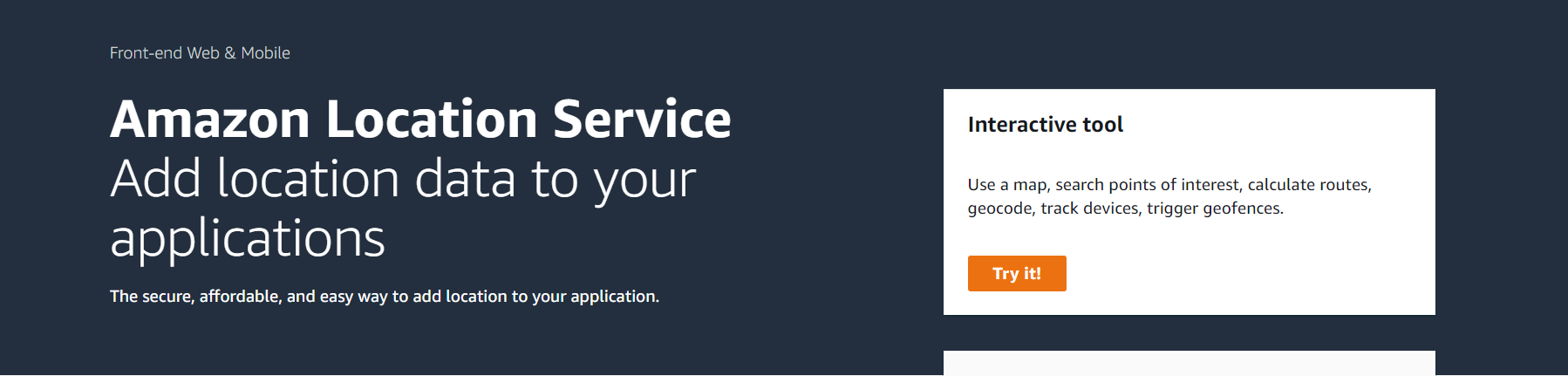
Amazon Location Service ハンズオン

2021/07/14

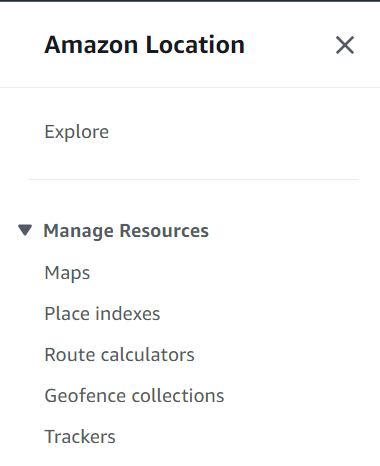
シニアエバンジェリスト　亀田治伸

【地図情報の作成】

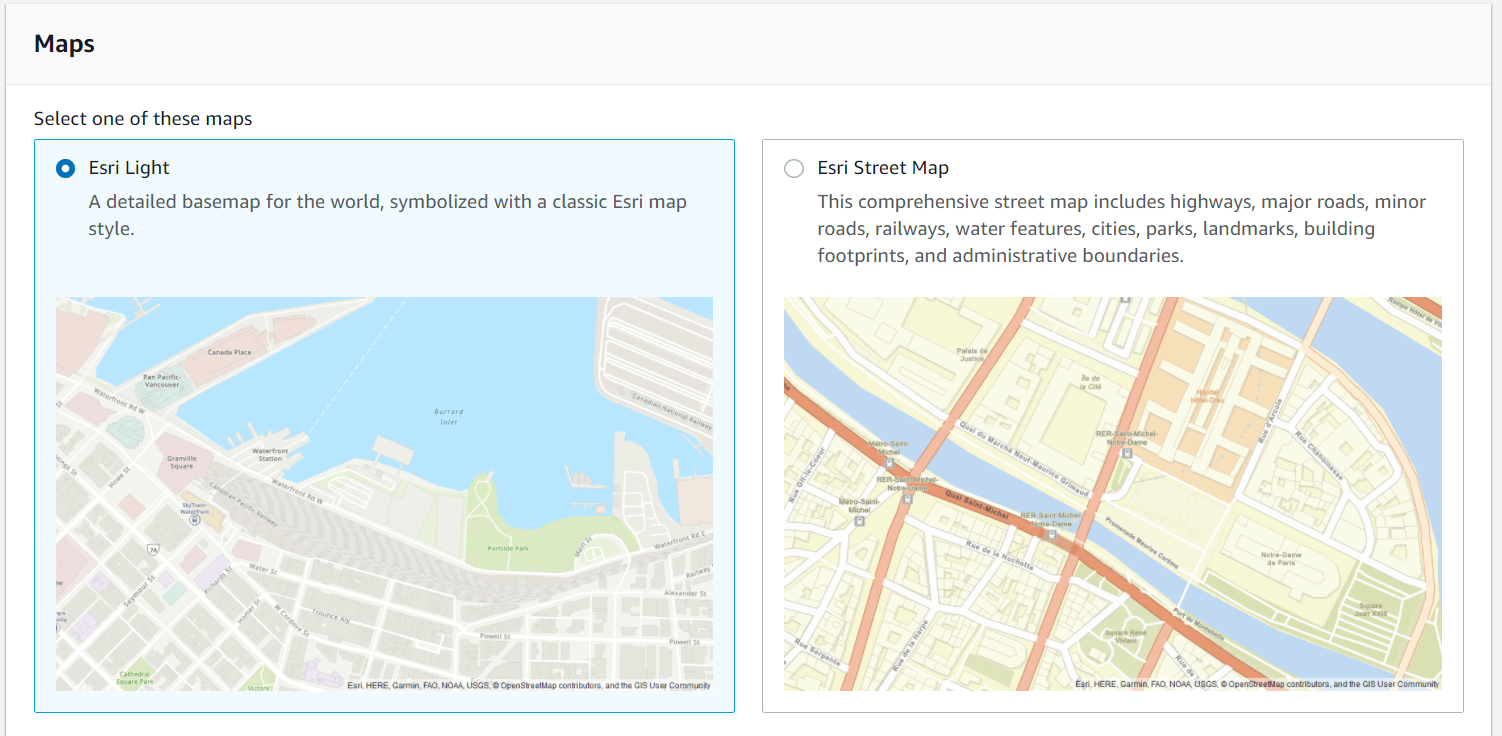
1. マネージメントコンソールでLocation Serviceにアクセスします



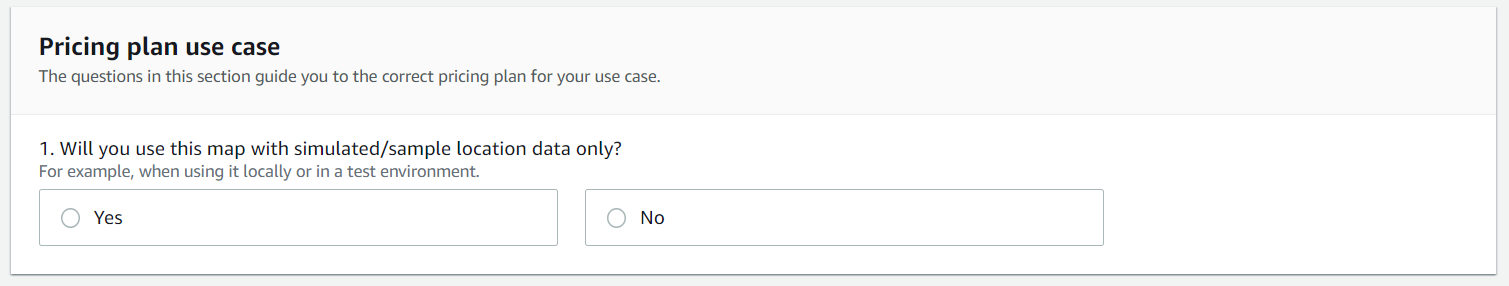
1. 左上、三本線（三）をクリックし、[Maps]をおします



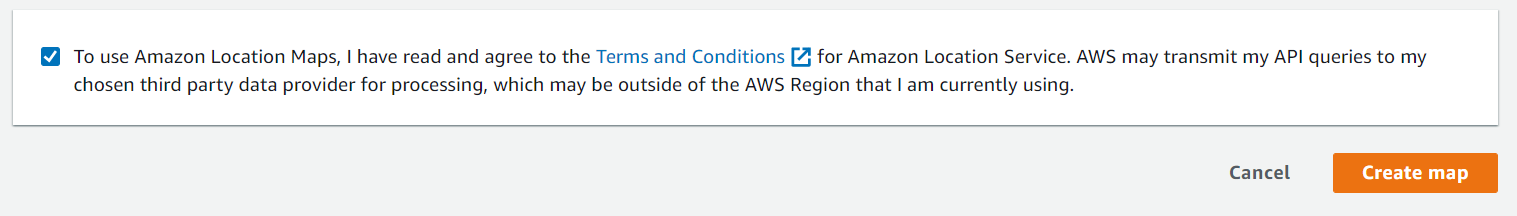
1. [Create map]ボタンをおします
2. 適当な名前をつけます。この名前は後で使うのでメモっておいてください
3. 利用するMapを選びます。種別はなんでもOKです



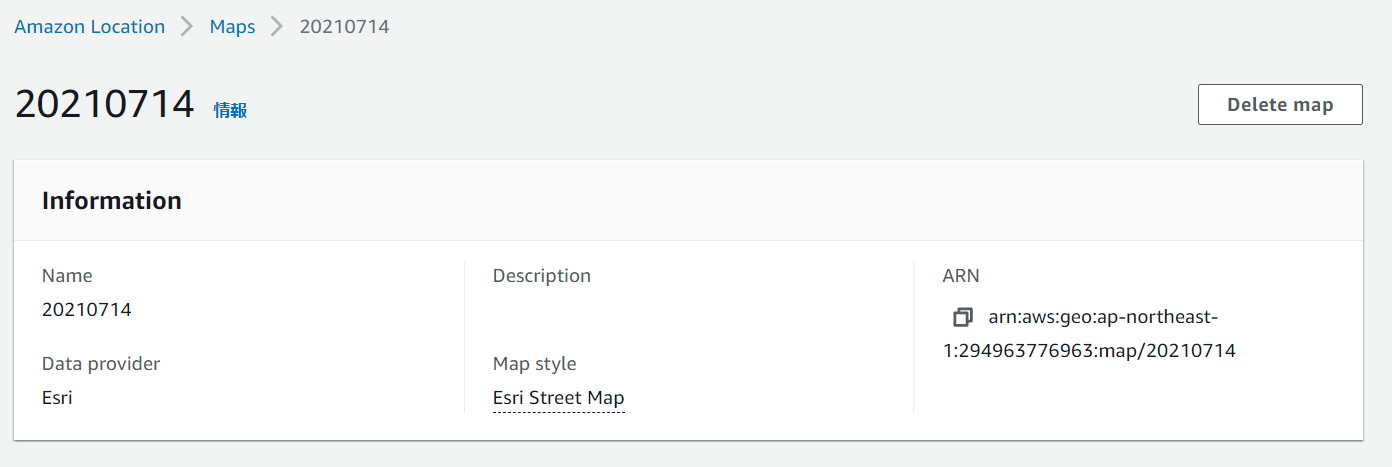
1. [Pricing plan use case]で[Yes]をえらびます



1. 以下にチェックをつけて[Create map]をおします



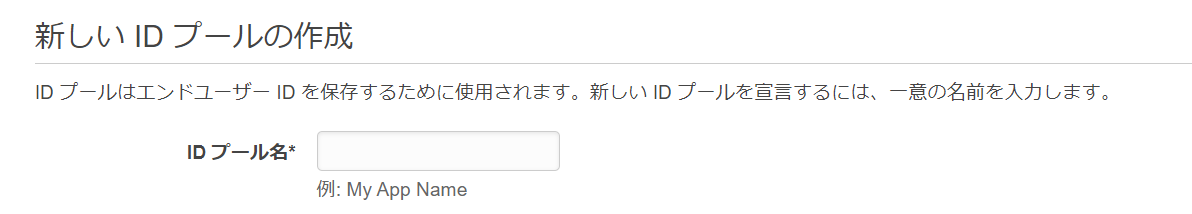
1. ARNを後で使うのでコピーしておきます



【認証基盤の作成】

以上で地図ができました。地図が埋め込まれたHTMLにアクセスするユーザーに地図へアクセスするための認証基盤を作ります。（一時的にIAMロールを引き渡す仕組みです）

1. Cognitoのマネージメントコンソールにアクセスします
2. [IDプールの管理]をおします
3. [新しいIDプールの作成]をおします
4. 適当な名前を以下につけます



1. [認証されていない ID に対してアクセスを有効にする]にチェックを付けます。これによるログイン不要でIAMロール権限が　HTMLにアクセスした人に一時的に引き渡されるようになります。



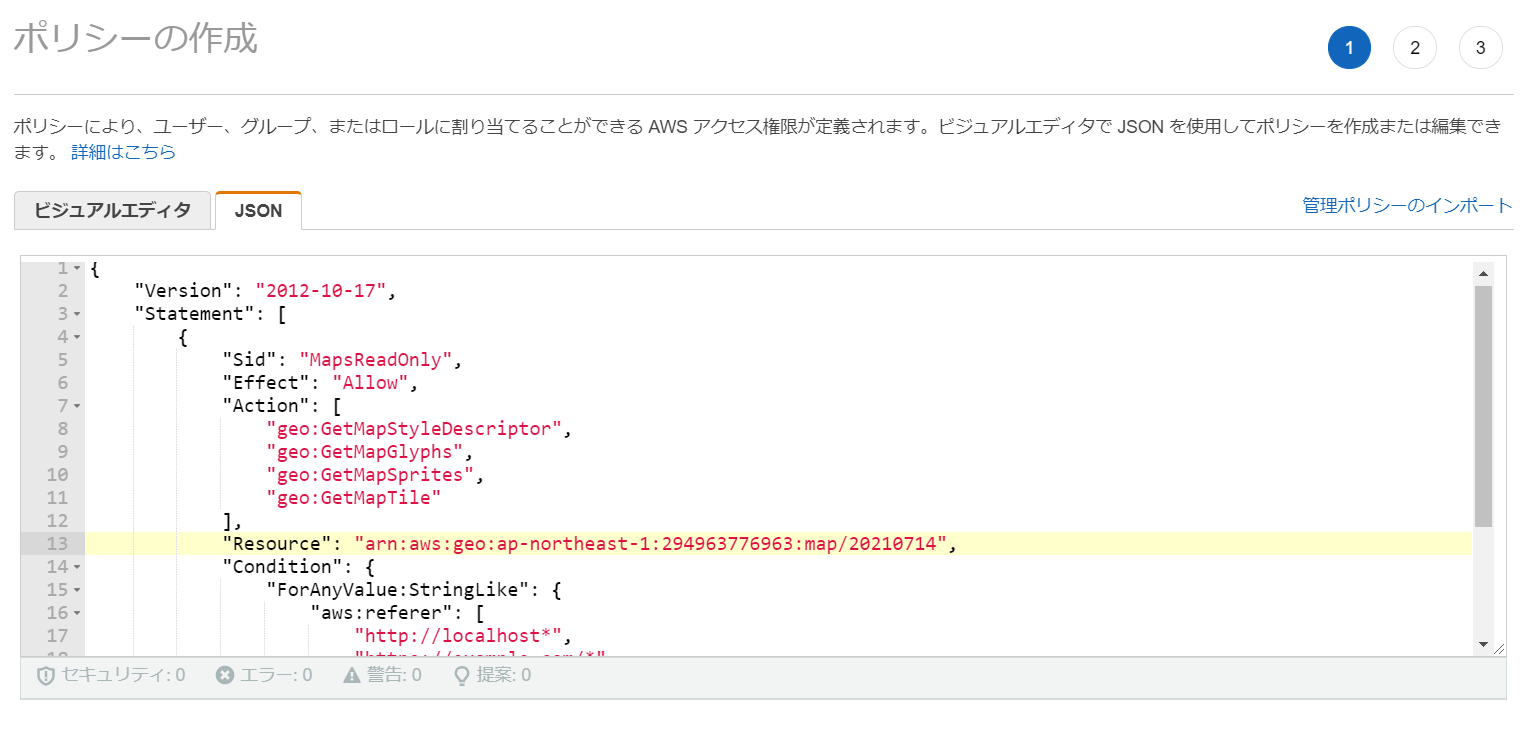
1. [プールの作成]をおします
2. [許可]をおします
3. 以下のプールのIDをメモっておきます



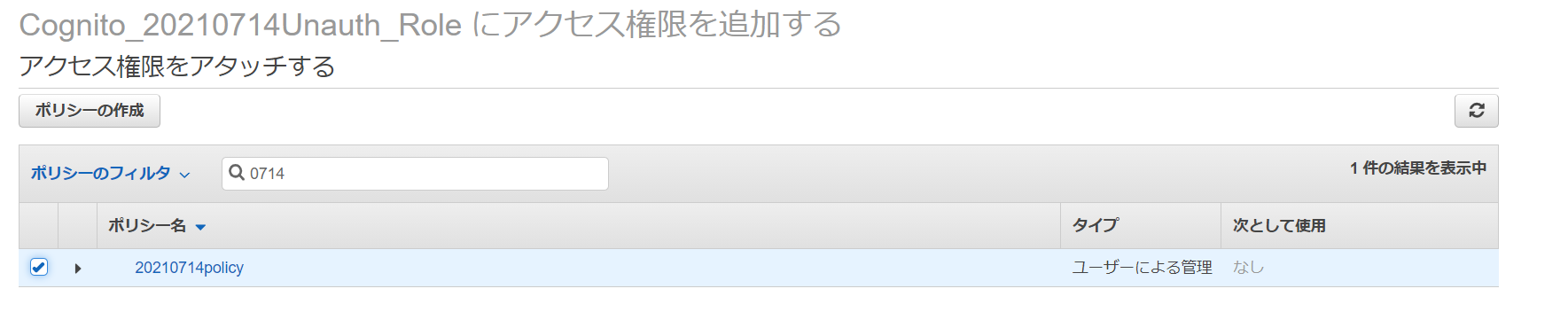
1. IAMロールのマネージメントコンソールにアクセスします
2. 先程作成したcognitoIDプールの名前で検索すると2つIAMロールができています。認証済ユーザー用と非認証一般ユーザー様です



1. このハンズオンでは誰でもアクセスできるHTML環境を作りますので、[Unauth]の方を選びクリックします
2. [ポリシーをアタッチします]をおします
3. [ポリシーの作成]おします。（ブラウザ別タブが起動します）
4. JSONタブを選び、コピペ用コマンド集の1番をコピーします
5. “Resource”の部分を先程作成したmapのARNに置き換えます



1. [次のステップ:タグ]をおし、次の画面で[次のステップ:確認]をおします
2. 適当名前をつけ、[ポリシーの作成]をおします
3. 先程のIAMロール設定のタブに戻り、先程作成したポリシーを検索から探して、チェックを付けます。出てこない場合、右上の丸い矢印マークをおすと出てきます



1. [ポリシーのアタッチ]をおします。

以上で、非認証一般ユーザーでも地図にアクセスできるようになる権限が作成されました。

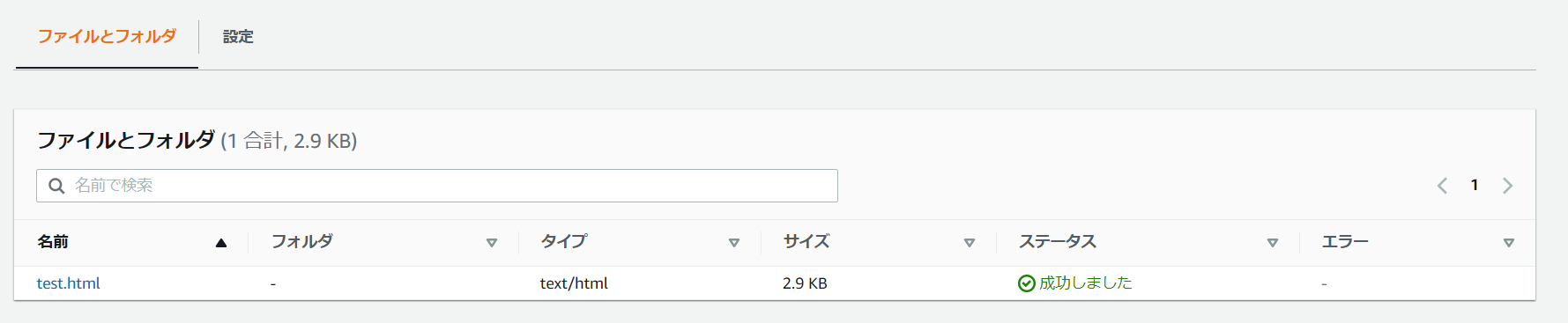
【地図埋め込みHTMLの作成とホスティング】

先程作成した地図をHTMLに埋め込み、一般ユーザーがアクセスできるように設定しS3にホスティング。

1. S3のマネージメントコンソールに移動します
2. [バケットを作成]をおします
3. 適当な名前を入力し、[バケットを作成]をおします
4. GitHubからtest.htmlをダウンロードしてエディタで開きます
5. identityPoolId に先程作成したCognitoプールIDのIDをコピペします。同様にconst mapNameに先程作成した地図の名前をいれます。（ARNではなく地図の名前です）
6. ファイルを保存し、先程作成したS3バケットのアップロードします。マネージメントコンソールでバケット名をクリックすると以下の画面が表示されます。[アップロード]ボタンをおしてください



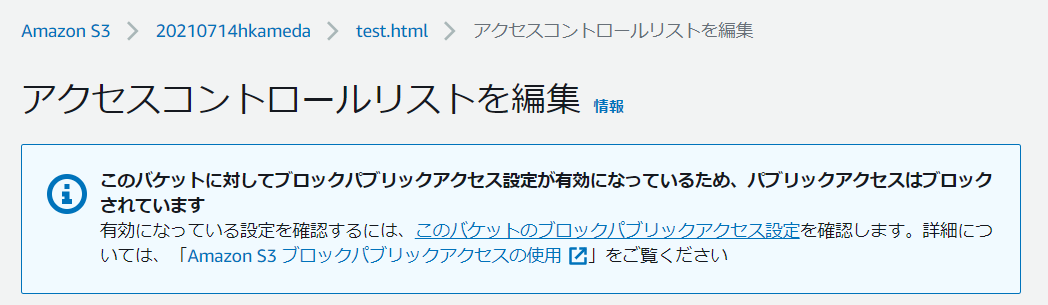
1. [ファイルの追加]をおし、ファイルが選択出来たら[アップロード]ボタンをおします
2. アップされたファイルをクリックします



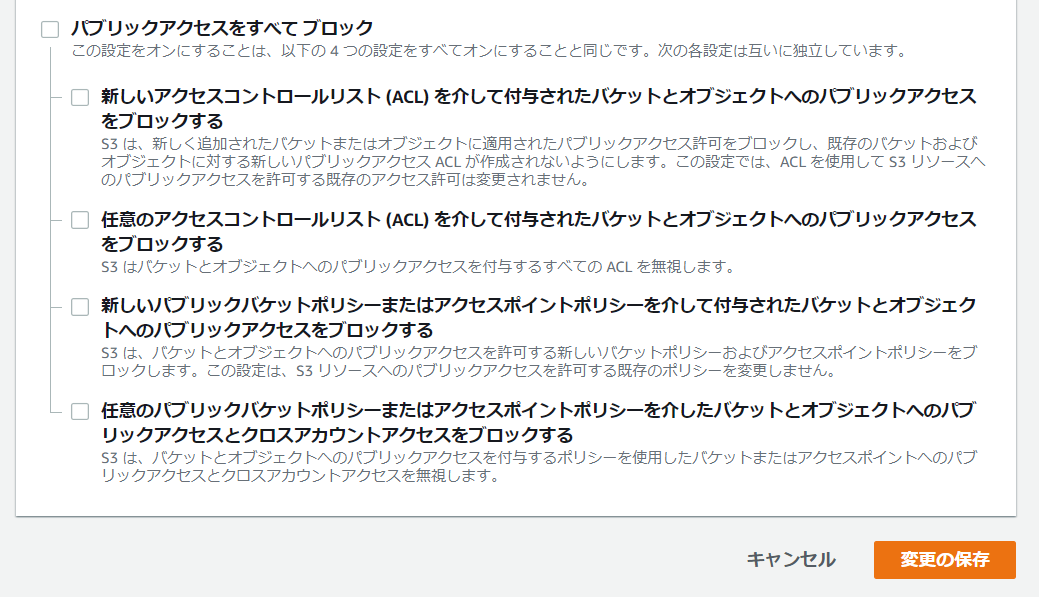
1. [アクセス許可]のタブをおします



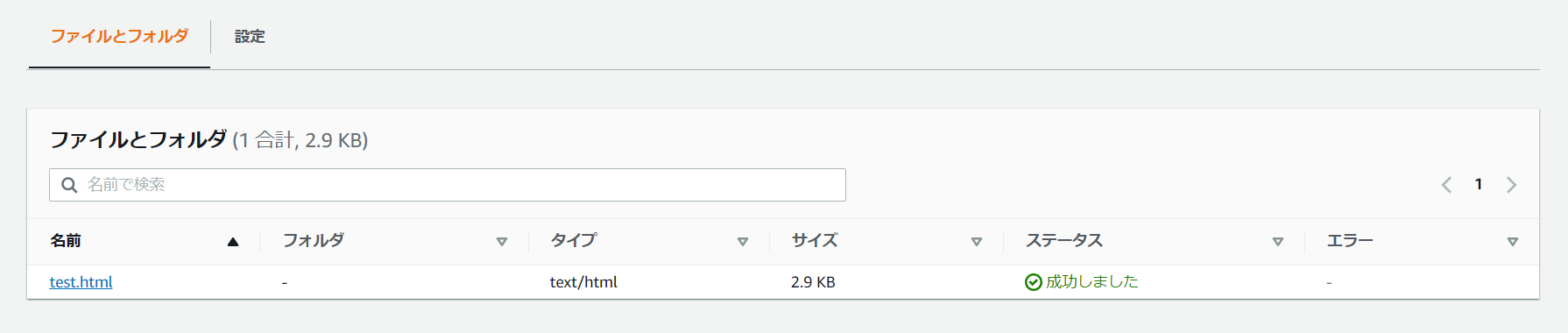
1. [編集]をおします
2. 以下の画面の[このバケットのブロックパブリックアクセス設定]をクリックします



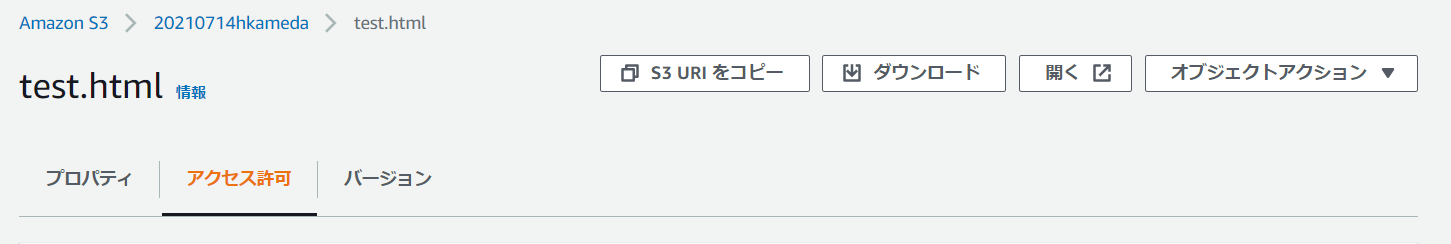
1. [ブロックパブリックアクセス (バケット設定)]の[編集]ボタンをおします
2. 以下の様にチェックを外して[変更の保存]をおします



1. 次の画面で[確認]と入力し[確認]ボタンをおします
2. 再度バケットの中にあるtest.htmlをクリックします



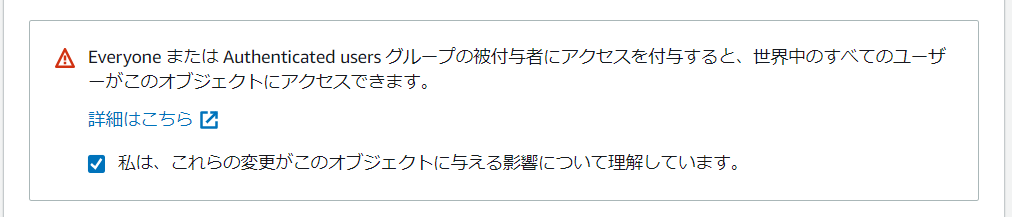
1. [アクセス許可]をタブを選びます



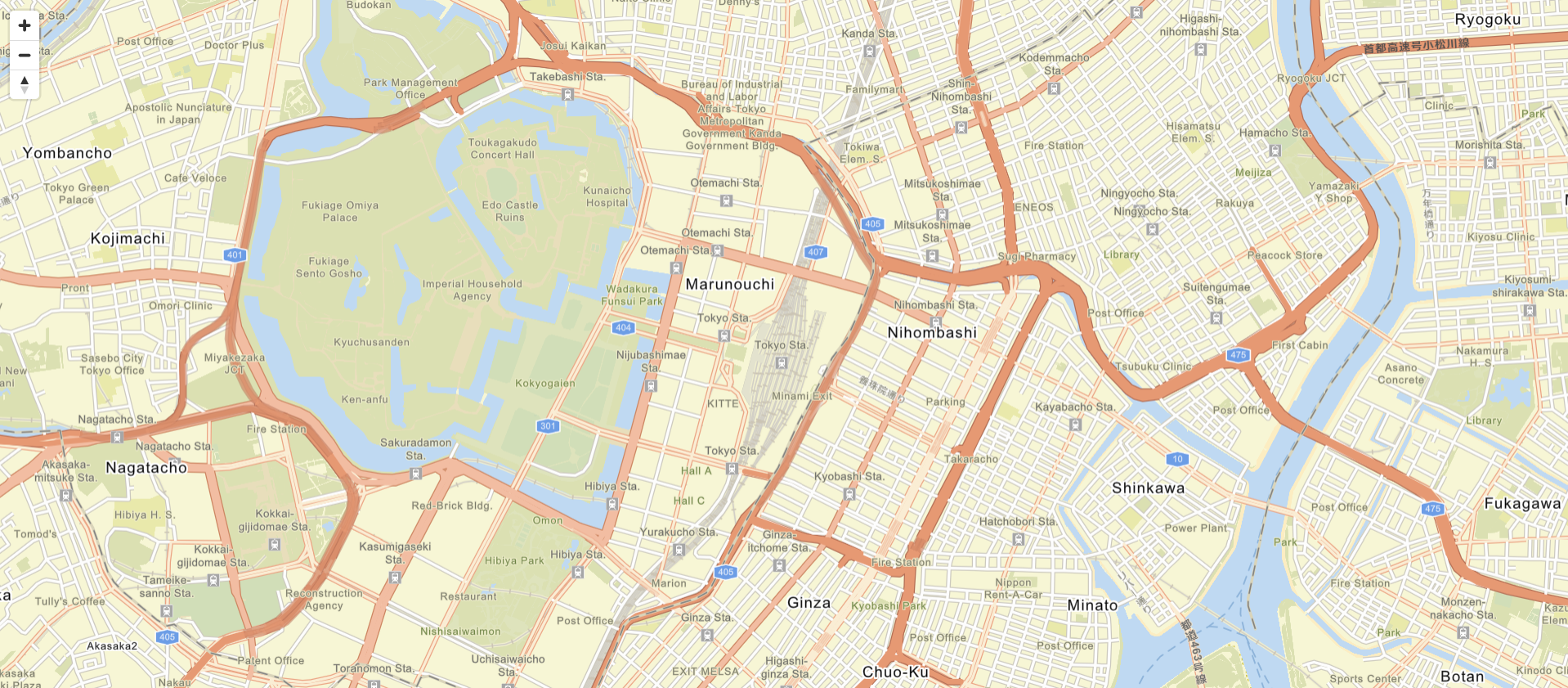
1. [編集]ボタンをおします
2. 全員 (パブリックアクセス)の設定を以下のようにします。



1. 注意事項にチェックをつけ、[変更の保存]をおします



1. 以下のように地図が表示されます



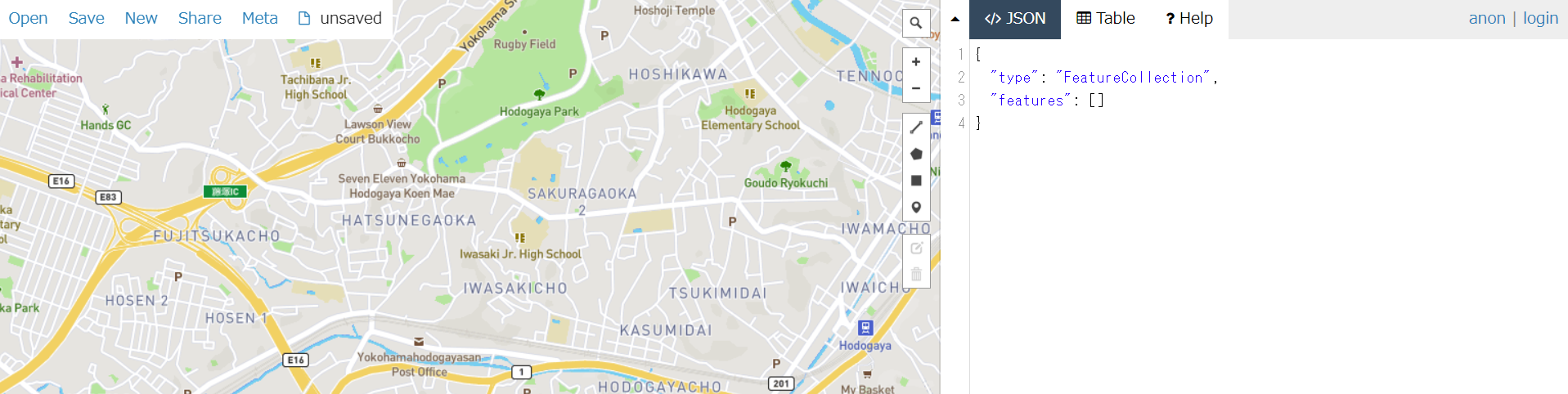
真っ白な画面が出る場合test.htmlの修正をミスしていると思われるので、再度確認してみてください。

【Geo Fence機能の実装】

Location ServiceにはGPSをベースとして任意の区画にデバイスが[Exit]した、もしくは[Enter]したというイベントを取得することができます。

AWS IoT Core経由でデバイスからGPS情報がMQTTで挿入され、それをLambdaでLocation Serviceに送信し、イベントをCloudWatch Logsに出力する環境を作ります。

1. <https://geojson.io/>　にアクセスします



1. 自分の最寄駅をさがします
2. 駅が見つかったら以下のボタンをおしてピンを落とします



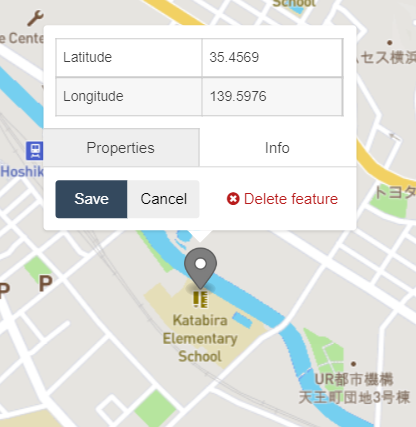


1. 画面右にGPS情報が出てきますので、coordinatesで表示される座標をメモリます



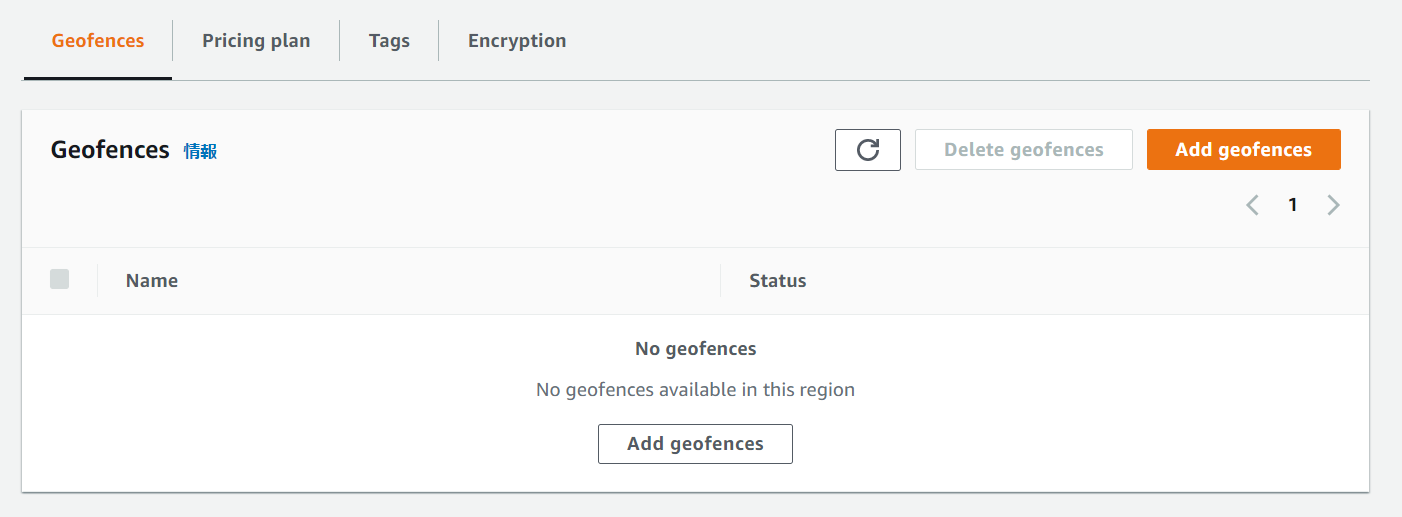
1. 地図上のピンをおして[Delete feature]をクリックし、ピンをけします
2. 今度は5角形のマークをおして、駅周辺に領域を作成します



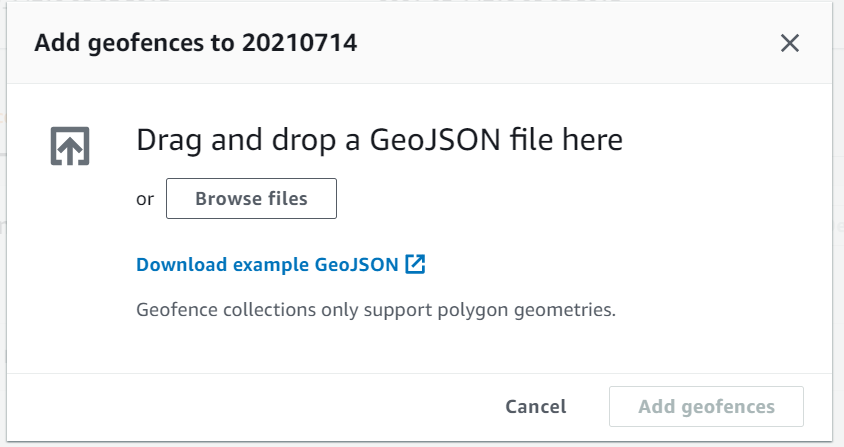
1. 画面右側に表示されるJSONをコピーし、[geo.json]として保存してください
2. 次に、領域の外にある場所のどこかでピンを再度落とし、GPS情報をメモって下さい
3. Location Serviceのマネージメントコンソールに戻り左のペインから[Geofence collections]をクリックします



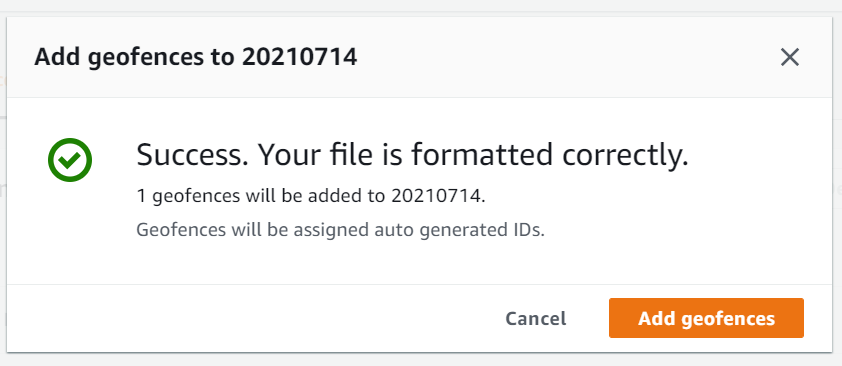
1. [Create geofence collection]をおします
2. 適当な名前をつけ、[Pricing plan use case]でYesを選び、[Create geofence collection]をおします
3. [Add geofence]をおします



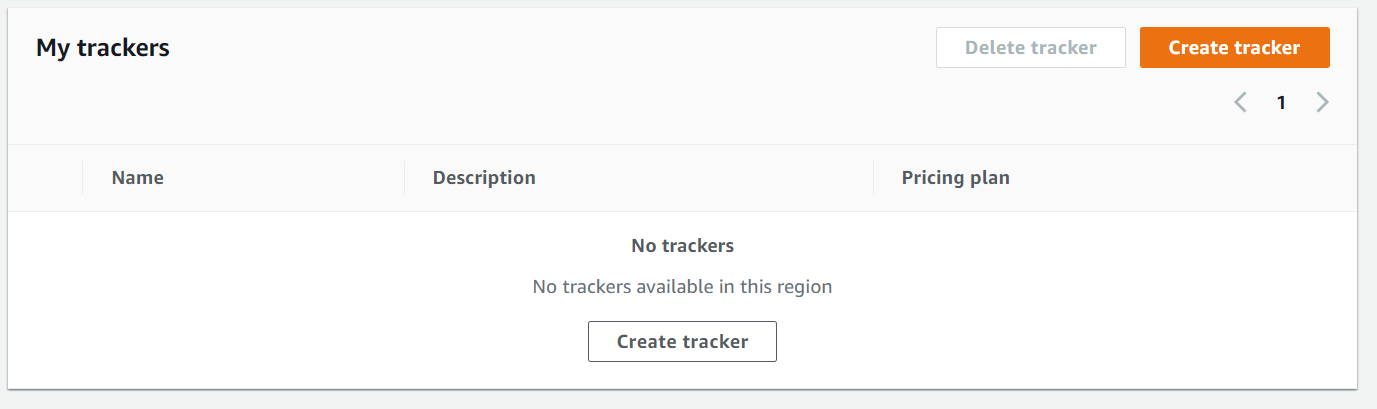
1. 先程作成したgeo.jsonをアップロードします



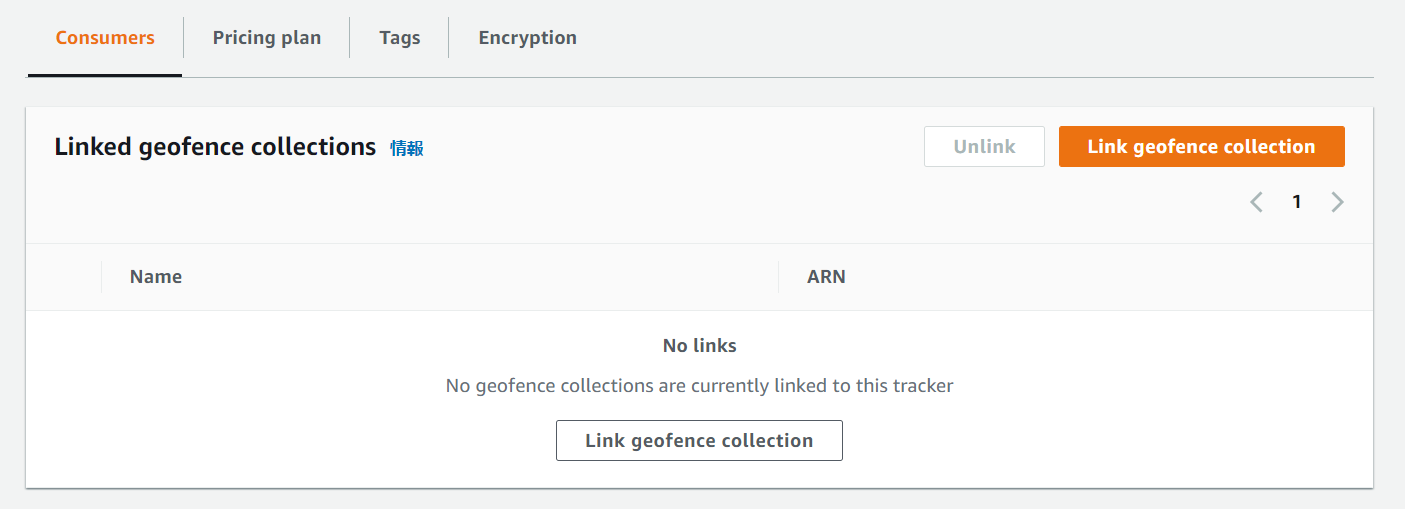
1. ファイルが正しければ以下が表示されますので[Add geofences]をおします



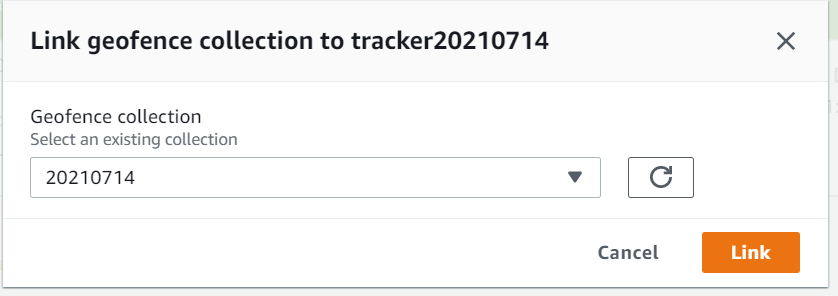
1. これで最寄駅を周辺とした領域をLocation Serviceが認識しました。次にGPSデバイスと連携させるために、左ペインから[Trackers]をクリックします



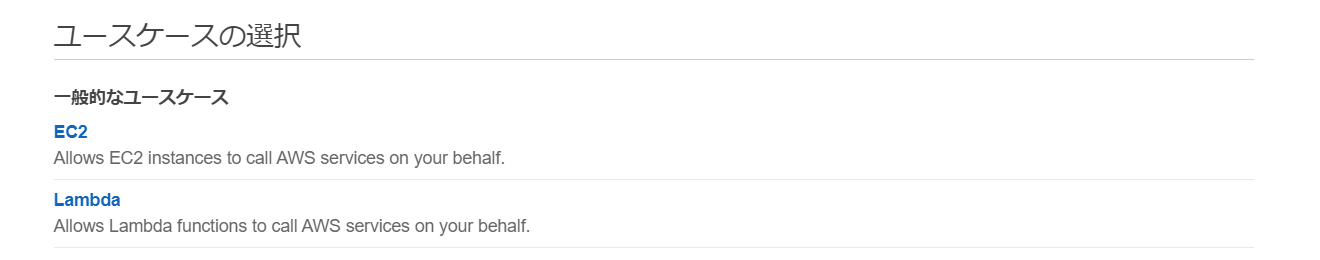
1. [Create trackers]をおします
2. 適当な名前をつけ、[Pricing plan use case]でYesを選び、[Create tracker]をおします
3. [Link geofence collection]をおします



1. 先程作成したgeofenceを選んで[Link]をおします



1. これで特定のTrackerが特定のGeo fenceと連携するようになりました。Trackerには複数のGeo fenceを連携させることも可能です。これからの手順ではTrackerに対してAWS IoT CoreからMQTTベースでGPS情報が送られるように設定をおこないます。
2. まずLambda関数にアタッチするIAMロールを作成するために、IAMのマネージメントコンソールにアクセスします
3. 左のペインから[ロール]をクリックします
4. [ロールを作成]をおします
5. 以下でLambdaを選び、[次のステップ:アクセス権限]をおします



1. [Administrator Access]を選び、[次のステップ:タグ]をおします



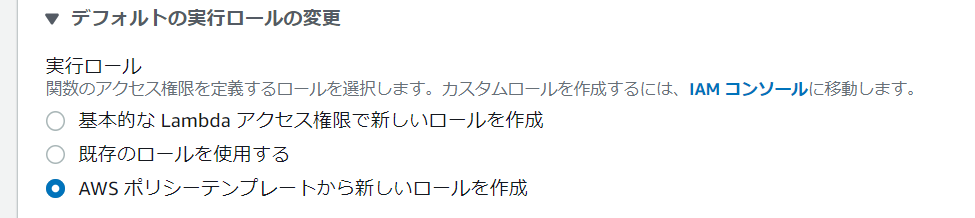
注意：本来はLocation Serviceのみのアクセス権でいいのですが、2021年7月時点でビルトインのLocation Service用ポリシーが要されておらず、そちらの作成は手順が煩雑となるため、このハンズオンでは割愛します。商用環境におけるAdmin権限はお勧めしませんので、詳しくは以下を参考にしてください。

https://docs.aws.amazon.com/ja\_jp/location/latest/developerguide/security-iam.html

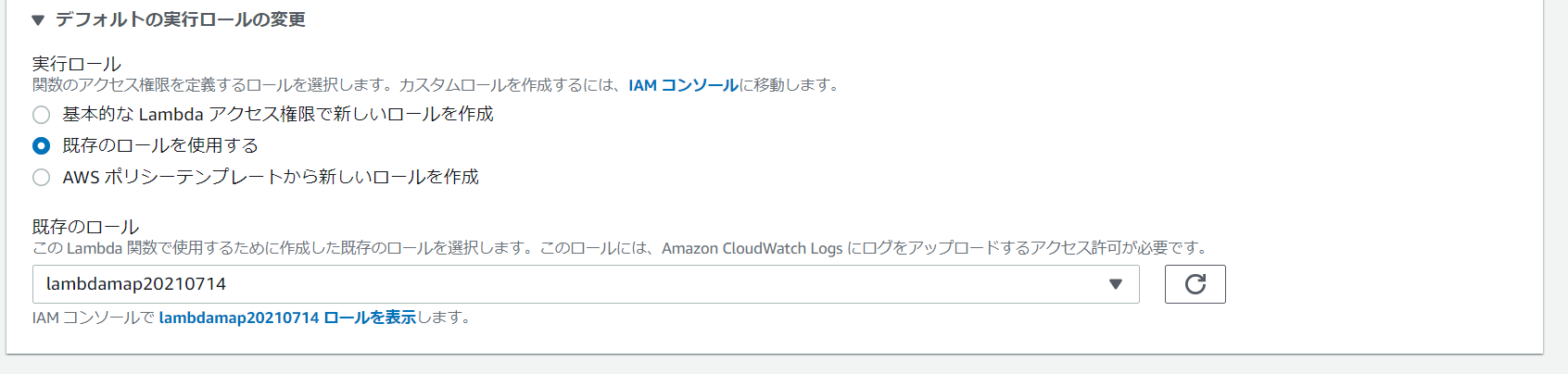
1. 次の画面ではそのまま[次のステップ:確認]をおします
2. 適当な名前をロールにつけ[ロールの作成]をおします
3. 次にLambda関数を作成します。この関数はAWS IoT Coreから上がってきたGPS情報をLocation Serviceに渡す役割をにないます。Lambdaのマネージメントコンソールにアクセスします
4. [関数の作成]をおします
5. 適当な名前を関数につけ、ランタイムにはPython3.8を選びます



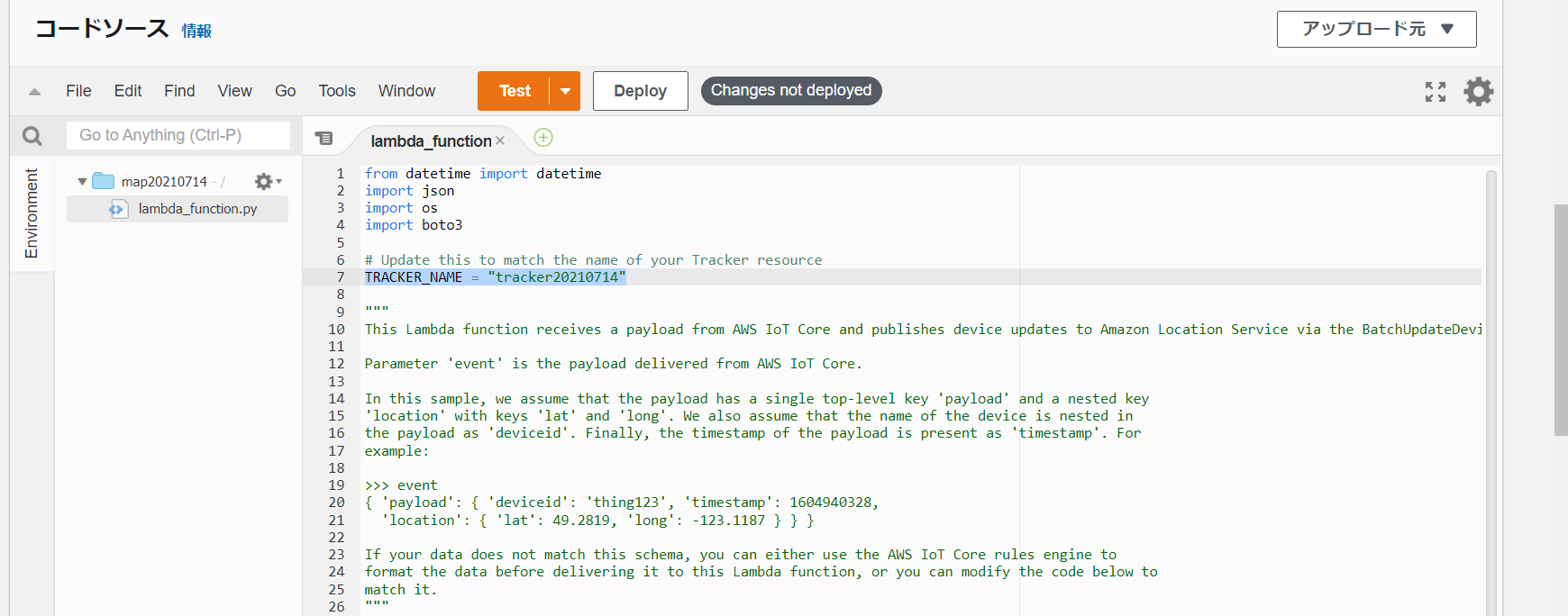
1. [デフォルトの実行ロールの変更]をクリックし、[既存のロールを使用する]を選びます



1. 先程作成したIAMロールを選びます、[関数の作成]をおします



1. GitHubからlambda.txtをダウンロードし、その中身をコードに貼り付けます
2. TRACKER\_NAMEの部分を先程作成したTrackerの名前に置き換えます



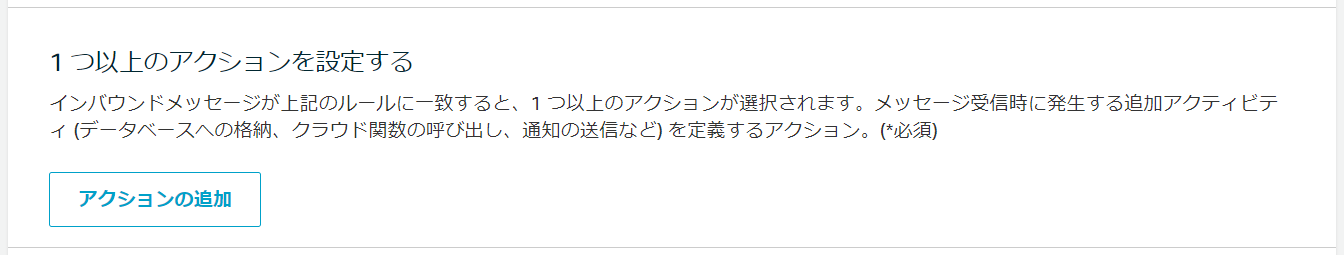
1. [Deploy]をおします
2. 次にIoT Coreのマネージメントコンソールにアクセスします
3. 左ペインから[ACT]→[ルール]をクリックします



1. [作成]をおします
2. 適当な名前をつけ、[ルールクエリステートメント]を以下で置換します

**SELECT \* FROM 'iot/location‘**

1. [1 つ以上のアクションを設定する]の[アクションの追加]をおします



1. [Lambda]を選び、[アクションの設定]をおします



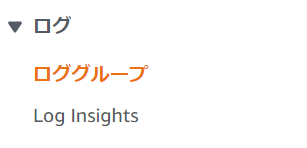
1. 先程作成した関数を選び[アクションの追加]をおします
2. [ルールの作成]をおします

ここまでの手順で、IoT CoreからMQTTベースて上がってきたGPS情報がLambda経由でLocation ServiceのTrackerに渡され、TrackerとLinkされているGeo fence情報と突合されその結果がCloudWatch Logsに出力されるようになりました。

先程作成したLambdaの関数画面に行くと正しくIoT Coreがイベントソースとして設定されていることがわかります

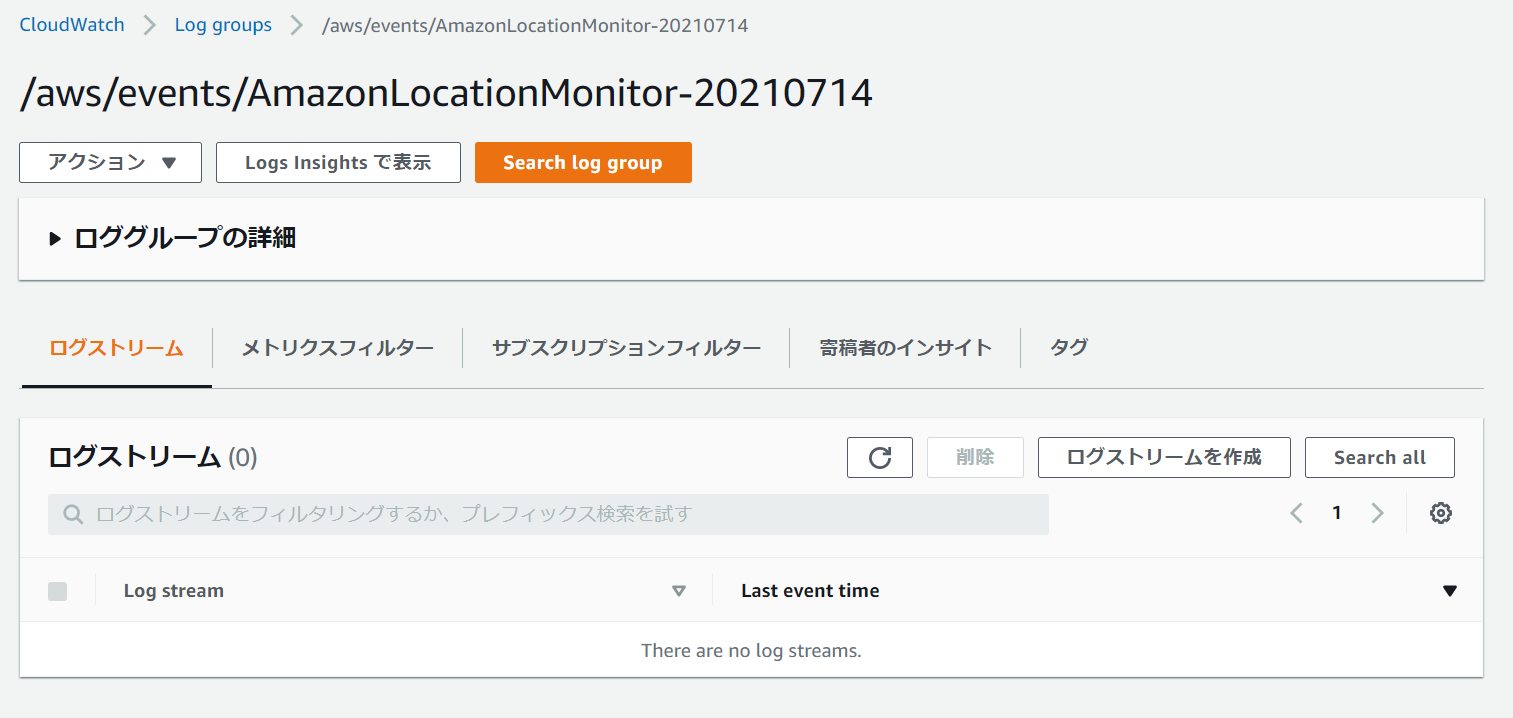


1. テストを行う前にCloudWatch Logsを開きます。CloudWatchマネージメントコンソールの左ペインから[ロググループ]をクリックします



1. [location]で検索すると今日作成したロググループが出てきますので、クリックしておきます





今の時点ではログは何もふくんでいません。この画面は開きっぱなしにしておいてください。

1. いよいよテストです。IoT Core左ペインから[MQTTテストクライアント]をクリックし、[トピックに公開する]タブを選んでください



1. トピック名は[**iot/location**]と入力します
2. コピペ用コマンド2番の内容を別のエディタにコピーし、先程取得した最寄り駅のGPS情報に置き換えます。置き換えたものを[メッセージペイロード]]に貼り付け、[発行]をおします
3. Location ServiceはデバイスがGeo fenceの中にいることを認識しました。次にtimestampの値を10増やし、geo fenceの外にあるGPS情報に置き換え、再度メッセージを発行してください
4. ClouudWatch Logsを確認すると2つのログがでています

EnterとEXITです





おつかれさまでした！ハンズオンは以上です

以下を削除してください

Location Service

Trackers

Geofence collection

Maps

Lambda関数

S3バケット

IAMポリシーとロール

AWS IoT Core

ルール