	lambdateemplate	In progress	-	2021/3/5 13:48	2021/3/5 13:48

CloudFormation の画面を見ると、スタックが3つ作成されています。API Gateway と、CodeBuild プロジェクト、S3 バケット、IAM ロール、KMS の鍵、そして Lambda 関数などです。

スタックの情報 イベント リソース 出力 パラメータ テンプレート 変更セット

### リソース (2)

リソースの検索

論理 ID	物理 ID	タイプ	ステータス	状況の理由	モジュール
CrudHttpApi	cq27ln72ub	AWS::ApiGatewayV2::Api	CREATE_COMPLETE	-	-
CrudHttpApiApiGatewayDefaultStage	\$default	AWS::ApiGatewayV2::Stage	CREATE_COMPLETE	-	-



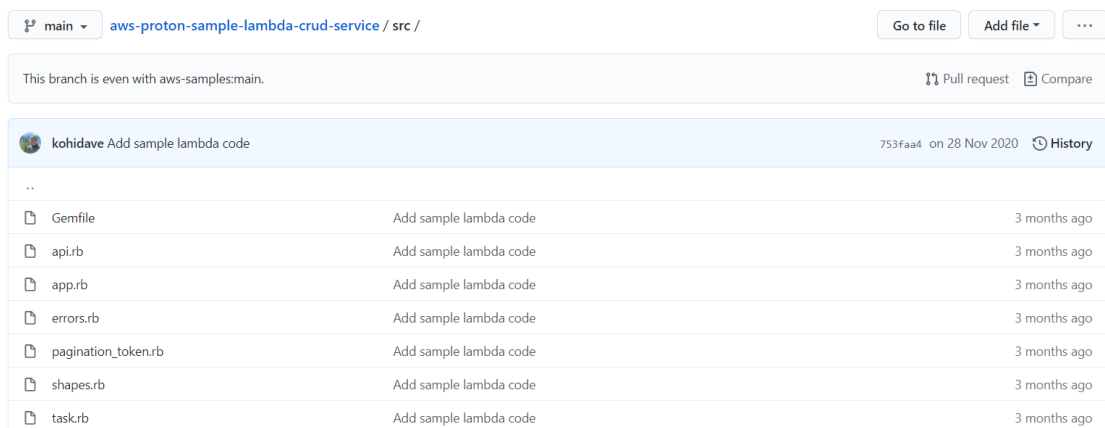
リソース (17)						
<div> <input type="text" value="リソースの検索"/> </div>						
論理 ID	物理 ID	タイプ	ステータス	状況の理由	モジュール	
BuildProject	BuildProject-wQcPXwQ08nRY	AWS::CodeBuild::Project	CREATE_COMPLETE	-	-	
Deploy1Project	Deploy1Project-AI4Vaqv7hHN5	AWS::CodeBuild::Project	CREATE_COMPLETE	-	-	
DeploymentRole	<a href="#">AWSProton-protonservice-cloudformat-DeploymentRole-1X5F5ISWIG90K</a>	AWS::IAM::Role	CREATE_COMPLETE	-	-	
DeploymentRoleDefaultPolicy	AWSPr-Depl-KMSYUCXGMQCS	AWS::IAM::Policy	CREATE_COMPLETE	-	-	
FunctionBucket	<a href="#">awsproton-protonservice-cloudformat-functionbucket-1939ac09il4pv</a>	AWS::S3::Bucket	CREATE_COMPLETE	-	-	
Pipeline	<a href="#">AWSProton-protonservice-cloudformation-WLNFHFGHSPNYCCEI-Pipeline-ORUGY7O7TVNO</a>	AWS::CodePipeline::Pipeline	CREATE_COMPLETE	-	-	

リソース (17)						
<div> <input type="text" value="リソースの検索"/> </div>						
論理 ID	物理 ID	タイプ	ステータス	状況の理由	モジュール	
CreateFunction	<a href="#">AWSProton-protonservice-testinstance-CreateFunction-XYFBXMEBTCEJ</a>	AWS::Lambda::Function	CREATE_COMPLETE	-	-	
CreateFunctionCreateApiPermission	<a href="#">AWSProton-protonservice-testinstance-cloudformation-CreateFunctionCreateApiPermission-17559VNHBMHMY</a>	AWS::Lambda::Permission	CREATE_COMPLETE	-	-	
CreateFunctionRole	<a href="#">AWSProton-protonservice-testinstance-CreateFunctionRole-ZH3LA6BGBAMW</a>	AWS::IAM::Role	CREATE_COMPLETE	-	-	
CrudHttpApi	<a href="#">cq27ln72ub</a>	AWS::ApiGatewayV2::Api	UPDATE_COMPLETE	-	-	
CrudHttpApiApiGatewayDefaultStage	\$default	AWS::ApiGatewayV2::Stage	CREATE_COMPLETE	-	-	

4.14. 無事作成されると以下のような画面が表示されます。

testinstance															
<div> <div>サービスインスタンスの詳細</div> <table> <tr> <td> <div>サービスインスタンス名</div> testinstance </td><td> <div>環境</div> testenvironmentproton </td><td> <div>デプロイのステータス</div> <div>  Succeeded </div> <div>デプロイステータスメッセージ</div> - </td><td> <div>作成済み</div> 2021/3/5 13:48 </td></tr> <tr> <td> <div>サービス名</div> protonservice </td><td></td><td> <div>テンプレート</div> <a href="#">lambdatemplate</a> </td><td> <div>最後に成功したデプロイ</div> 2021/3/5 13:48 </td></tr> <tr> <td> <div>ARN</div> arn:aws:proton:ap-northeast-1:294963776963:service/protonservice/service-instance/testinstance </td><td></td><td> <div>テンプレートのバージョン</div> 1.0 </td><td> <div>前回のデプロイの試行</div> 2021/3/5 13:48 </td></tr> </table> </div>				<div>サービスインスタンス名</div> testinstance	<div>環境</div> testenvironmentproton	<div>デプロイのステータス</div> <div>  Succeeded </div> <div>デプロイステータスメッセージ</div> -	<div>作成済み</div> 2021/3/5 13:48	<div>サービス名</div> protonservice		<div>テンプレート</div> <a href="#">lambdatemplate</a>	<div>最後に成功したデプロイ</div> 2021/3/5 13:48	<div>ARN</div> arn:aws:proton:ap-northeast-1:294963776963:service/protonservice/service-instance/testinstance		<div>テンプレートのバージョン</div> 1.0	<div>前回のデプロイの試行</div> 2021/3/5 13:48
<div>サービスインスタンス名</div> testinstance	<div>環境</div> testenvironmentproton	<div>デプロイのステータス</div> <div>  Succeeded </div> <div>デプロイステータスメッセージ</div> -	<div>作成済み</div> 2021/3/5 13:48												
<div>サービス名</div> protonservice		<div>テンプレート</div> <a href="#">lambdatemplate</a>	<div>最後に成功したデプロイ</div> 2021/3/5 13:48												
<div>ARN</div> arn:aws:proton:ap-northeast-1:294963776963:service/protonservice/service-instance/testinstance		<div>テンプレートのバージョン</div> 1.0	<div>前回のデプロイの試行</div> 2021/3/5 13:48												

API Gateway, S3 バケット, Lambda 関数ができていることを確認します。  
CloudFormation テンプレートの中では Lambda は指定されていませんが、先ほど指定した github のレポジトリから Lambda の関数を引っ張ってきています。

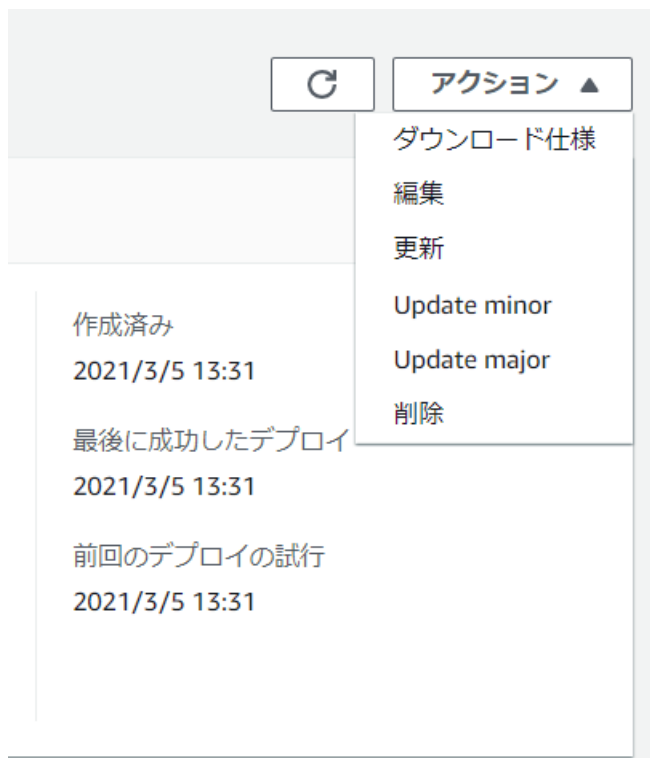


## 5. アプリケーションの実行

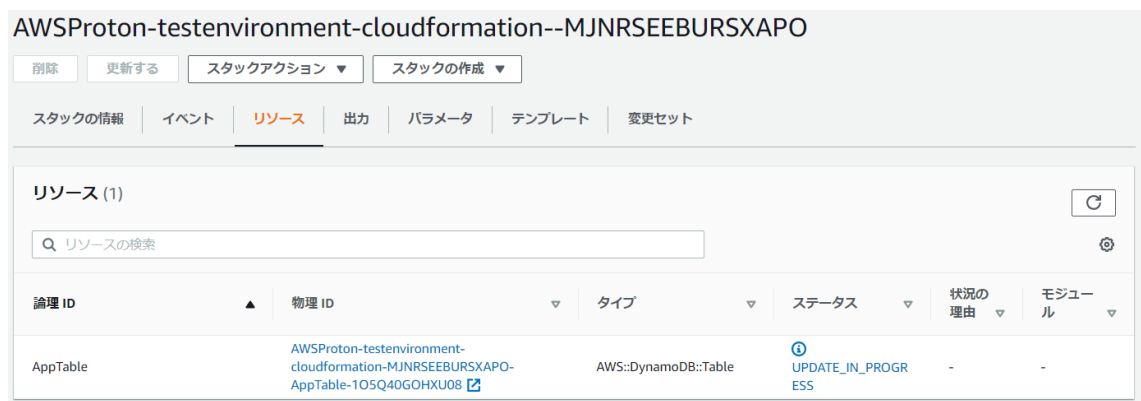
- 5.1. アプリケーションの実行の詳細は子のシナリオからは割愛しますが、詳細は  
<https://github.com/harunobukameda/aws-proton-sample-lambda-crud-service>  
<https://github.com/aws-samples/aws-proton-sample-lambda-crud-service>  
から確認ください。

## 6. 環境、サービスの更新

- 6.1. 今まで構築された環境及びサービスで誓われている CloudFormation テンプレートは、各詳細画面右上ボタンの[アクション]→[ダウンロード仕様]から確認が可能です。
- 6.2. 先ほどの手順では DynamoDB の ttl を設定しませんでした。仮に運用途中で必要になったとします。環境の詳細画面、アクションから[更新]を選びます。



- 6.3. [ttl 5]と指定すると、CloudFormation スタックが再実行され DynamoDB の設定が更新されます。



- 6.4. 次は、サービスを選択して、パイプラインのタブを選びます



- 6.5. [パイプラインを編集]を押して、ユニットテストに用いる文字列を少し変更し、パ

イプラインを更新]を押します

パイプラインを更新

**Pipeline inputs**

Pipeline unit test command - オプション  
The command to run to unit test the application code

Pipeline packaging command - オプション  
The commands which packages your code into a file called function.zip

キャンセル **パイプラインを更新**

CloudFormaiton が再度 Update されます

📦 **スタック (5)**

🔄

🔍 スタック名によるフィルター

アクティブ▼ ☒ ネスト表示

< 1 >

**AWSProton-protonservice-cloudformation--WLNFFGHSPNYCCEI**

2021-03-05 13:48:33 UTC+0900

📄 UPDATE\_IN\_PROGRESS

- 6.6. このワークショップ環境は、CICD パイプラインが Code Pipeline、Code Deploy で管理されています。サービス詳細画面、パイプラインタブの出力欄の URL をクリックすることで CICD パイプライン全体を確認することができます

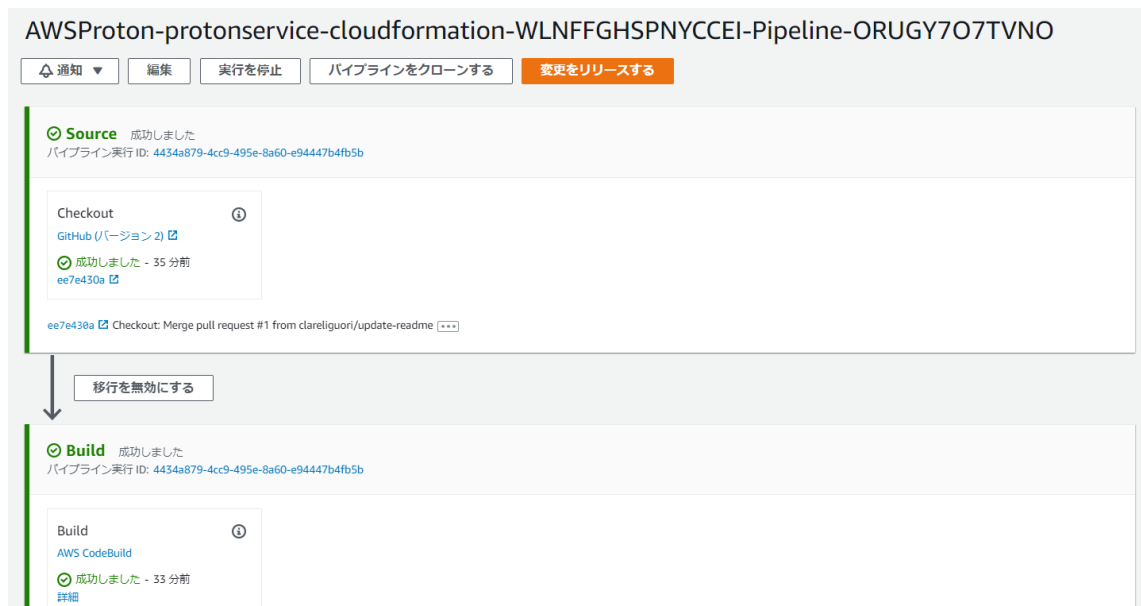
出力 (1)

🔍 キーワードで検索

< 1 >

キー ▲ 値 ▼

PipelineEndpoint	<a href="https://ap-northeast-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/AWSProton-protonservice-cloudformation-WLNFFGHSPNYCCEI-Pipeline-ORUGY7O7TVNQ/view?region=ap-northeast-1">https://ap-northeast-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/AWSProton-protonservice-cloudformation-WLNFFGHSPNYCCEI-Pipeline-ORUGY7O7TVNQ/view?region=ap-northeast-1</a>
------------------	---



## 7. おつかれさまでした

環境の削除は以下の順番で実行します

- サービス（サービスインスタンスも同時に削除されます）
- 環境
- サービステンプレート
- 環境テンプレート
- ソース接続
- S3 バケット
- github でフォークしたレポジトリ