SCHOOL OF ARCHITECTURE, COMPUTING & ENGINEERING

CLOUD BASED PET SECURITY SYSTEM

STUDENT NAME: SREENIVAS PALEPU

STUDENT NUMBER: U2013209

MODULE LEADER: GAURAV MALIK

MODULE CODE: CN7026

TABLE OF CONTENTS

List of Figures

List of Table

Abstract: The aim of this paper is to develop a cloud based pet security system. The communication between human and the physical devices to address the real-world problems is gaining more and more attention in recent times because of the advancements in Internet of things a cutting-edge technology. The Iot stills extends its features by integrating with cloud based solutions to process and analyze the data collected from the edge devices and to gain insights. The proposed architecture collects data from Iot edge devices called as sensors in real-time from Pets and transmit the data to the nearby Iot device. The Iot device in turn push the data to the cloud based system for real-time processing and if something is sensed wrong with the pet an immediate action must be provided like warning the owner with the alarm or text. Another use case of this data collection is to process the data and convert them into dataset and analyze them over a period of time to gain insights about the pet behavior.

1. Introduction

Pet Security system aims to focus on pets based on day-to-day activities and behaviors. The proposed architecture develops a cloud based solution to collect the data from pet and store in the cloud platform for future analysis via IoT. The study focus to improve the performance of pet wearables sensors which should be easy and comfortable for the pet to wear it and also effective to record its day–to-day changes. The use cases of Pet Security systems are

1. To Monitor the day-to-day activities of the pet
2. A micro sensor connected with Iot edge device
3. Track the pet based on its location
4. Periodic pattern changes that sleeping , eating and other activities of Pet
5. Response of the pet to strangers and also other emotional studies
6. To safeguard the pets from intruders and other external threats

The pet security system with cloud based solution proves a simple example with the following scenario, most of the pet respond to the owner in a very friendly way at most of the times. So the emotional graph stays constant throughout the day. But this is not the true case, there will be so many changes during sleep or when the pet is happy which will less tracked and analyzing the data over the period of time proves best results.

This project is simple or sample architecture to implement in real-time to analyze and improve pet security systems in market. An enhanced plan is written to focus the various aspects of the architecture from planning to development and deployment. The architecture is implemented using AWS services. The paper also discuss fairly about the functional and non-functional requirements of the system, then proceeds to with cloud architecture, datacenter needs.

2. Project Plan

The Project plan includes the scope of the project and various milestones that are encountered during various phases of development.Table.1 shows the detailed project plan

S. No	Task Name	Duration (Days)	Planned Start Date	Actual End Date	Actual (Days)	Completion %
1	Requirement Definition	9			9	100%
1.1	Problem Identification	2			2	100%
1.2	Requirement Gathering	5			5	100%
1.3	Requirement Review	1			1	100%
1.4	Requirement Sign-Off	1			1	100%

2	Analysis and Design	15			15	100%
2.1	Requirement Analysis	2			2	100%
2.2	Requirement Clarification	3			3	100%
2.3	High Level Architecture	3			3	100%
2.4	Detailed Designing	3			3	100%
2.5	Technical Specification	1			1	100%
2.6	Design Review	2			2	100%
2.7	Design Sign-Off	1			1	100%
3	Build-Pet Security System	20			7	35%
3.1	Aws IoT Core	3			3	100%
3.2	Aws Lambda	3			3	100%
3.4	Implement AWS Service	4				100%

Table 1 Project Planning – Pet Security System

3. Functional Requirement

Table 2 describes the Functional requirements to implement the Pet Security System.

Req ID	Functional Requirement	Description
1	Pet Sensor data	
1.1		The sensor should collect proper metrics from the pet
1.2		Should able to record various activities of the pet throughout the day
1.3		Alarm the Pet owner if the pet is lost
2	AWS IoT Core	
2.1		Collect data from the Iot Edge device
2.2		Store the data in the appropriate Database
2.3		Process the data in the cloud environment if needed
3	Tracking Pet	
3.1		Track the pet if the pet is lost
3.2		Analyze the behavioral changes for different activities
4	Analyzing Pet data	
4.1		Process the data over a period of time to gain insights from the pet activity
4.2		Able or provide better solution for pet owners to keep track and save repeatable information's about the pet

Table 2 Functional Requirement

4. Non-Functional Requirement

Non-functional Requirements build a robust and standard Pet Security System API.

Req ID	Non-Functional Requirement	Description
1	Scalability	
1.1		To handle the increasing workloads efficiently by optimizing the cloud environment
1.2		To check how best the developed system withstand of unexpected happening or increase in resource.
2	Availability	
2.1		The developed system should be ready all the time and reduce downtime.
2.2		Minimize the latency
3	Manageability	
3.1		To handle the unexpected glitches
3.2		To manage and recover faster in case of failures
3.3		Follow proper standards and methods to develop the system

Table 3 Non-Functional Requirement

5. Cloud Choice-Amazon Web Service

Amazon web services (AWS) is the core of the Pet security system and its important features are,

1. No upfront , as most of the AWS services are pay-as-you go pricing model
2. Highly secure datacenter and data protection
3. Enhanced reliability, disaster management and cost efficient
4. Fault tolerant and highly availability
5. Scalability and elasticity to manage incoming network traffic

6. Choice of Datacenter

The dedicated space with server, network and storage to provide the underlying infrastructure is called as datacenter. To reduce the latency between multiple resources used to build the pet security system the key aim is to select the appropriate datacenter. Most cases the datacenter I selected near or the closest datacenter will be selected to. It is important to choose datacenters to provide high availability, to withstand disaster and to reduce latency.

REGION CODE	REGION NAME
us-east-1	US East (N. Virginia)
eu-west-1	EU (Ireland)
us-west-2	US West (Oregon)

Table 4 Data center

The operational standards for Pet Security System.

1. Uptime Institute-Operational Sustainability

2. ISO 9000 - Quality System
3. ISO 27001 - Information Security
4. EN50600-2-6 Management and Operational Information

7. Pet Security System Architecture

The architecture of the Pet Security system is shown in Figure 1.

7.1. Lambda Function

The best possible solution to build a cloud-based service for an education Pet Security System is to use Lambda an Amazon Web Services that runs the code only when it needs to be executed and scales down automatically. The advantages of Lambda are

1. No need to run servers continuously.
2. Executes the skill code in response to the voice and manages the compute resources automatically.
3. The Lambda free tier is sufficient for the function supporting a Pet Security System and the first one million requests each month are free, helps students and developers to learn and deploy.

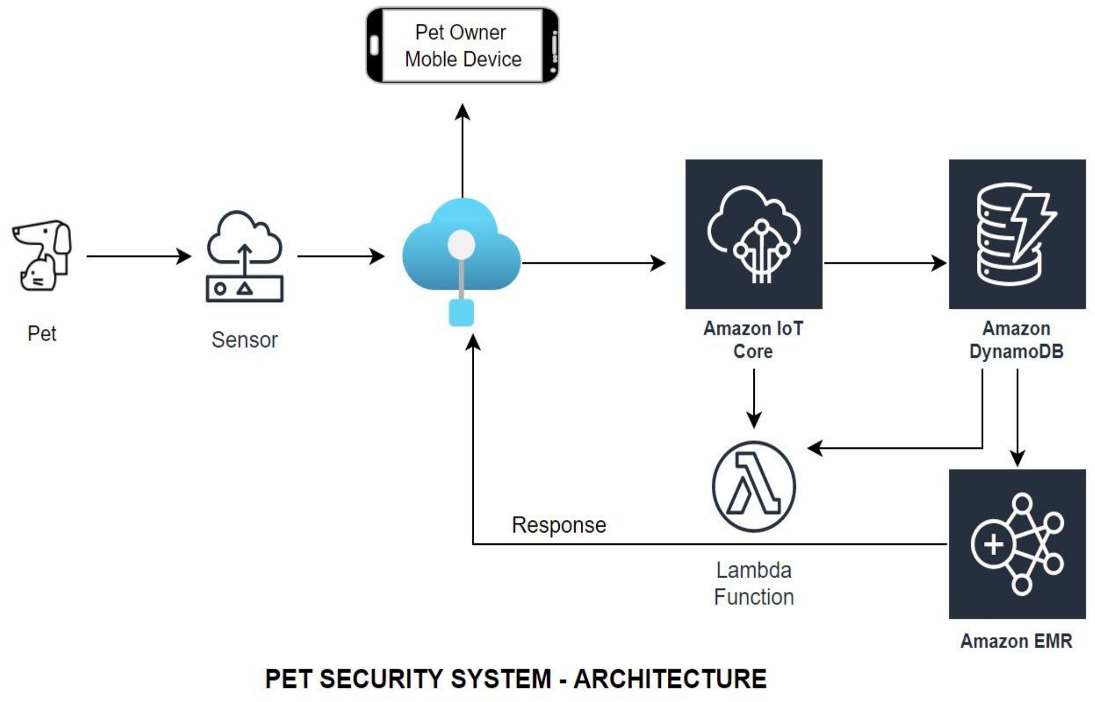


Figure 1 Pet Security System Architecture

7.2. DynamoDB

DynamoDB is a fully managed NoSQL database service that provides good performance. DynamoDB tables can store any amount of data and handle high amount of data requests. The advantages are

1. DynamoDB is server less hence there is no server requirement to provision, patch, manage.
2. The global table replicates data across multiple AWS Regions so that stored data can be retrieved from your local location for distributed applications.

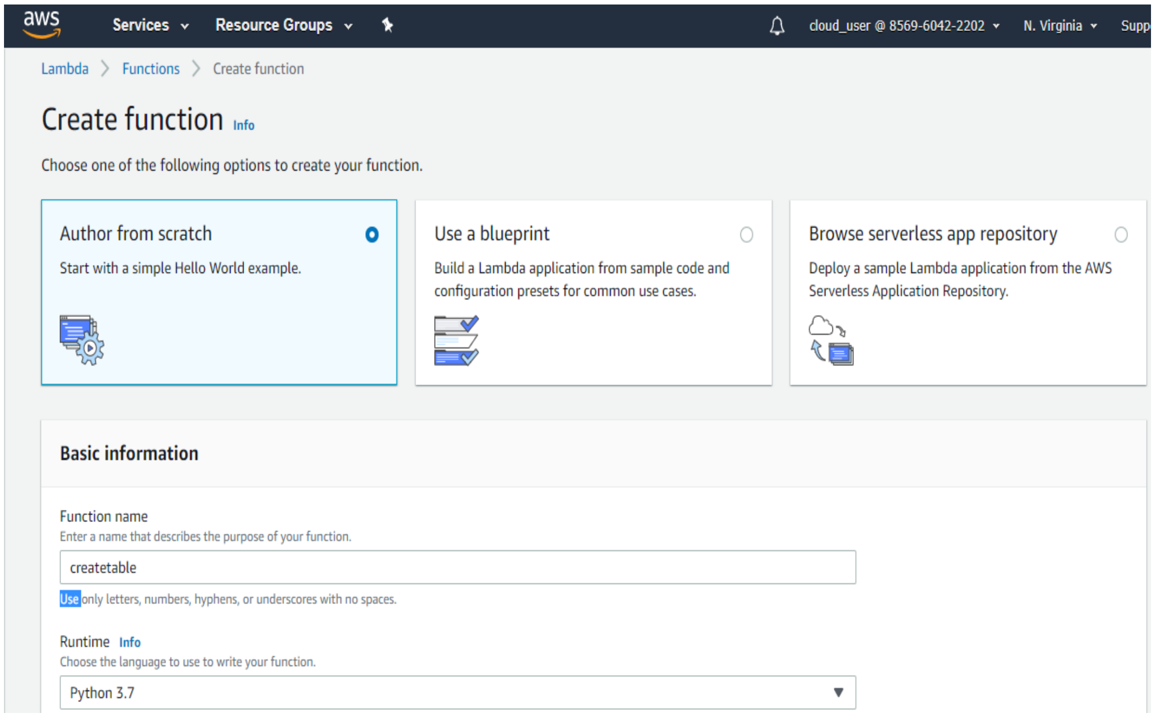
8. Pet Security system Implementation

Pet Security System is implemented in AWS management console. This work implements the Aws Lambda function and a Cloudwatch event to trigger the Lambda function

8.1. Create a Lambda Function

Lambda Function is created to receive the input and process it.

1. Open Lambda Function console
2. Select the Region us-east-1 (US East (N. Virginia))
3. Click on "Create Function" – To create a new Lambda Function
4. Select "Author from scratch" – Enter the function name and select the Runtime to run the code.
5. Click "Create Function"



Create function Info

Choose one of the following options to create your function.

Author from scratch [✓]
Start with a simple Hello World example.

Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

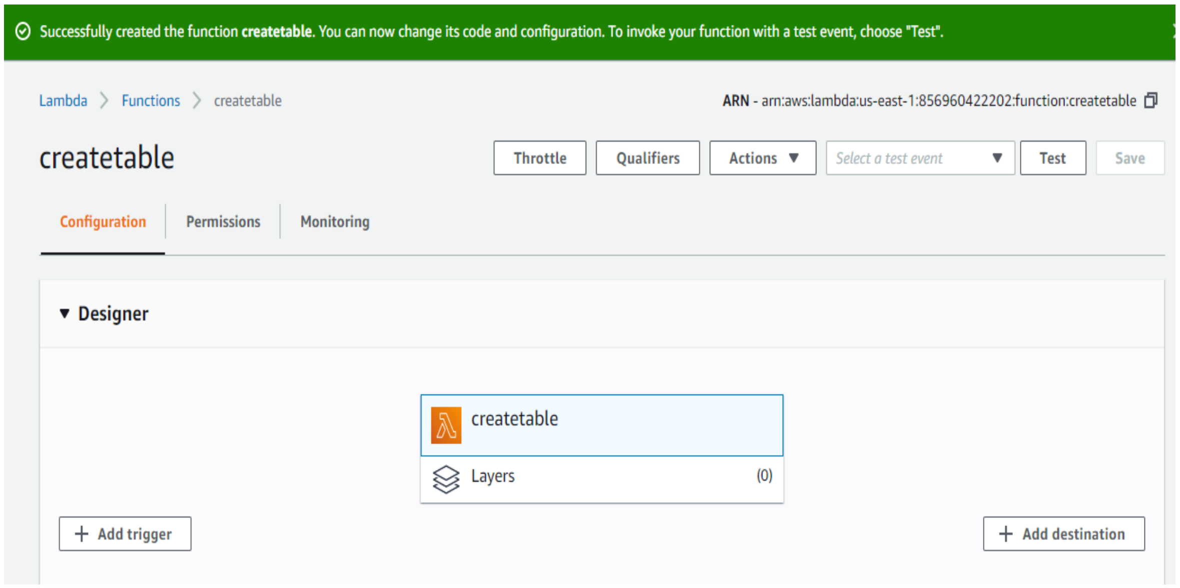
Browse serverless app repository
Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name
Enter a name that describes the purpose of your function.
createtable
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function.
Python 3.7

Figure 2 Create a Lambda Function


Successfully created the function **createtable**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > createtable

ARN - arn:aws:lambda:us-east-1:856960422202:function:createtable

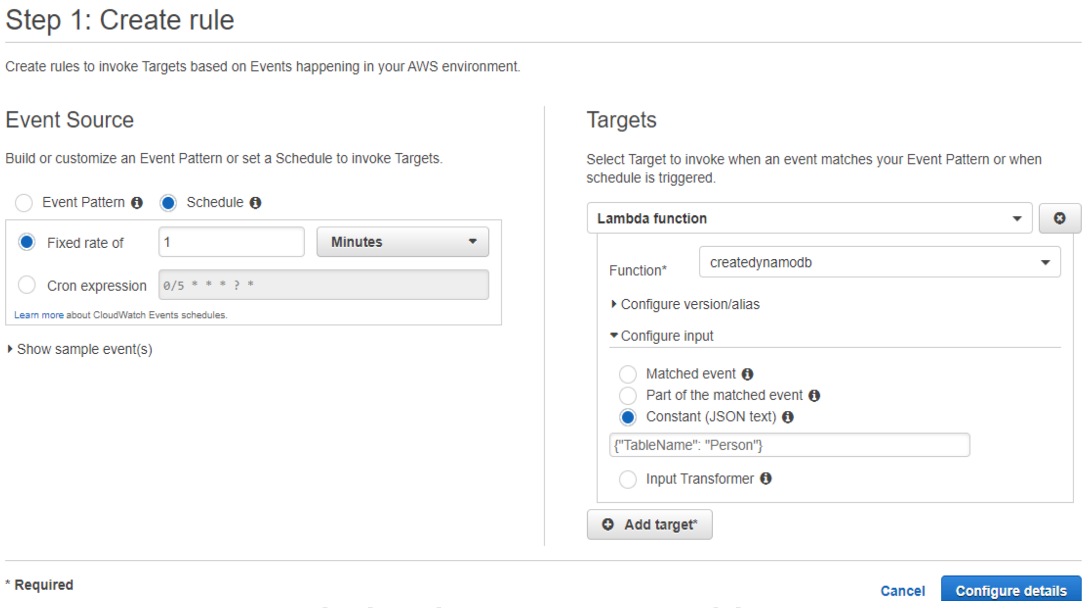
createtable

Figure 3 Lambda Function created successfully

8.2. CloudWatch to trigger Lambda Function

Set up a rule to run an AWS Lambda Function on a schedule.

1. Open the CloudWatch console
2. Events- Create rule.
    - Choose Schedule.
    - Choose Fixed rate of and specify the schedule interval- 1 minute
3. Choose Add target, Lambda Function.
4. Function-Lambda Function created.
5. Click Configure details.
6. For Rule definition, type a name and description for the rule.
7. Choose Create rule.


Step 1: Create rule

Create rules to invoke Targets based on Events happening in your AWS environment.

Event Source

Build or customize an Event Pattern or set a Schedule to invoke Targets.

Event Pattern [✓ Schedule]

[✓ Fixed rate of] 1 Minutes

Cron expression 0/5 * * * ? *

Learn more about CloudWatch Events schedules.

Show sample event(s)

Targets

Select Target to invoke when an event matches your Event Pattern or when schedule is triggered.

Lambda function

Function* createdynamodb

Configure version/alias

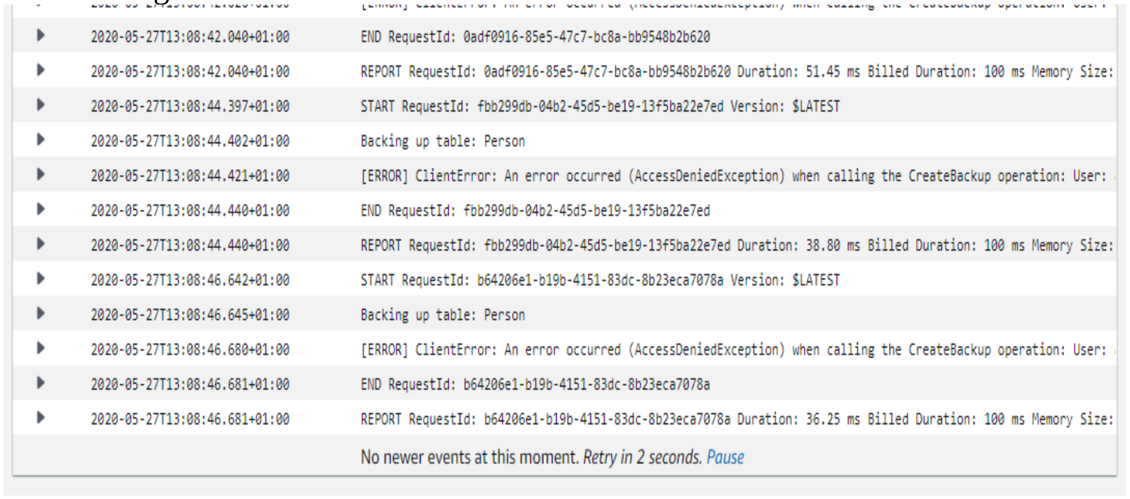
Configure input

Matched event
Part of the matched event
[✓ Constant (JSON text)]
{"TableName": "Person"}
Input Transformer

Figure 4 AWS CloudWatch Event to Trigger Lambda Function

8.3. CloudWatch Logs

The cloud watch logs shows no new events occurred.



```
2020-05-27T13:08:42.040+01:00    END RequestId: 0adf0916-85e5-47c7-bc8a-bb9548b2b620
2020-05-27T13:08:42.040+01:00    REPORT RequestId: 0adf0916-85e5-47c7-bc8a-bb9548b2b620 Duration: 51.45 ms Billed Duration: 100 ms Memory Size:
2020-05-27T13:08:44.397+01:00    START RequestId: fbb299db-04b2-45d5-be19-13f5ba22e7ed Version: $LATEST
2020-05-27T13:08:44.402+01:00    Backing up table: Person
2020-05-27T13:08:44.421+01:00    [ERROR] ClientError: An error occurred (AccessDeniedException) when calling the CreateBackup operation: User:
2020-05-27T13:08:44.440+01:00    END RequestId: fbb299db-04b2-45d5-be19-13f5ba22e7ed
2020-05-27T13:08:44.440+01:00    REPORT RequestId: fbb299db-04b2-45d5-be19-13f5ba22e7ed Duration: 38.80 ms Billed Duration: 100 ms Memory Size:
2020-05-27T13:08:46.642+01:00    START RequestId: b64206e1-b19b-4151-83dc-8b23eca7078a Version: $LATEST
2020-05-27T13:08:46.645+01:00    Backing up table: Person
2020-05-27T13:08:46.680+01:00    [ERROR] ClientError: An error occurred (AccessDeniedException) when calling the CreateBackup operation: User:
2020-05-27T13:08:46.681+01:00    END RequestId: b64206e1-b19b-4151-83dc-8b23eca7078a
2020-05-27T13:08:46.681+01:00    REPORT RequestId: b64206e1-b19b-4151-83dc-8b23eca7078a Duration: 36.25 ms Billed Duration: 100 ms Memory Size:
                                 No newer events at this moment. Retry in 2 seconds. Pause
```

Figure 5 CloudWatch Logs

9. AWS Services Costing

Total Cost Optimization (TCO) is important for deploying Pet Security System in AWS cloud platform. The table below represents the ballpark estimate of cloud.

AWS SERVICES	AVERAGE PRICE
S3 Standard - General purpose storage(First 50 TB / Month)	$0.023 per GB
LAMBDA(Requests)	$0.20 per 1M requests
DYNAMO DB write request units	$1.25 per million write request units
DYNAMO DB read request units	$0.25 per million read request units
CLOUD WATCH (First 10,000 metrics)	$0.30
TCO	$2.023/Month

Table 5 TCO Calculation

10. Analysis and Reflection

The study proposes that the use of DynamoDB offers on-demand backup and restore to create full backups of the table for data archiving to help the corporate and government regulatory. The Pet Security system is developed in accordance to the Government compliance and regulation specific to region, encryption of PII (Personally Identifiable Information). The key characteristics of any Cloud based service is high-availability, fault tolerance, scalability and elasticity. The proposed architecture is high available because of Lambda function and DynamoDB.

11. Conclusion

The proposed architecture for pet security system aims at providing best solution for cloud based deployments. The architecture has AWS components such AWS Iot Core to process Iot Data and AWS EMR to perform data analysis and to provide insights about the pet security system. The architecture has lambda for server handling which reduces the time for provisioning and managing EC2 instances. The lambda function requires only AWS resources to run the Pet Security System code which reduces resource costs. The system also has DynamoDB which is the major advantage for Security systems to store data globally and can be accessed for any location despite of datacenter region. The proposed system can be enhanced to meet any unprecedented situations such as pandemics, handles large volumes of data requests simultaneously and to offers offline support.

12. References

[1]. S.Jeschke, c.Brecher, H.Song and D.B.Rawat, "Industrial Internet of Things", Cyber Manufacturing systems. Springer 2017.

[2]. G. Dartmann, H. Song, and A. Schmeink, "Big Data Analytics for Cyber-Physical Systems: Machine Learning for the Internet of Things". Elsevier, 2019.

[3]. Karley Beckman, Sue Bennett and Lori Lockyer "Understanding students' use and value of technology for learning" Learning, Media and Technology volume 39, issue 3, pp.346-367,2014.

[4]. K.-D. Thoben, S. Wiesner, and T. Wuest, "industrie 4.0 and smart manufacturing-a review of research issues and application examples," International Journal of Automation Technology, vol. 11, no. 1, pp. 4–16, 2017.