AWS Proton ワークショップ

2021/03/05

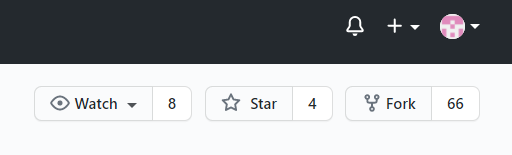
シニアエバンジェリスト

亀田

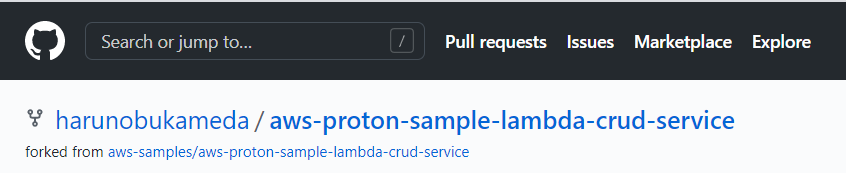
1. githubアカウントの設定
   1. ご利用のブラウザでgithubにログインを行いアクセスします
   2. 別のタブで以下のURLにアクセスします

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

* 1. 画面右上[fork]のボタンをおします



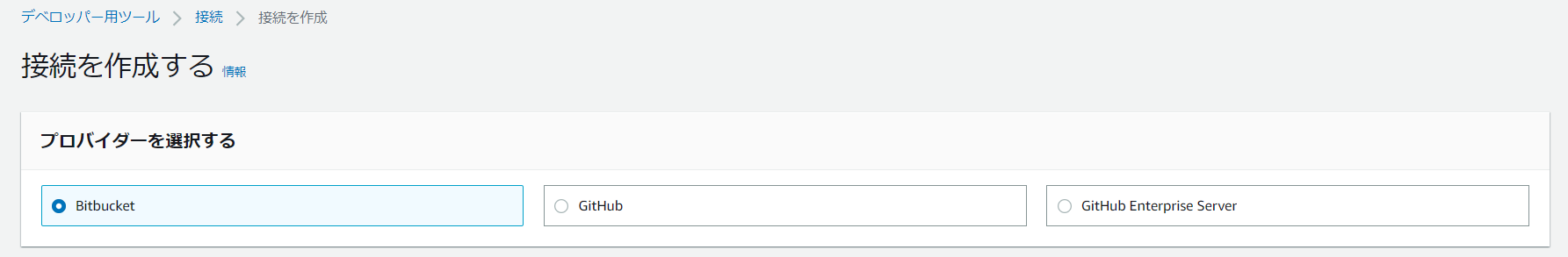
* 1. 自分のアカウントにレポジトリがフォークされたことを確認します



* 1. 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



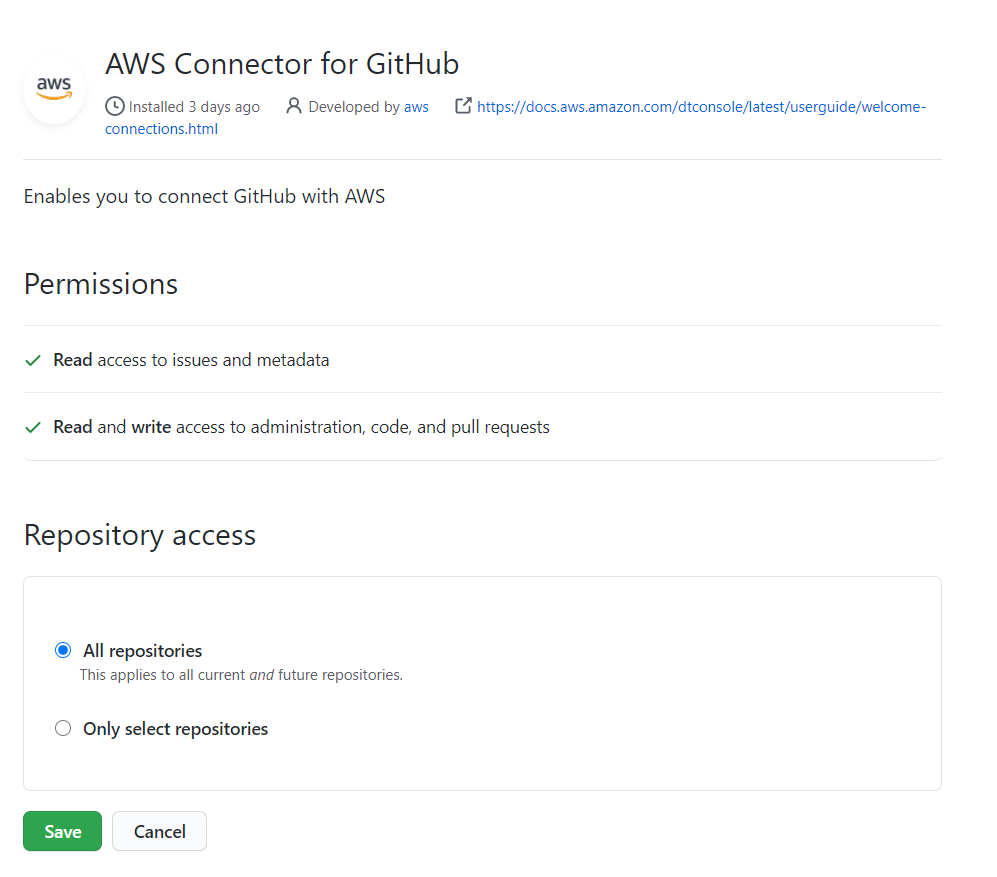
* 1. [接続の作成]ボタンをおします
  2. [GitHub]を選択し、適当な接続名をいれて、[GitHubに接続する]ボタンをおします



* 1. [新しいアプリをインストールする]ボタンをおします



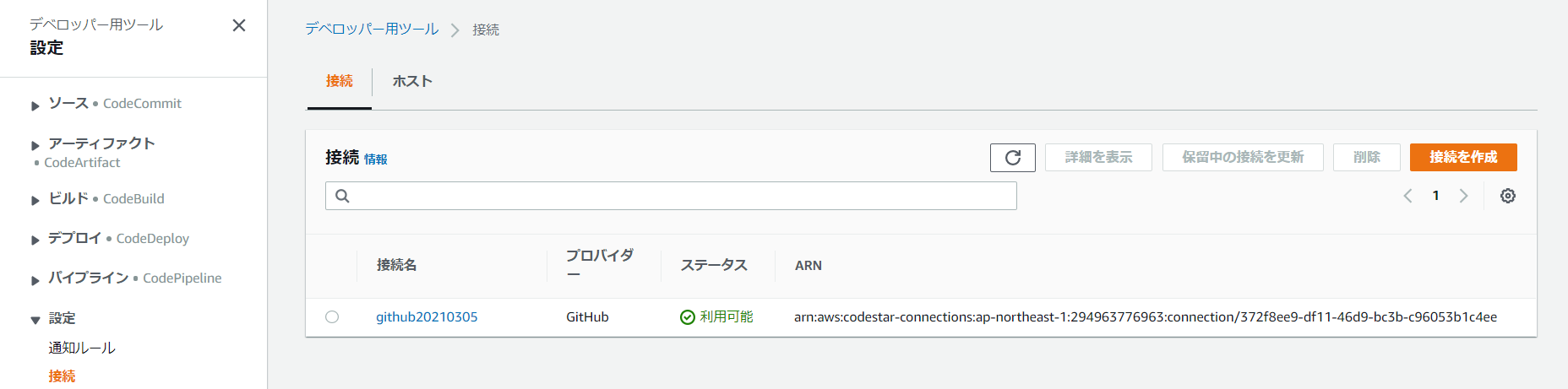
* 1. 画面がgithubへ遷移しますので、[Save]ボタンをおします



* 1. 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



* 1. 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



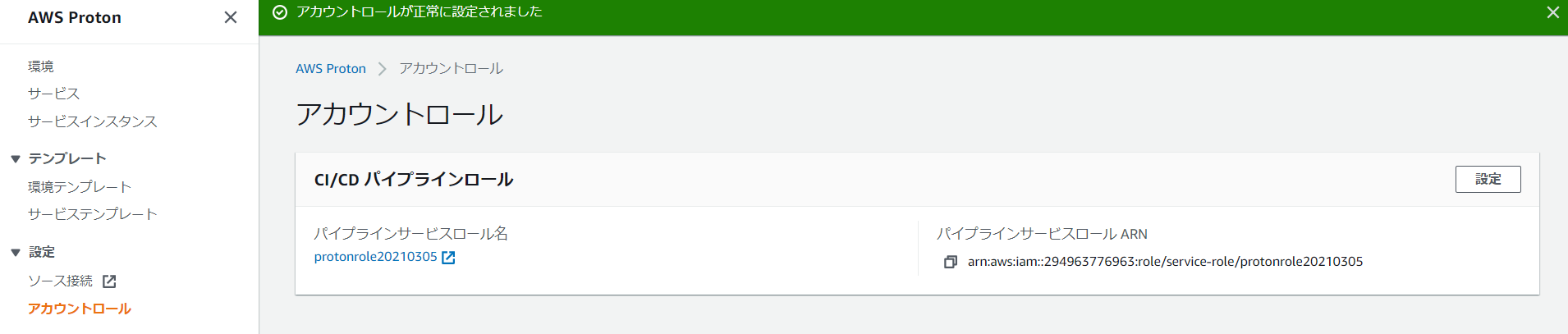
1. Proton用IAMロールの作成
   1. 左のペインから[アカウントロール]をクリックし[設定]のボタンを押します



* 1. [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



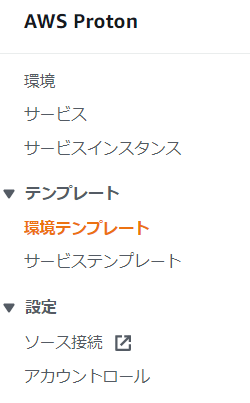
* 1. 以下の画面が表示されれば、アカウントロールの設定が完了です



1. 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。

* 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



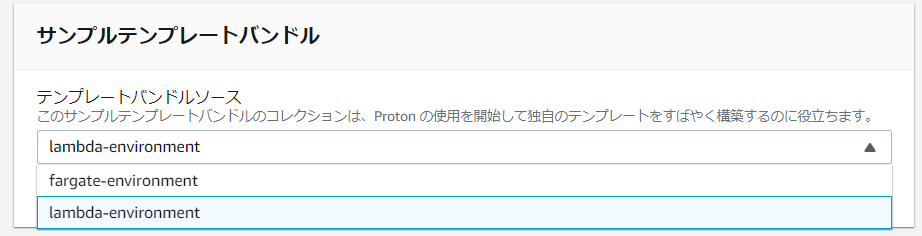
* 1. [環境テンプレートを作成]のボタンをおします



* 1. [Use one of our….]を選びます。



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



* 1. テンプレート名に適当な名前を付け、[環境テンプレートを作成]をおします
  2. しばらく待つと、ステータスが[Draft]になります。



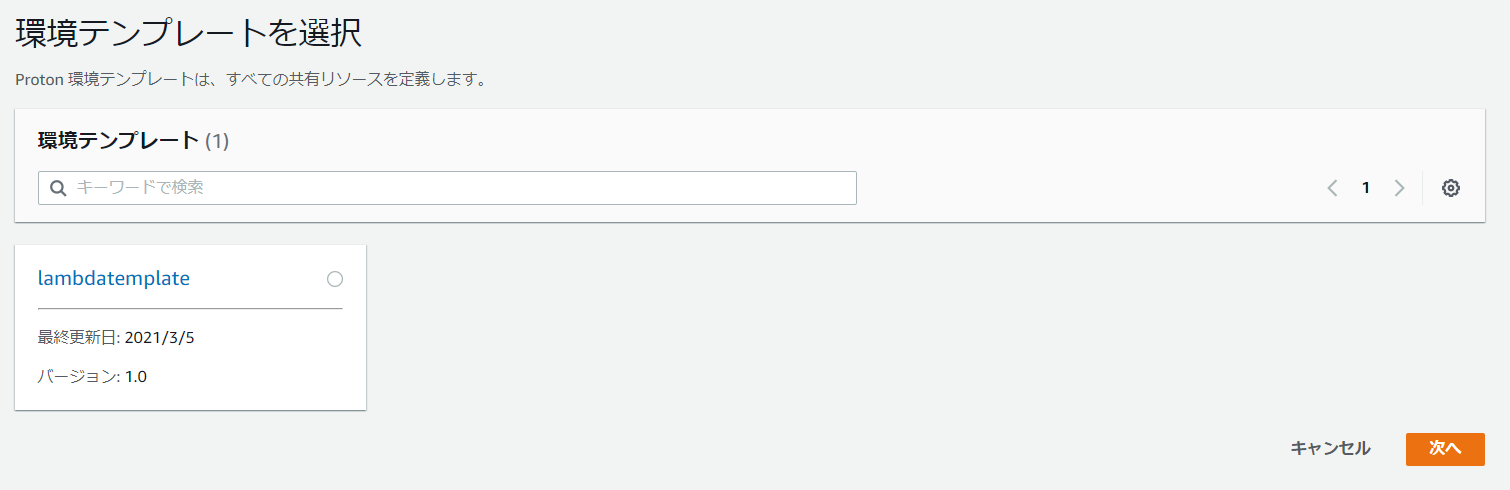
* 1. この時点では、まだ公開されておらずDraft状態ですので、テンプレートを選んで[公開]ボタンをおします
  2. 以下のようにステータスが変わればテンプレートが公開されています



* 1. [環境を作成する]ボタンをおします



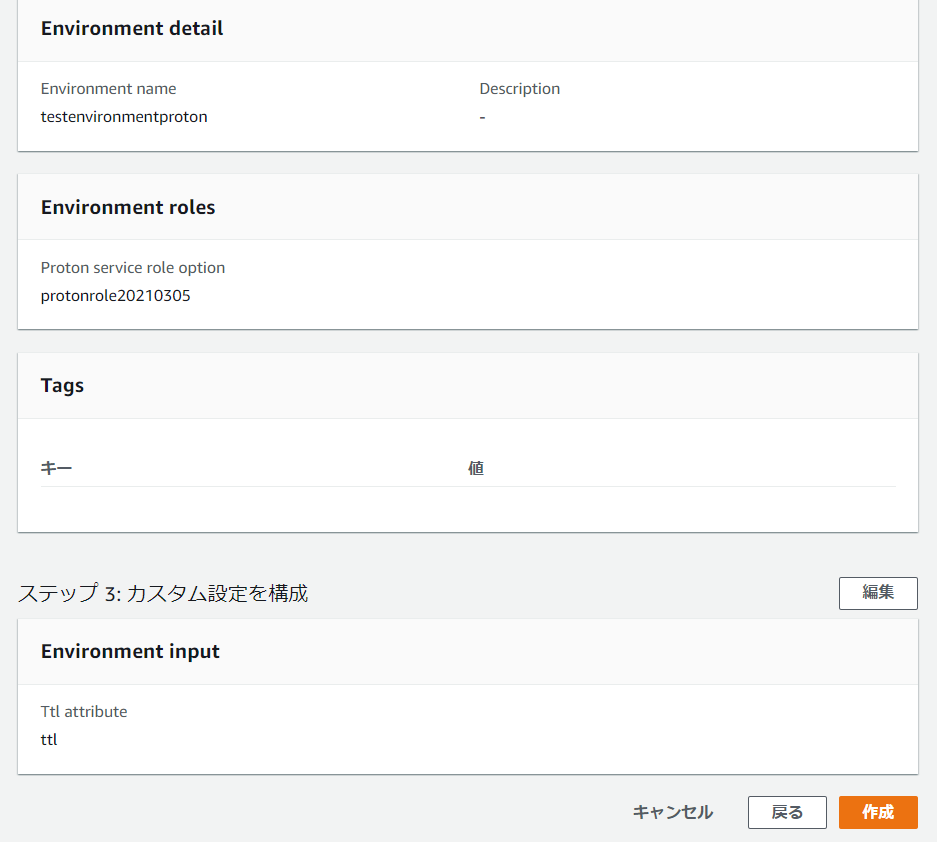
* 1. テンプレートを選んで、[次へ]をおします



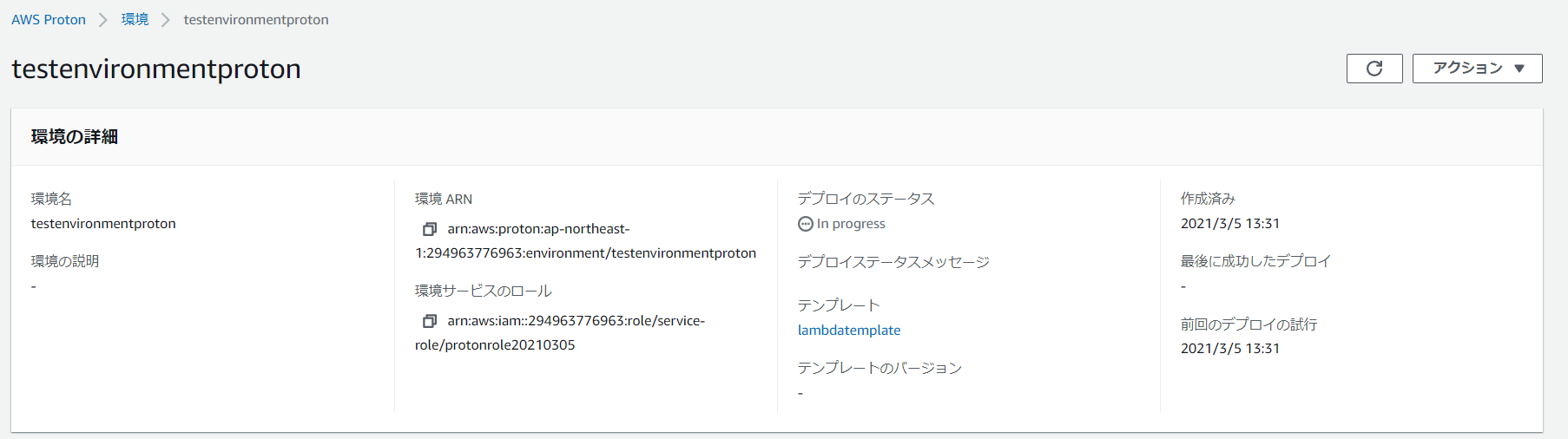
* 1. 環境名に適当な名前を付け、先ほど作成したIAMロールを[環境ロール]から選びます



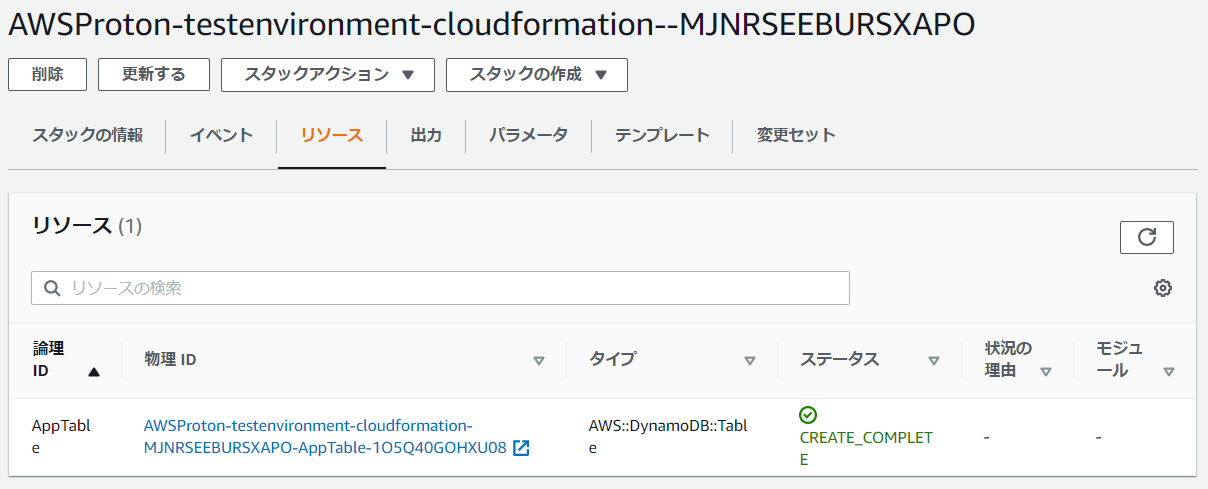
* 1. 次の画面はデフォルトのサンプルテンプレートがパラメータとして取得を必要としている、DynamoDBのTTLを設定する画面ですが、空欄で作業可能ですので、そのまま進めるため[次へ]をおします。さらにその次の画面で[作成]を押します



* 1. 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。



その間CloudFormaiton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。

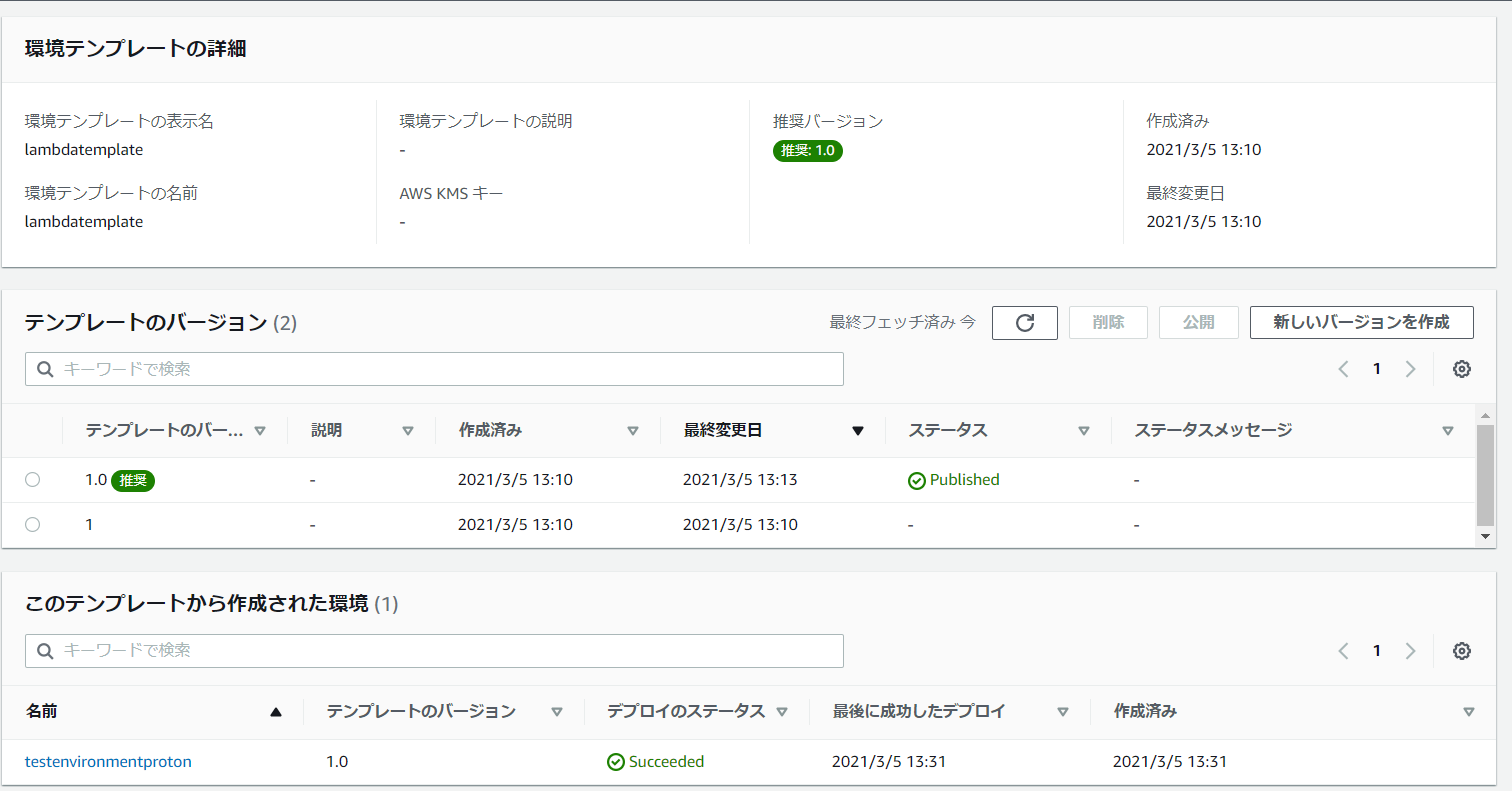


このスタックではDynamoDBが生成されています。念のためDynamoDBの画面でテーブルを確認してください。



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

* 1. 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。



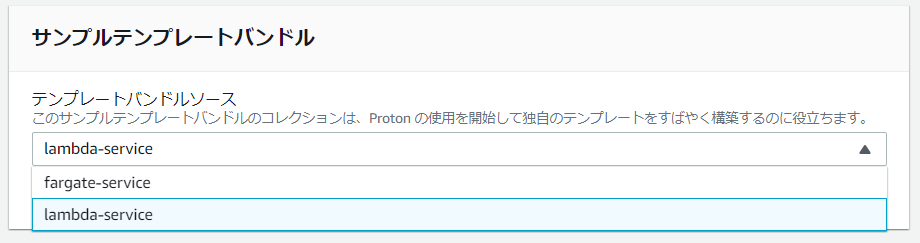
1. サービスの構築

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。

* 1. 左のペインから[サービステンプレート]をクリックし、[サービステンプレートの作成]をおします



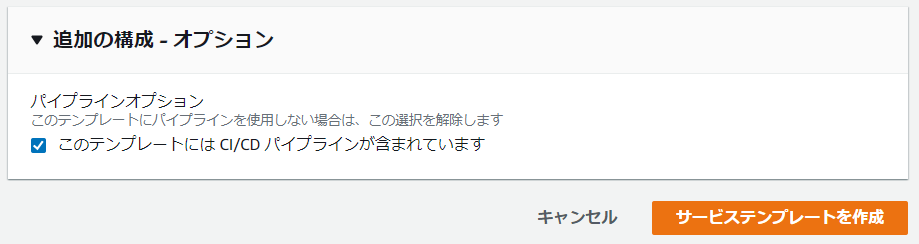
* 1. [サンプルテンプレートバンドルの1つを使用]を選び、 [lambda service]をドロップダウンから選びます



* 1. 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



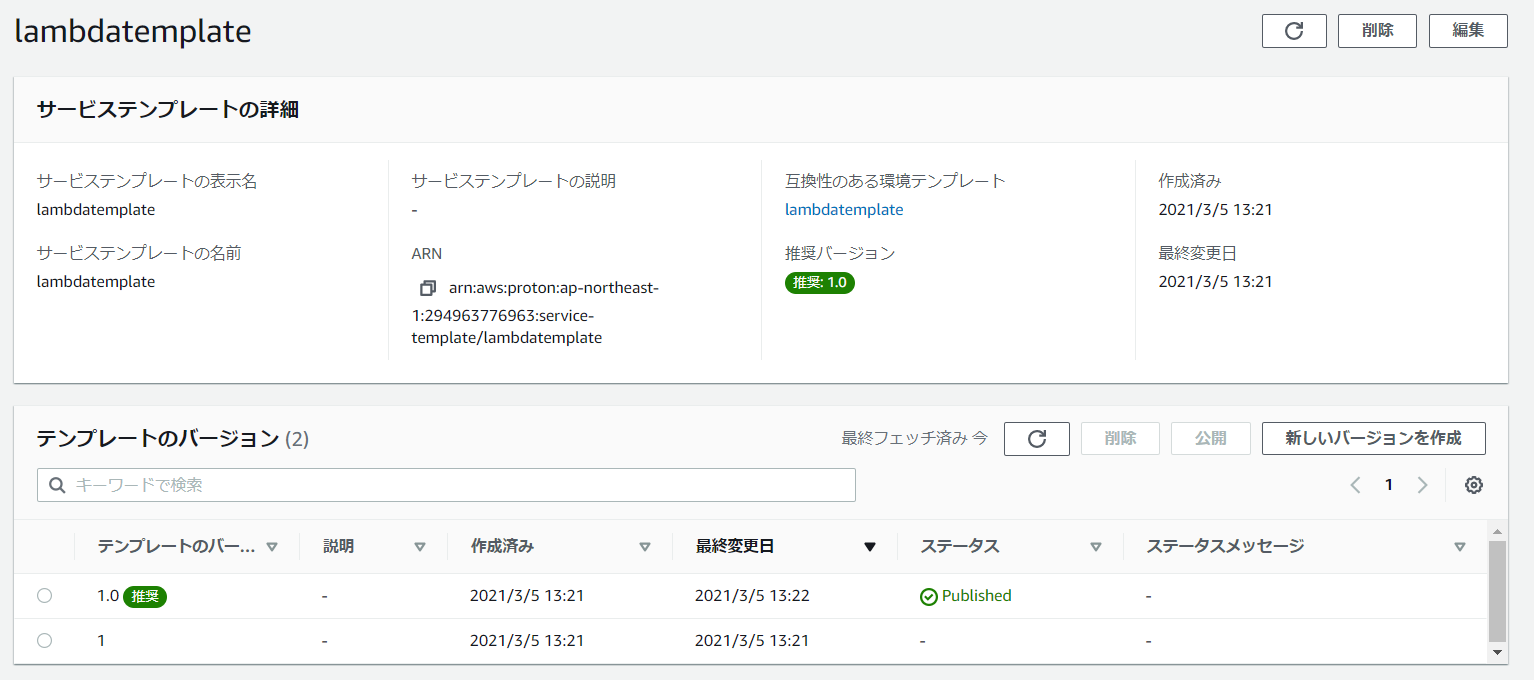
* 1. [パイプラインオプション]をオンにして[サービステンプレートを作成]ボタンをおします



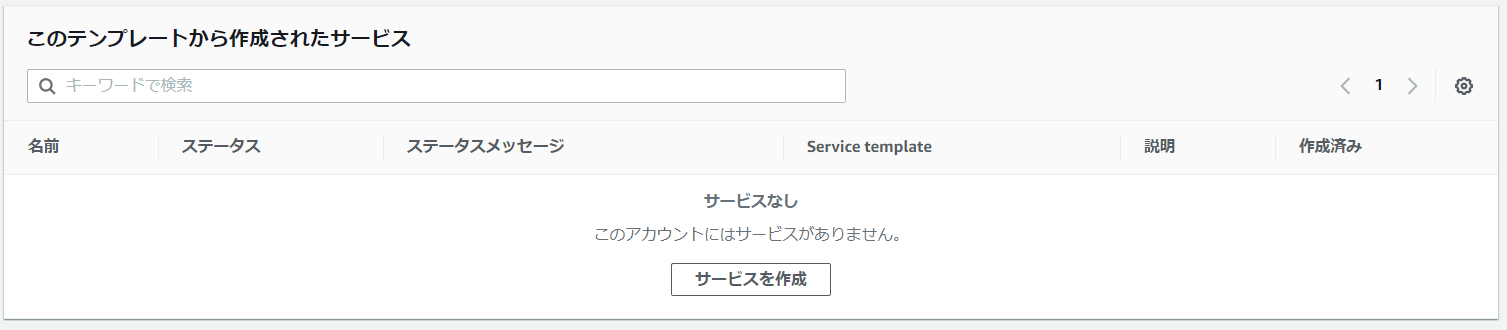
* 1. しばらく待つとステータスが[Draft]になります



* 1. 1.0を選んで[公開]ボタンをおします



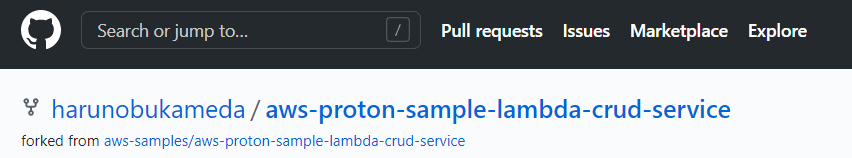
* 1. ステータスがPublishedになりました。
  2. [サービスの作成]を押します



* 1. 先ほど作成されたサービステンプレートを選んで、[設定]ボタンをおします



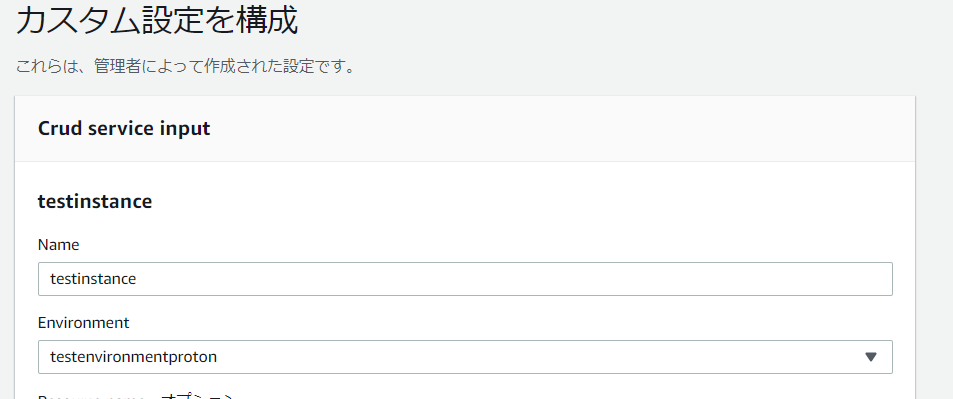
* 1. 適当なサービス名を付け、[サービスレポジトリの設定]パートで[ブランチ名]に[main]と記載します。（異なるgithubブランチを使用している方は別の名前を付けてください）。レポジトリ名はgithubの以下の部分を使用します。以下の例ですと[harunobukameda/aws-proton-sample-lambda-crud-service]になります。githubからコピーすると[ / ]の前後に半角スペースが入るので気を付けてください。



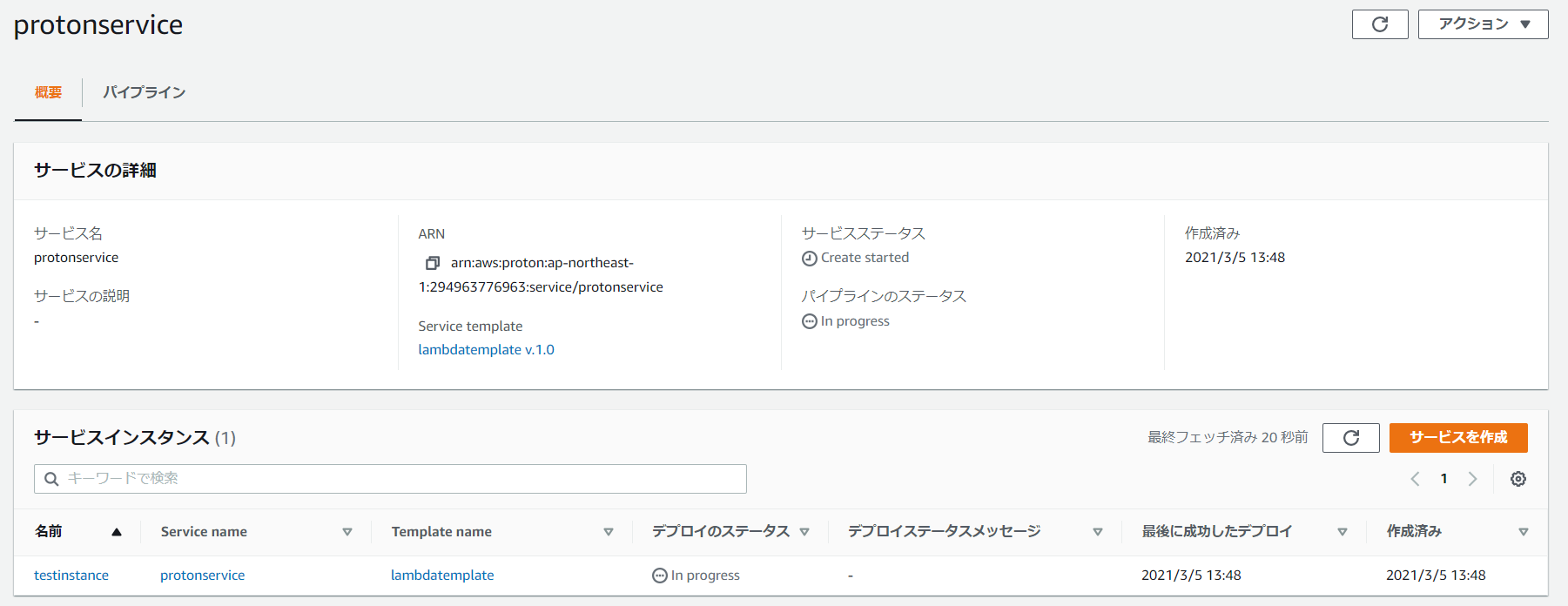
* 1. [リポジトリ接続]から先ほど作成したgithub接続を選択します



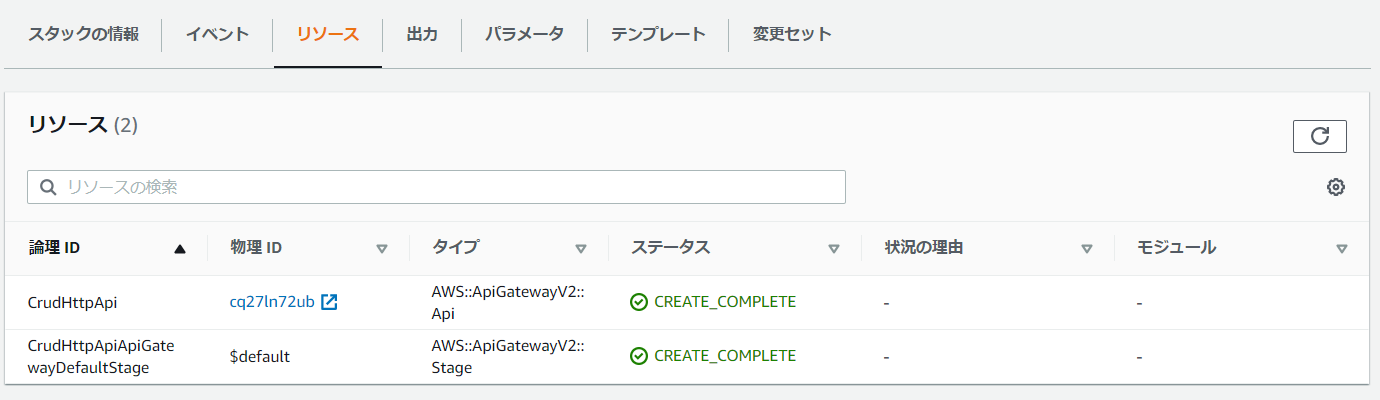
* 1. [次へ]ボタンをおすと、CloudFormationテンプレートが要求しているParameter入力画面が表示されます。適当な名前を入力し、[Environment]は先ほど作成した環境を指定し、残りはそのままで[次へ]ボタンをおします。さらに次の画面で[作成]ボタンをおします。

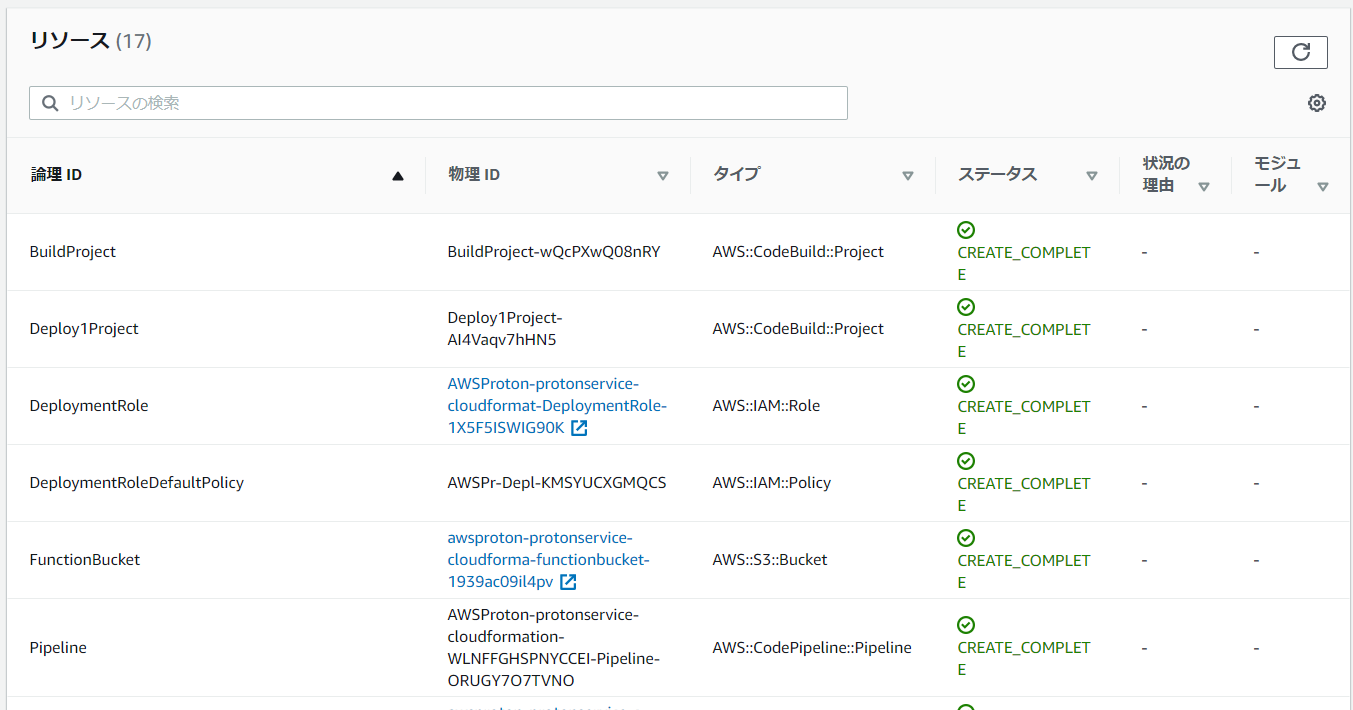


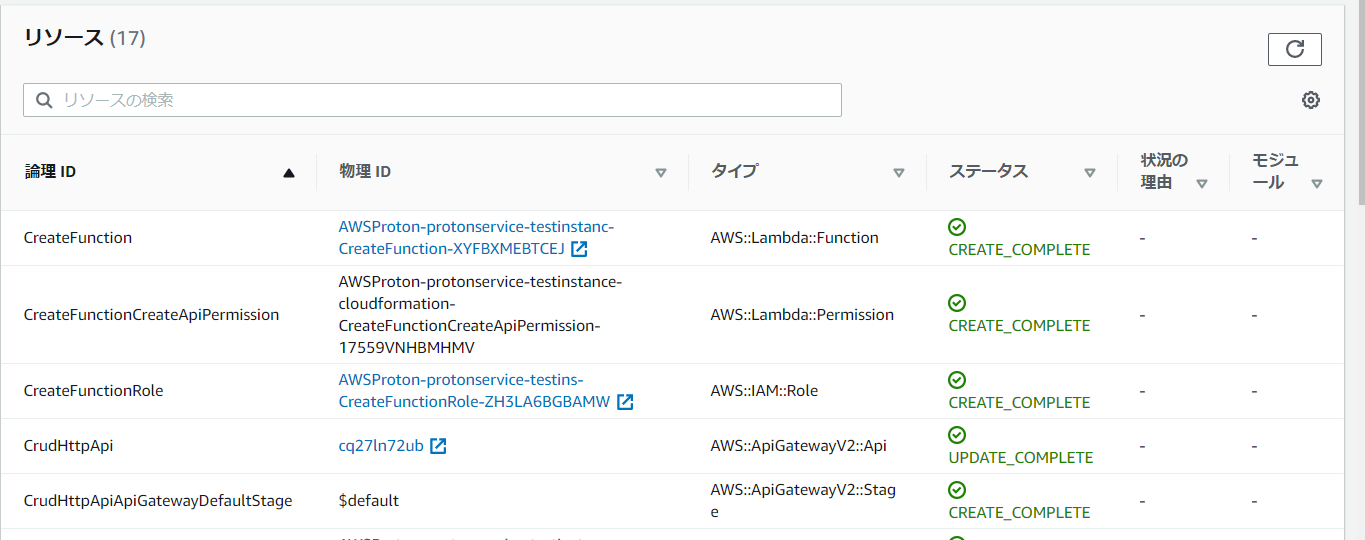
* 1. サービスインスタンスが作成中となります。ここでいうインスタンスはEC2インスタンスとは別物です。サービスのテンプレートで指定されたCloudFormationスタックのことを意図しています。



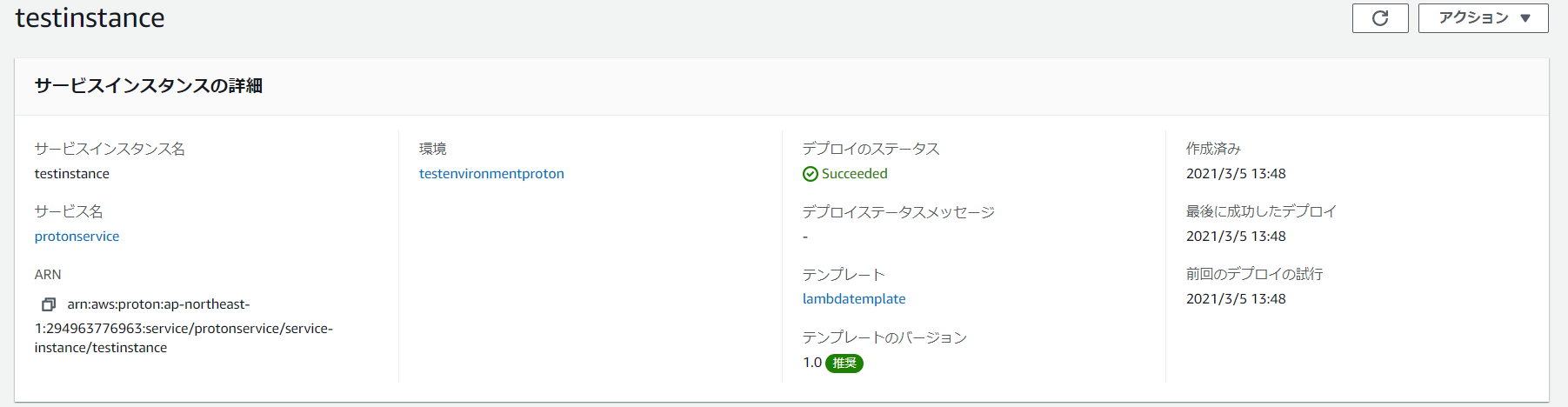
CloudFormationの画面を見ると、スタックが３つ作成されています。API Gatewayと、CodeBuiltプロジェクト、S3バケット、IAMロール、KMSの鍵、そしてLambda関数などです。





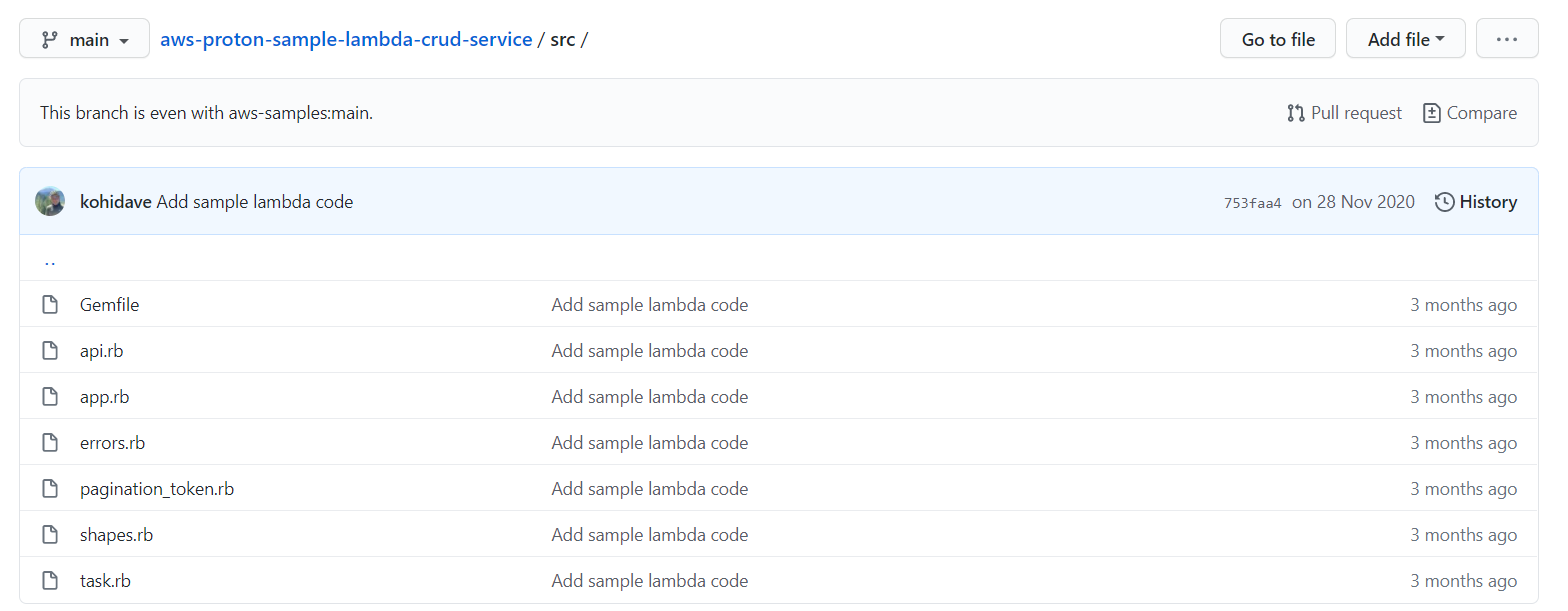


* 1. 無事作成されると以下のような画面が表示されます。



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。



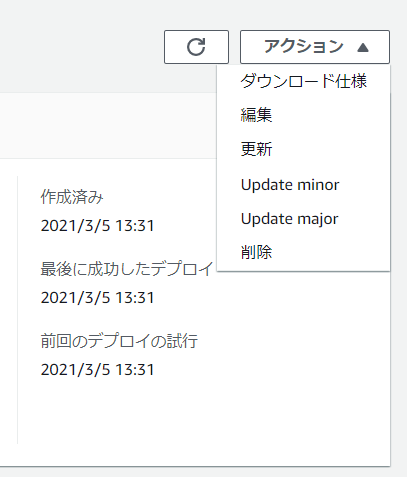
1. アプリケーションの実行
   1. アプリケーションの実行の詳細は子のシナリオからは割愛しますが、詳細は

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

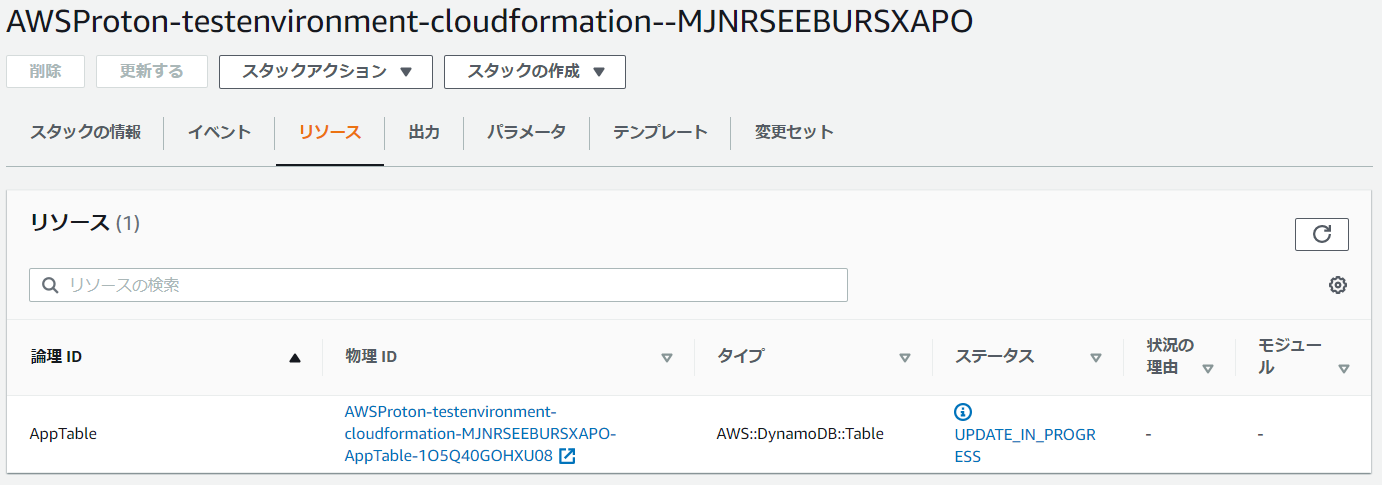
<https://github.com/aws-samples/aws-proton-sample-lambda-crud-service>

から確認ください。

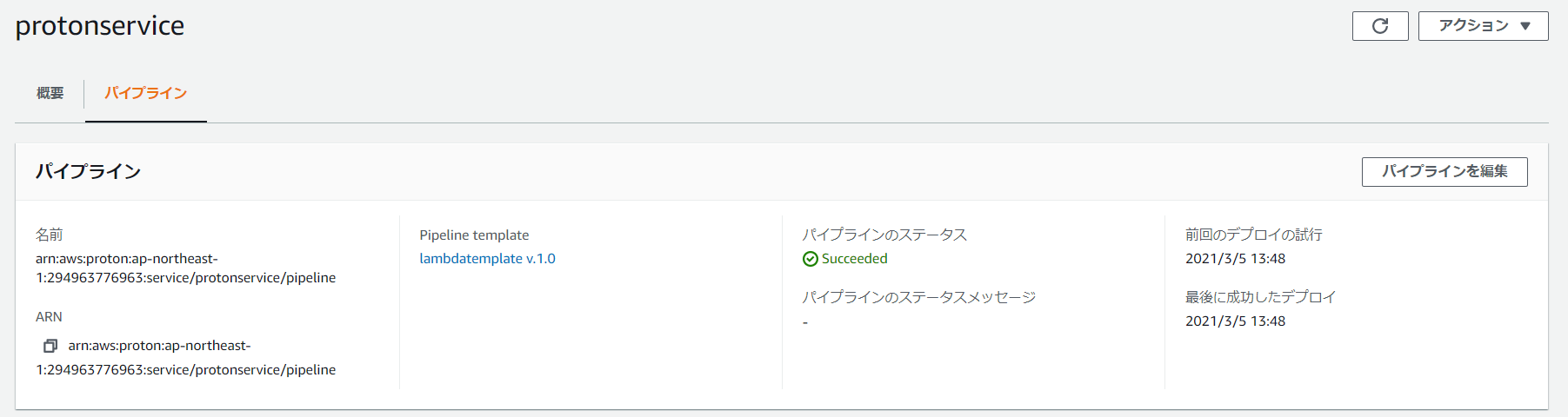
1. 環境、サービスの更新
   1. 今まで構築された環境及びサービスで誓われているCloudFormationテンプレートは、各詳細画面右上ボタンの[アクション]→[ダウンロード仕様]から確認が可能です。
   2. 先ほどの手順ではDynamoDBのttlを設定しませんでしたが、仮に運用途中で必要になったとします。環境の詳細画面、アクションから[更新]を選びます。



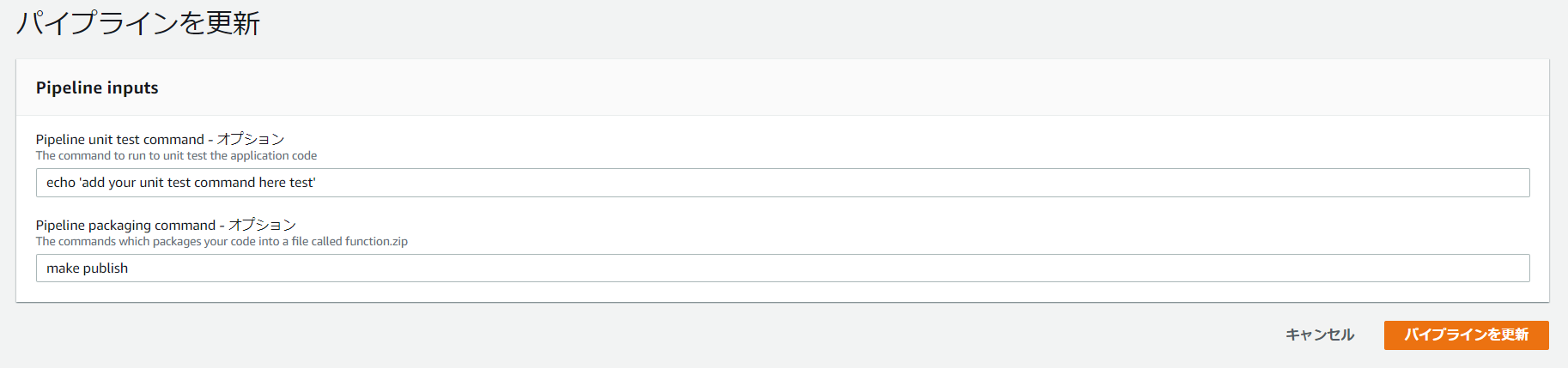
* 1. [ttl 5]と指定すると、CloudFormationスタックが再実行されDynamoDBの設定が更新されます。



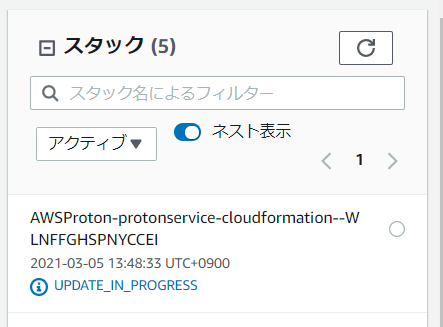
* 1. 次は、サービスを選択して、パイプラインのタブを選びます



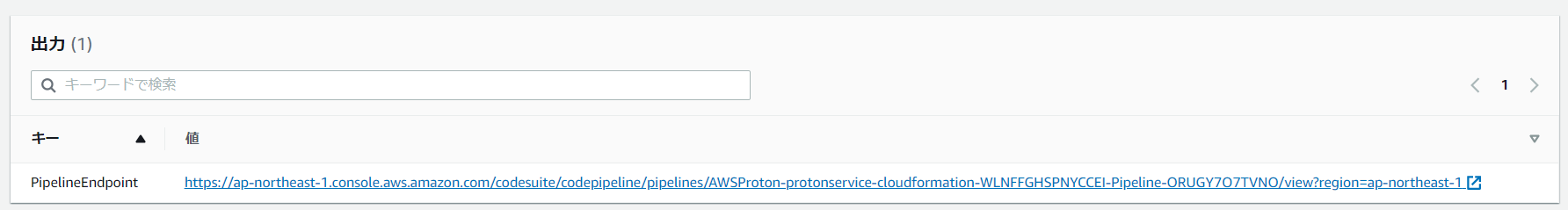
* 1. [パイプラインを編集]を押して、ユニットテストに用いる文字列を少し変更し、パイプラインを更新]を押します。（echo 'add your unit test command here'の部分です）

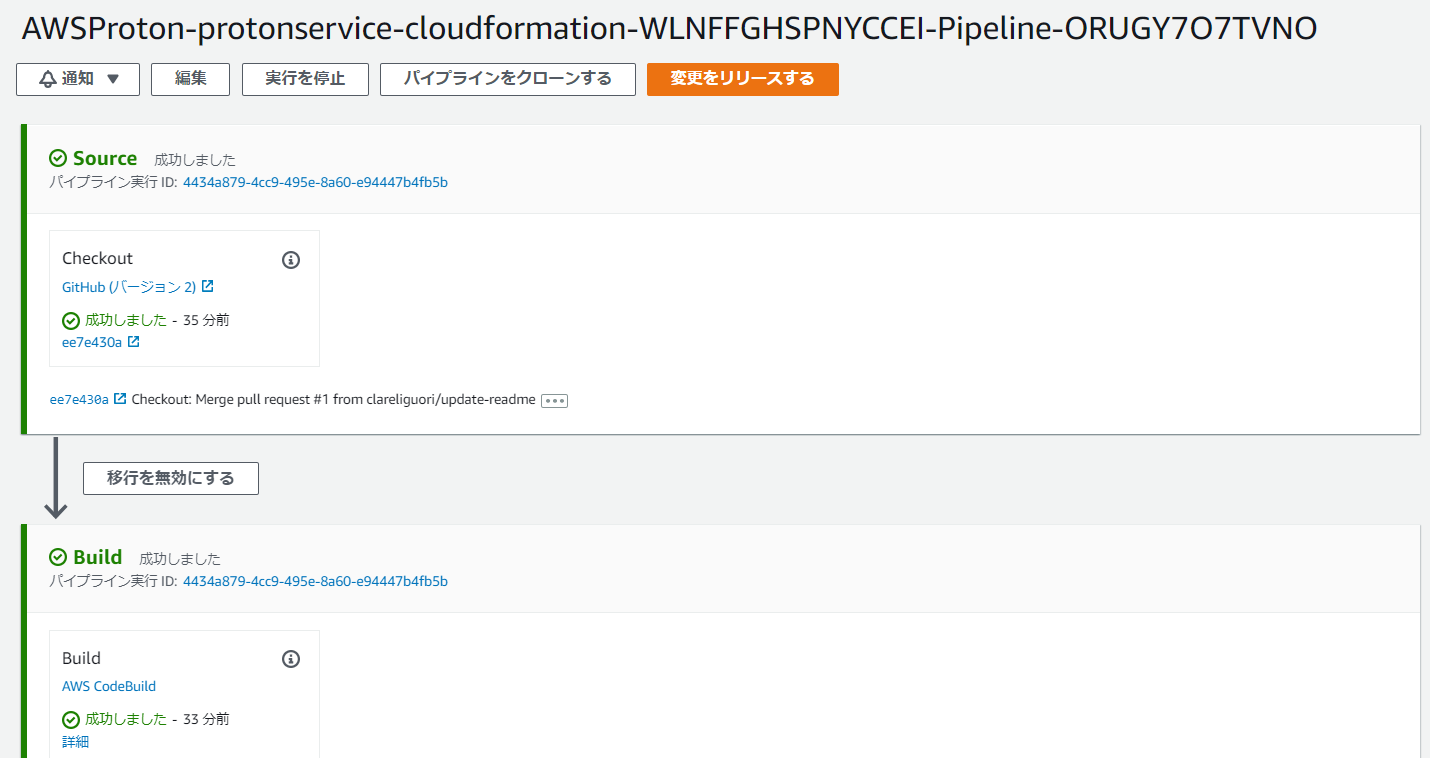


CloudFormaitonが再度Updateされます



* 1. 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





1. 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