# AWS Free Tier Auto-Shutdown Project (Zero Billing Goal)

## 📊 Project Goal

Automatically detect AWS usage costs and immediately stop or delete services to **ensure no bill is generated**, staying strictly within the AWS Free Tier.

## 🔧 Free Tier Services Used

| AWS Service | Usage Purpose | Free Tier Eligible? |
| --- | --- | --- |
| AWS Lambda | Execute cleanup logic | ✅ Yes (1M calls/month) |
| Amazon CloudWatch | Schedule and monitor | ✅ Yes (basic use) |
| AWS Budgets | Detect billing threshold | ✅ Yes |
| AWS IAM | Role and permissions | ✅ Yes |
| Amazon S3 | Logging (optional) | ✅ Yes (5GB) |
| EventBridge | Scheduled rule to run Lambda | ✅ Yes |
| Amazon SNS | Trigger Lambda from Budget | ✅ Yes |

## ⚖️ Billing Strategy

* **AWS Budget Threshold**: $0.01 (lowest allowed)
* **Trigger Lambda cleanup** immediately when $0.01 is reached
* **No resource runs long enough to generate a bill**

## ✅ Step-by-Step Setup

### ✅ Step 1: IAM Role for Lambda

* Go to IAM > Roles > Create Role
* Use-case: **Lambda**
* **Role Name**: FreeTierAutoShutdownRole
* **Description**: “Grants Lambda permission to stop, delete, or clean up AWS services when budget threshold is reached to prevent billing.”
* Attach policies:
  + AmazonEC2FullAccess
  + AmazonS3FullAccess
  + AmazonDynamoDBFullAccess
  + AmazonAPIGatewayAdministrator
  + AWSLambda\_FullAccess
  + AWSBudgetsActionsWithAWSResourceControlAccess
  + IAMReadOnlyAccess *(to log IAM usage only)*
  + AmazonSNSFullAccess

### ✅ Step 2: Create AWS Budget

* Go to **Billing > Budgets > Create Budget**
* **Type**: Cost Budget
* **Amount**: $0.01
* **Threshold**: 100% of actual cost
* **Notifications**: Enable **Amazon SNS Alerts**
* **SNS Topic ARN**: Provide the ARN for the SNS Topic created in Step 3

⚠️ Make sure the SNS topic exists and permissions are correctly set

### ✅ Step 3: SNS Topic

* Go to **SNS > Create Topic**
* **Name**: BudgetAlert
* Copy the topic ARN for later use

#### ➤ Set SNS Topic Access Policy

Navigate to SNS > Topics > BudgetAlert > **Edit Access Policy** and replace with:

{  
 "Version": "2008-10-17",  
 "Id": "\_\_default\_policy\_ID",  
 "Statement": [  
 {  
 "Effect": "Allow",  
 "Principal": {  
 "Service": "budgets.amazonaws.com"  
 },  
 "Action": "SNS:Publish",  
 "Resource": "arn:aws:sns:eu-north-1:YOUR\_ACCOUNT\_ID:BudgetAlert"  
 }  
 ]  
}

✅ Replace YOUR\_ACCOUNT\_ID with your AWS Account ID.

#### 📬 Subscribe Lambda to SNS

1. Go to SNS > Topics > BudgetAlert
2. Select **Create Subscription**
3. Protocol: Lambda
4. Endpoint: Choose your Lambda function

Ensure Lambda has sns:Subscribe permissions

### ✅ Step 4: Lambda Function for Cleanup

1. Go to **Lambda > Create Function**
2. **Name**: AutoCleanupLambda
3. Runtime: Python or Node.js (your preference)
4. Assign the **IAM Role**: FreeTierAutoShutdownRole

Paste cleanup logic (example Python):

import boto3  
  
def lambda\_handler(event, context):  
 ec2 = boto3.client('ec2')  
 ec2.stop\_instances(InstanceIds=['i-xxxxxxxxxxxxxxxxx'])  
  
 apigateway = boto3.client('apigateway')  
 for api in apigateway.get\_rest\_apis()['items']:  
 apigateway.delete\_rest\_api(restApiId=api['id'])  
  
 s3 = boto3.resource('s3')  
 for bucket in s3.buckets.all():  
 for obj in bucket.objects.all():  
 obj.delete()  
 bucket.delete()  
  
 dynamodb = boto3.client('dynamodb')  
 for table in dynamodb.list\_tables()['TableNames']:  
 dynamodb.delete\_table(TableName=table)  
  
 return {  
 'statusCode': 200,  
 'body': 'Resources cleaned up.'  
 }

Customize with your resource IDs or dynamic listing as needed.

### ✅ Step 5: Schedule Cleanup Rule (Optional)

* Go to **EventBridge > Rules > Create Rule**
* **Type**: Schedule
* **Frequency**: cron(0 \* \* \* ? \*) (every hour)
* **Target**: Lambda function

## ❌ Preventing Hidden Costs

| Service | Caution |
| --- | --- |
| Elastic IPs | Charges when unattached |
| CloudWatch Logs | Never expire by default (set TTL) |
| NAT Gateway | Always billed, avoid entirely |
| Snapshots/Backups | May persist after EC2 termination |

## 📊 Monitoring & Summary

* Monitor costs via **Cost Explorer**
* Test Lambda manually before enabling automation
* Keep CloudWatch logs short-lived

### 🗕️ Summary Table

| Resource | Action | Triggered By |
| --- | --- | --- |
| EC2 | Stop | Lambda + SNS |
| Lambda | Delete | Lambda |
| API Gateway | Delete | Lambda |
| S3 | Empty/Delete | Lambda |
| DynamoDB | Delete | Lambda |
| IAM | Log Only | Optional |
| Lex | Manual Delete | Advanced Use Case |

## 📁 Output

* PDF available
* GitHub Repository:
  + Lambda script
  + Setup guide (README)
  + IAM policy templates
  + CloudFormation template (optional)

## 🗖️ Final Notes

This setup ensures **zero billing** from AWS by proactively cleaning up free-tier-exceeding resources and monitoring usage every hour or at first sign of cost.

Using **SNS + Lambda**, you have complete control over the cleanup logic. This method is highly recommended for reliability and flexibility.