



Sviluppare efficacemente applicazioni serverless con AWS SAM

Stefano Marseglia
EPSILON S.R.L.

SPONSOR

DIAMOND



GUCCI

splunk>



PLATINUM

improve



GOLD



CODICEPLASTICO



Cubbit

 CLOUD DAY 2022

#CLOUDDAY2022

PARTNER



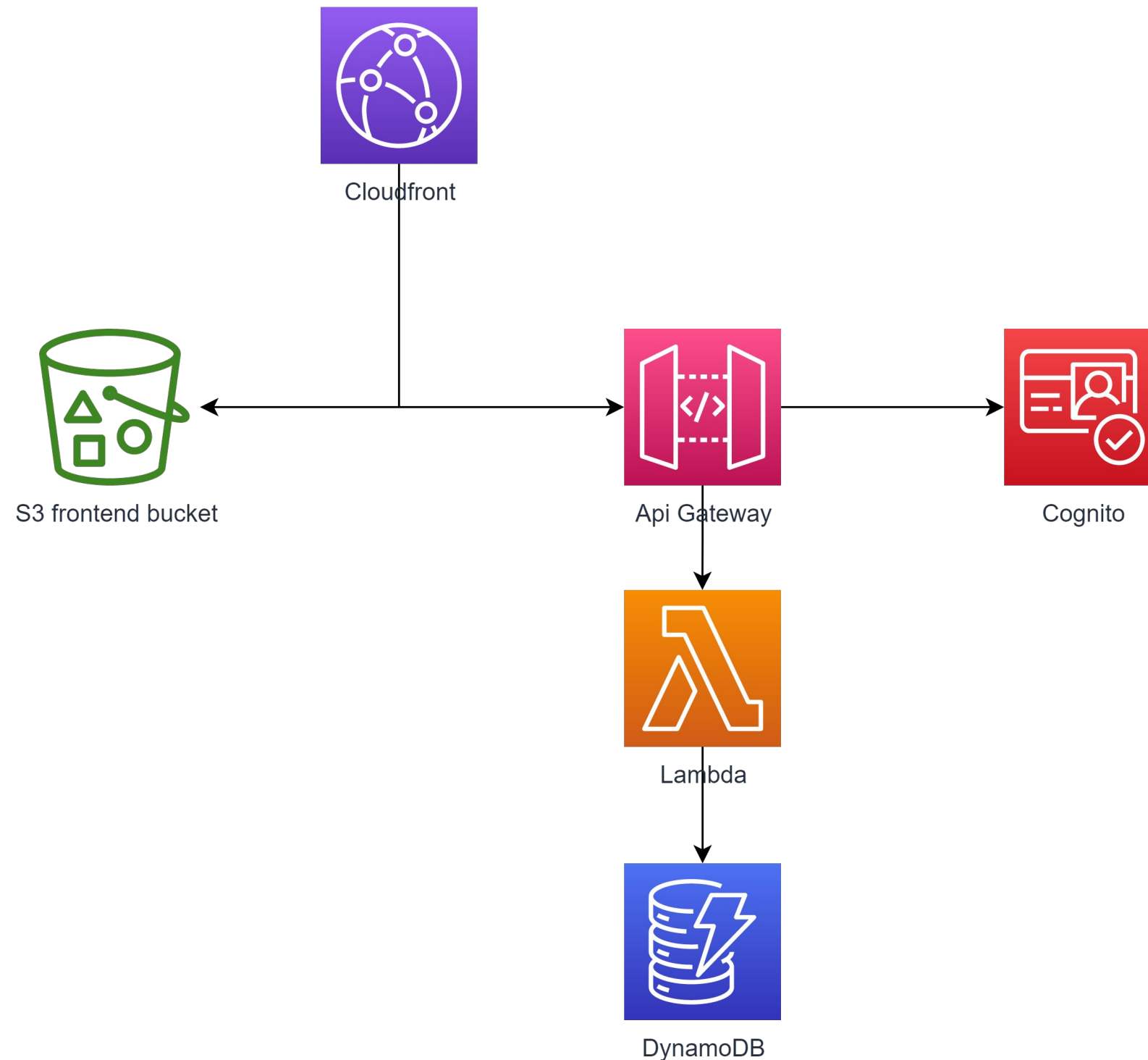
CLOUD DAY 2022



#CLOUDDAY2022

SERVERLESS COMPUTING

Serverless



DynamoDB: Storage NoSQL completamente gestito

Lambda: Implementazione backend delle API ed esecuzione in modalità serverless

Api Gateway: Esposizione API Lambda

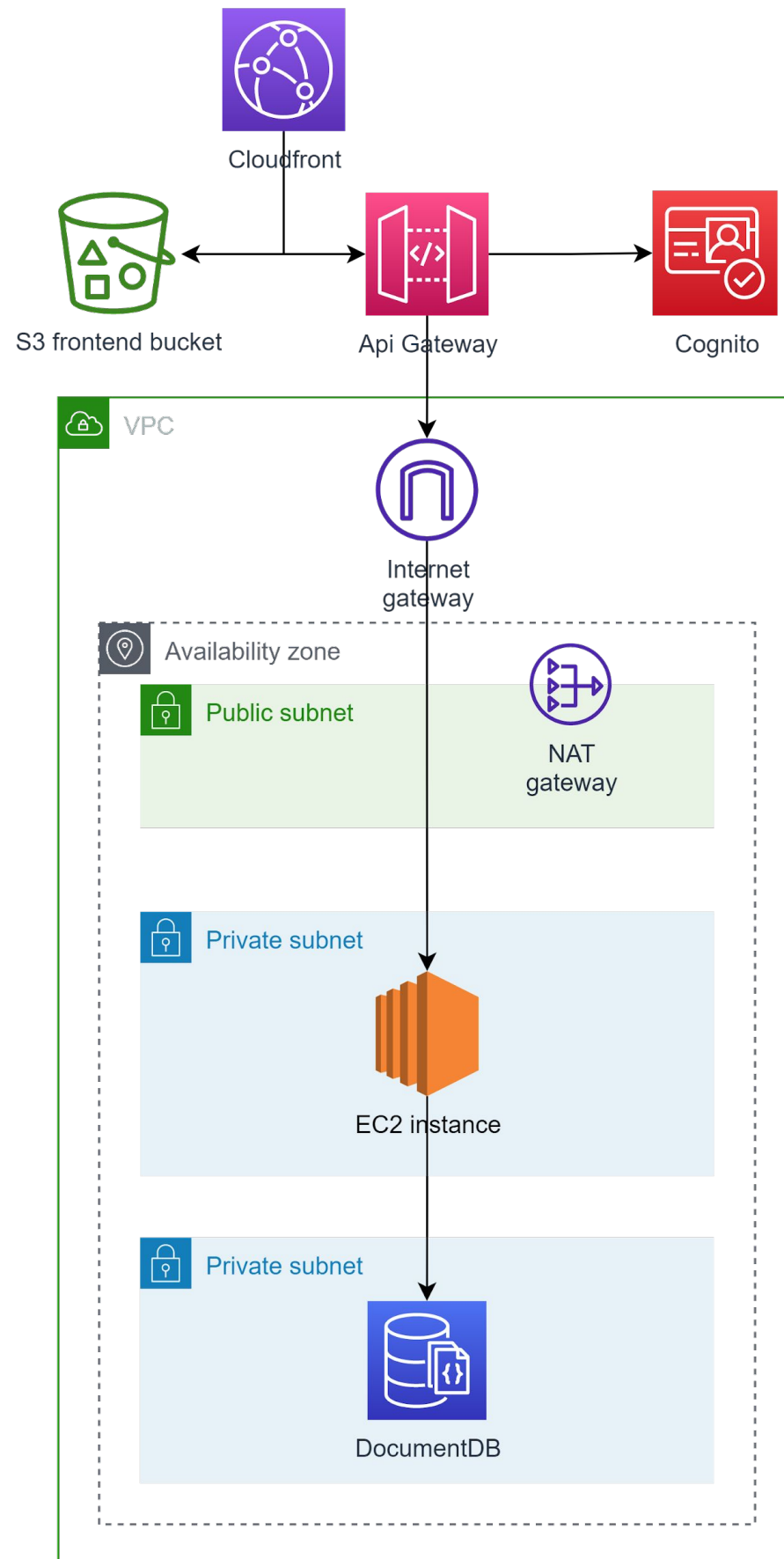
Cognito: Gestione utenti

S3: Storage a oggetti per ospitare contenuti statici del frontend

Cloudfront: Content Delivery Network

Costo: 😊
Gestione: 😊

IaaS - AZ singola



Implementiamo la stessa applicazione senza utilizzare servizi serverless per le api di backend:

~ **DynamoDB => DocumentDB:** Storage NoSQL compatibile con mongoDB

~ **Lambda => EC2:** Implementazione backend delle API su istanze di macchine virtuali

+ **NAT gateway:** NAT Gateway per permettere alle EC2 nelle subnet private di comunicare con il mondo esterno

+ **Internet Gateway:** Per permettere al mondo esterno di comunicare con l'istanza EC2

Costo: 😐
Gestione: 😐

Costi

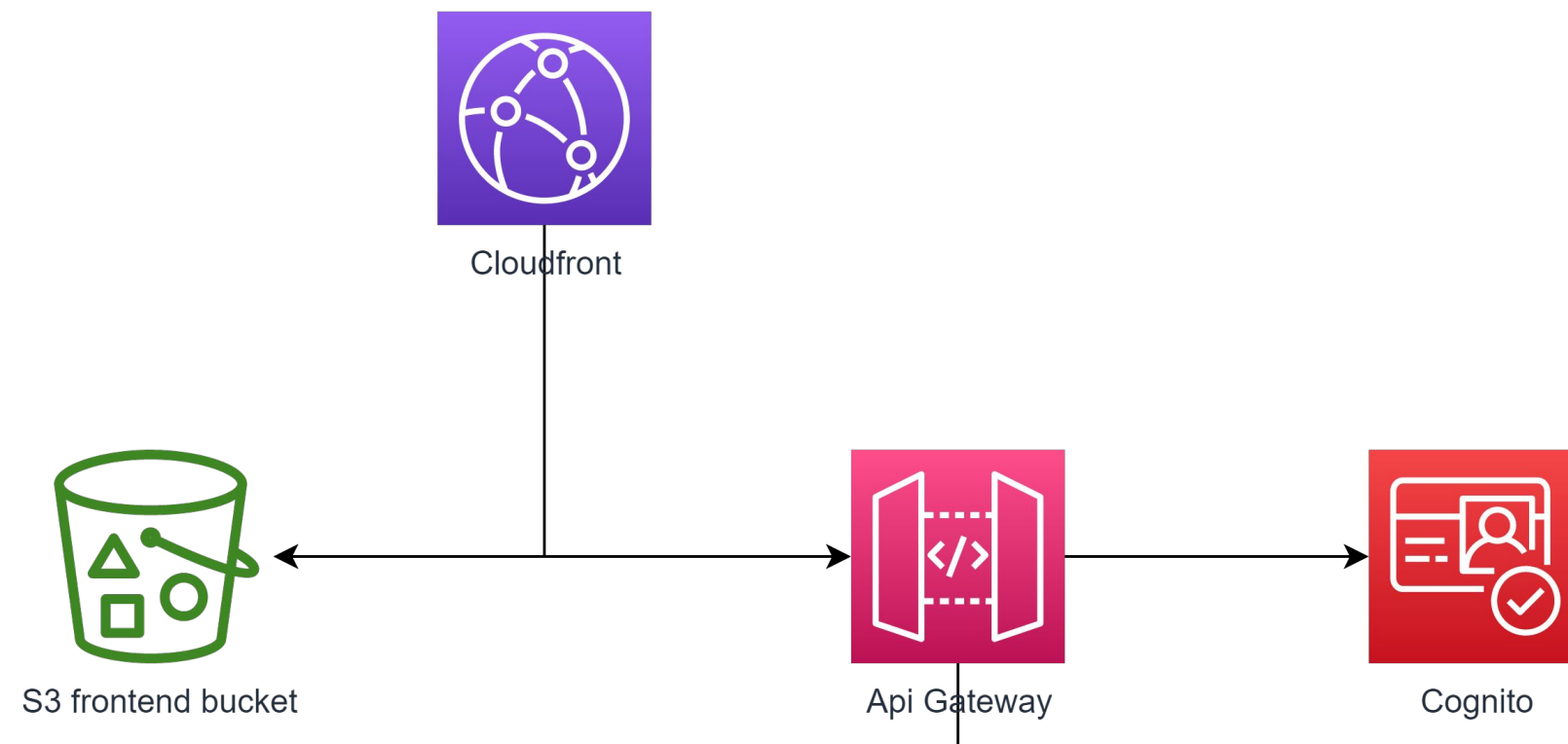
500 chiamate al minuto

http://

500MB di storage S3



50 utenti cognito attivi al mese



~5\$/mese



Costi

Backend tradizionale

EC2 t3.medium: 4GiB + 2vCPU

DB t3.medium: 4GiB + 2vCPU

~150\$/mese



Backend serverless

Lambda functions

DynamoDB

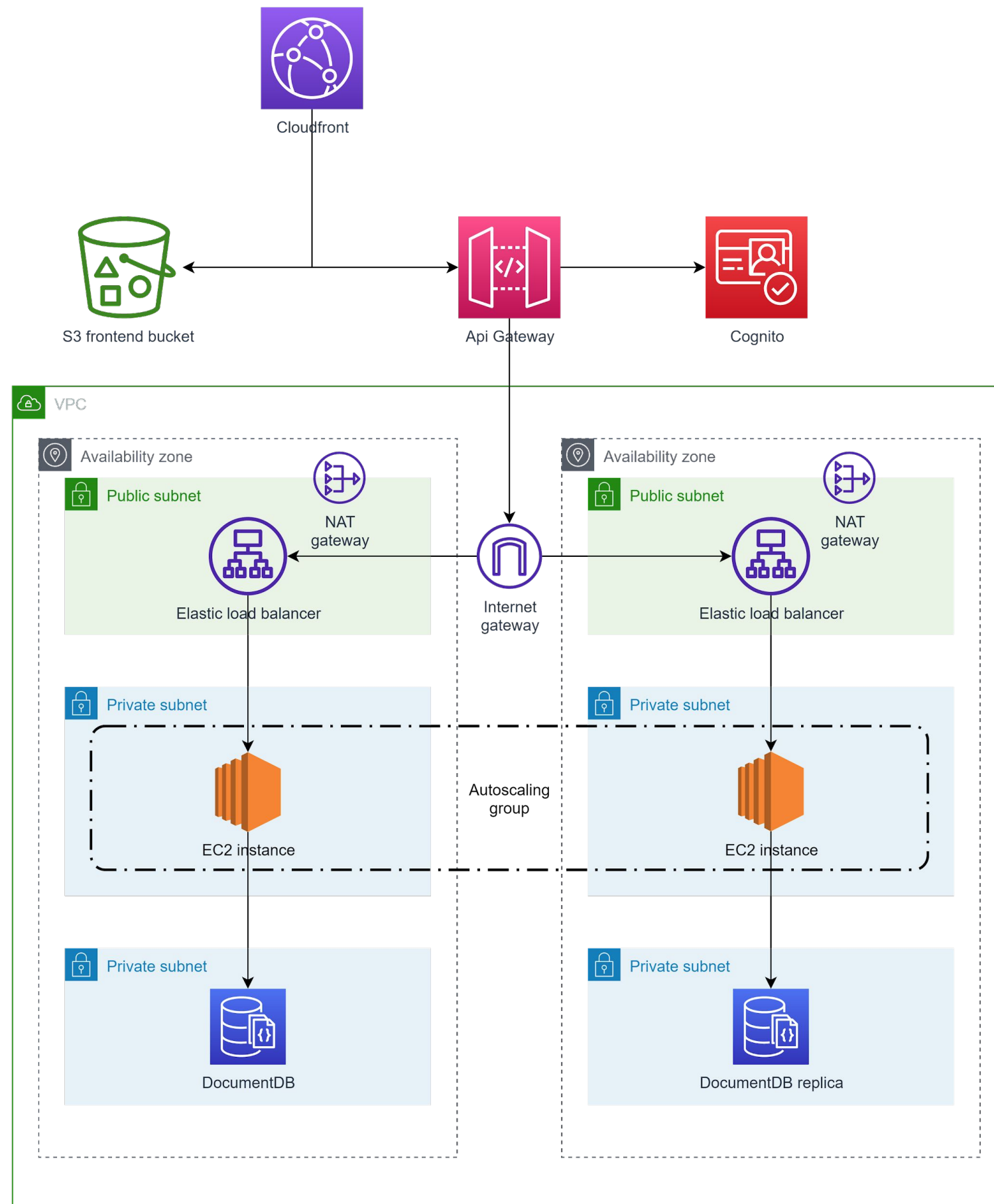
~3\$/mese



**ABBATTIMENTO DEI COSTI
DEL 98% !!!**



IaaS - AZ multiple



Fault tolerance? Autoscaling?

- Aggiunta di *read replica* DocumentDB per la nuova *Availability Zone* (AZ)
- Setup di un *autoscaling group* per effettuare *scale in/out* delle istanze in base al carico e per tolleranza al fallimento di una delle due AZ
- Aggiunto un *Elastic Load Balancer* deployato su due AZ per inoltrare il traffico in ingresso verso le istanze dell'autoscaling group
- Aggiunta di un secondo NAT gateway per la nuova AZ

Costo: 😞
Gestione: 😞

Serverless - svantaggi

Funzioni stateless

“Le mie lambda terminano improvvisamente e perdo tutte le sessioni in memoria”



Estrai le sessioni dalle tue lambda e utilizza uno storage esterno (es: *DynamoDB*)

Componenti COTS

“Sto utilizzando delle API non implementate da me che non sono state scritte per Lambda”



Utilizza *ECS* con provider *Fargate*

Developer experience discutibile

“Come faccio a gestire infrastruttura e codice contemporaneamente? Come testo continuamente la mia applicazione se questa gira su cloud? Come posso effettuare facilmente il deploy?”



Anche per questo c'è una soluzione



AWS SAM

Serverless Application Model



AWS SAM (Serverless Application Model)

- **AWS SAM template specification.** You use this specification to define your serverless application. It provides you with a simple and clean syntax to describe the functions, APIs, permissions, configurations, and events that make up a serverless application. You use an AWS SAM template file to operate on a single, deployable, versioned entity that's your serverless application.
- **AWS SAM command line interface (AWS SAM CLI).** You use this tool to build serverless applications that are defined by AWS SAM templates. The CLI provides commands that enable you to verify that AWS SAM template files are written according to the specification, invoke Lambda functions locally, step-through debug Lambda functions, package and deploy serverless applications to the AWS Cloud, and so on.

SAM roadmap

1. Inizializzazione progetto (`sam init`)
2. build (`sam build`)
3. debug in locale (`sam local`)
4. debug su cloud (`sam sync`)
5. deploy (`sam deploy`)



ATTENZIONE: CODICE IN ARRIVO!!!



SAM init

Sviluppiamo due semplici API per la memorizzazione e la lettura di *TODO*

Il comando *sam init* crea un'applicazione di esempio selezionabile tra alcuni template messi a disposizione da AWS

```
stefano@stefano-laptop: ~/work/cloudday$ sam init
```

```
Which template source would you like to use?
```

- 1 - AWS Quick Start Templates
- 2 - Custom Template Location

```
Choice: 1
```

```
Choose an AWS Quick Start application template
```

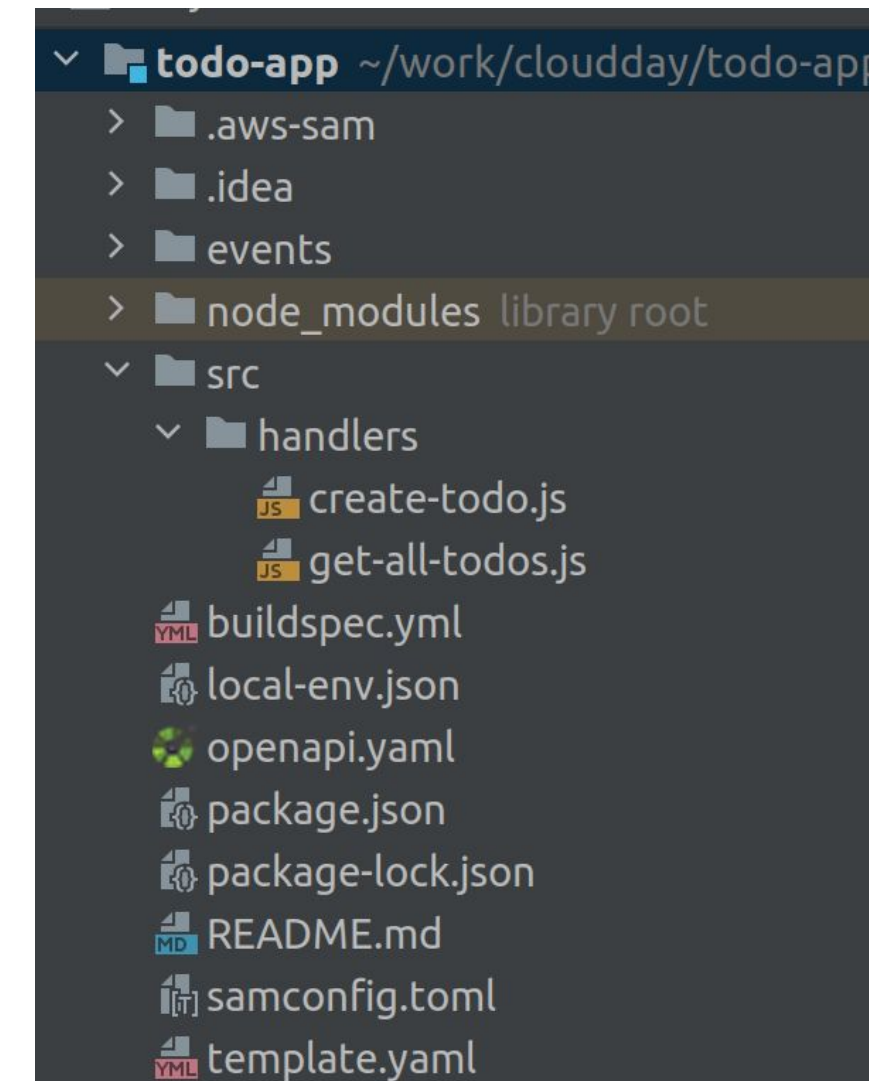
- 1 - Hello World Example
- 2 - Multi-step workflow
- 3 - Serverless API
- 4 - Scheduled task
- 5 - Standalone function
- 6 - Data processing
- 7 - Infrastructure event management
- 8 - Lambda EFS example
- 9 - Machine Learning

```
Template: 5
```

```
Which runtime would you like to use?
```

- 1 - dotnet6
- 2 - dotnetcore3.1
- 3 - nodejs16.x
- 4 - nodejs14.x
- 5 - nodejs12.x

```
Runtime: 3
```



Template - Risorse AWS::Serverless::Function



```
getAllTodosFunction :
  Type: AWS::Serverless::Function
  Properties:
    Handler:
      src/handlers/get-all-todos.handler
    Runtime: nodejs16.x
    Architectures:
      - x86_64
    MemorySize: 128
    Timeout: 100
    Description: Gets all todos
    Policies:
      - DynamoDBCrudPolicy:
          TableName: !Ref TodoTable
    Environment:
      Variables:
        TODO_TABLE: !Ref TodoTable
    Events:
      Api:
        Type: Api
        Properties:
          Path: /todos
          Method: GET
```

- Specifica il percorso in cui cercare l'entry point della lambda function che gestisce l'evento

```
/* ./src/handlers/get-all-todos.js */

exports.handler = async (event) => {

  // parse body

  // prepare response
  const response = { ... };

  // return response
  return response;
};
```

- Crea un ruolo da applicare alla lambda function a partire dal template *DynamoDBCrudPolicy* che fornisce permessi in lettura e scrittura su una tabella DynamoDB creata nello stesso template:

```
TodoTable:
  Type: AWS::Serverless::SimpleTable
  Properties:
    PrimaryKey:
      Name: id
      Type: String
    ProvisionedThroughput:
      ReadCapacityUnits: 2
      WriteCapacityUnits: 2
```

- Crea una variabile d'ambiente per la funzione lambda che fa riferimento al nome della tabella DynamoDB. L'applicazione può leggere la variabile per effettuare operazioni sulla tabella
- Specifica quali tipi di eventi causano l'esecuzione della Lambda, in questo caso una chiamata effettuata su ApiGateway
- Genera automaticamente una risorsa *AWS::ApiGateway::RestApi*

#CLOUDDAY2022

Template - Risorse AWS::Serverless::Api

Se volessimo più controllo sulla risorsa Api Gateway creata?

```
Events:
  Api:
    Type: Api
    Properties:
      RestApiId: !Ref ApiGatewayApi
      Path: /todos
      Method: GET
```

+



```
ApiGatewayApi:
  Type: AWS::Serverless::Api
  Properties:
    EndpointConfiguration:
      Type: REGIONAL
    StageName: Prod
    Cors:
      AllowMethods: "'POST, GET, DELETE'"
      AllowHeaders: "'X-Forwarded-For'"
      AllowOrigin: "'*'"
    OpenApiVersion: 3.0.1
    DefinitionBody:
      'Fn::Transform':
        Name: AWS::Include
        Parameters:
          Location: openapi.yaml
```

- La Lambda viene automaticamente integrata con una risorsa api gateway creata nello stesso template e referenziata
- La risorsa *ApiGatewayApi* può essere definita tramite un documento *OpenApi* (o *swagger*)
- Il documento può essere presente all'interno dello stesso progetto SAM o su S3
- Tramite la funzione cloudformation *Fn::Transform* di tipo *AWS::Include* è possibile includere il documento *OpenApi* all'interno del quale si può fare riferimento alle risorse e alle variabili generate all'interno dello stesso template

Openapi document



```
openapi: 3.0.0
info:
  title: Todo API
  description: Todo API
  version: 1.0.0
servers:
  - url: !Sub "https://${ApiGatewayApi}.execute-api.${AWS::Region}.amazonaws.com/Prod"
tags:
  - name: tasks
    description: Tasks
paths:
  /todos:
    get:
      tags:
        - tasks
      summary: List the available tasks
      x-amazon-apigateway-integration:
        uri: { "Fn::Sub": "arn:aws:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/${getAllTodosFunction.Arn}/invocations" }
        httpMethod: "POST"
        passthroughBehavior: "when_no_match"
        type: "aws_proxy"
      operationId: listTodos
      responses:
        '200':
          description: successful operation
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Todo'
      [...]
```

#CLOUDDAY2022

Build - sam build



Il Comando *sam build* effettua la build adatta al runtime scelto (nodejs16)

```
stefano@stefano-laptop: ~/work/cloudday/todo-app$ sam build
```

```
Building codeuri: /home/stefano/work/cloudday/todo-app runtime: nodejs16.x metadata: {} architecture: x86_64 functions:
```

```
getAllTodosFunction, createTodoFunction
```

```
Running NodejsNpmBuilder:NpmPack
```

```
Running NodejsNpmBuilder:CopyNpmrcAndLockfile
```

```
Running NodejsNpmBuilder:CopySource
```

```
Running NodejsNpmBuilder:NpmInstall
```

```
Running NodejsNpmBuilder:CleanUpNpmrc
```

```
Running NodejsNpmBuilder:LockfileCleanUp
```

```
Build Succeeded
```

```
Built Artifacts : .aws-sam/build
```

```
Built Template : .aws-sam/build/template.yaml
```

```
Commands you can use next
```

```
=====
```

```
[*] Validate SAM template: sam validate
```

```
[*] Invoke Function: sam local invoke
```

```
[*] Test Function in the Cloud: sam sync --stack-name {stack-name} --watch
```

```
[*] Deploy: sam deploy --guided
```

Testing - sam local



Il comando *sam local* permette di simulare localmente, tramite l'utilizzo di appositi container docker, i servizi *Lambda* e *Api Gateway*

```
stefano@stefano-laptop: ~/work/cloudday/todo-app$ sam local start-api --env-vars local-env.json --config-env dev --profile lab-aws
```

```
Mounting createTodoFunction at http://127.0.0.1:3000/todos [POST]
```

```
Mounting getAllTodosFunction at http://127.0.0.1:3000/todos [GET]
```

```
You can now browse to the above endpoints to invoke your functions. You do not need to restart/reload SAM CLI while working on your functions, changes will be reflected instantly/automatically. You only need to restart SAM CLI if you update your AWS SAM template
```

```
2022-10-24 15:23:10 * Running on http://127.0.0.1:3000/ (Press CTRL+C to quit)
```

```
Invoking src/handlers/get-all-todos.handler (nodejs16.x)
```

```
Skip pulling image and use local one: public.ecr.aws/sam/emulation-nodejs16.x:rapid-1.55.0-x86_64.
```

```
Mounting /home/stefano/work/cloudday/todo-app/.aws-sam/build/getAllTodosFunction as /var/task:ro,delegated inside runtime container
```

```
START RequestId: b3fdb054-6bcb-40f4-b3bc-2ca434017c45 Version: $LATEST
```

```
2022-10-24T13:23:29.450Z b3fdb054-6bcb-40f4-b3bc-2ca434017c45 INFO received: {
```

```
  "id": "f4bf9ec4-02d3-4301-b833-9892070fec84",
```

```
  "title": "Completare le slide per il Cloud Day 2022"
```

```
}
```

```
2022-10-24T13:23:29.452Z b3fdb054-6bcb-40f4-b3bc-2ca434017c45 INFO response from: / statusCode: 200 body:
```

```
{"id":"f4bf9ec4-02d3-4301-b833-9892070fec84","name":"Completare le slide per il Cloud Day 2022"}
```

```
2022-10-24 15:23:30 127.0.0.1 - - [24/Oct/2022 15:23:30] "POST /todos HTTP/1.1" 200 -
```

Testing - SAM accelerate



L'approccio della simulazione dei servizi in locale ha dei limiti, ad esempio:

- Impossibilità di simulare qualsiasi servizio aws in locale
- Problemi relativi a permessi *IAM*

La risposta: invece di cercare di emulare altri servizi in locale, velocizziamo e semplifichiamo il deploy su cloud con *SAM accelerate*

```
stefano@stefano-laptop: ~/work/cloudday/todo-app sam sync --watch --stack-name todo-app --profile lab-aws
```

```
Syncing Lambda Function createTodoFunction...
```

```
Syncing Lambda Function getAllTodosFunction...
```

```
Manifest is not changed for (createTodoFunction), running incremental build
```

```
Building codeuri: /home/stefano/work/cloudday/todo-app runtime: nodejs16.x metadata: {} architecture: x86_64
```

```
functions: createTodoFunction
```

```
download_dependencies is False and dependencies_dir is None  
directory.
```

```
Running NodejsNpmBuilder:NpmPack
```

```
Running NodejsNpmBuilder:CopyNpmrcAndLockfile
```

```
Running NodejsNpmBuilder:CopySource
```

```
Running NodejsNpmBuilder:CleanUpNpmrc
```

```
Running NodejsNpmBuilder:LockfileCleanUp
```

```
Running NodejsNpmBuilder:LockfileCleanUp
```

```
Finished syncing Lambda Function createTodoFunction.
```

```
stefano@stefano-laptop: ~/work/cloudday/todo-app sam logs --stack-name todo-app --tail --region eu-west-1 --profile lab-aws
```

```
2022/10/24/[$LATEST]c1204052ba03419d9c051831834adcb5 2022-10-24T12:54:45.805000 START RequestId:
```

```
bbc46025-8c95-4085-a81f-bdb531b03315 Version: $LATEST
```

```
2022/10/24/[$LATEST]c1204052ba03419d9c051831834adcb5 2022-10-24T12:54:45.808000 2022-10-24T12:54:45.807Z
```

```
bbc46025-8c95-4085-a81f-bdb531b03315 INFOreceived: {
```

```
  "id": "f4bf9ec4-02d3-4301-b833-9892070fec84",
```

```
  "name": "Completare le slide per il Cloud Day 2022"
```

```
}
```

```
2022/10/24/[$LATEST]c1204052ba03419d9c051831834adcb5 2022-10-24T12:54:46.356000 2022-10-24T12:54:46.356Z
```

```
bbc46025-8c95-4085-a81f-bdb531b03315 INFOresponse from: / statusCode: 200 body:
```

```
{"id":"f4bf9ec4-02d3-4301-b833-9892070fec84","name":"Completare le slide per il Cloud Day 2022"}
```

```
2022/10/24/[$LATEST]c1204052ba03419d9c051831834adcb5 2022-10-24T12:54:46.394000 END RequestId:
```

```
bbc46025-8c95-4085-a81f-bdb531b03315
```

```
2022/10/24/[$LATEST]c1204052ba03419d9c051831834adcb5 2022-10-24T12:54:46.394000 REPORT RequestId:
```

```
bbc46025-8c95-4085-a81f-bdb531b03315 Duration: 588.83 msBilled Duration: 589 ms Memory Size: 128 MBMax Memory
```

```
Used: 74 MB Init Duration: 280.16 ms
```

Deploy - sam deploy

```
stefano@stefano-laptop: ~/work/cloudday/todo-app$ sam deploy --guided --profile lab-aws
```

Configuring SAM deploy

=====

Looking for config file [samconfig.toml] : Found
Reading default arguments : Success

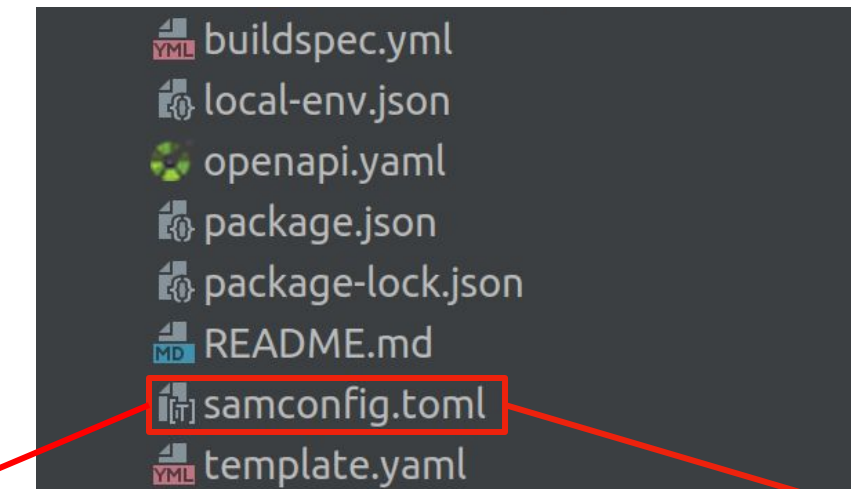
Setting default arguments for 'sam deploy'

=====

Stack Name [sam-app]: todo-app
AWS Region [us-west-2]: eu-west-1
#Shows you resources changes to be deployed and require a 'Y' to initiate deploy
Confirm changes before deploy [y/N]: y
#SAM needs permission to be able to create roles to connect to the resources in your template
Allow SAM CLI IAM role creation [Y/n]:
#Preserves the state of previously provisioned resources when an operation fails
Disable rollback [y/N]: y
Save arguments to configuration file [Y/n]:
SAM configuration file [samconfig.toml]:
SAM configuration environment [default]: dev

Looking for resources needed for deployment:
Managed S3 bucket: aws-sam-cli-managed-default-samclisourcebucket-1i9jybuyi232r
A different default S3 bucket can be set in samconfig.toml

Saved arguments to config file
Running 'sam deploy' for future deployments will use the parameters saved above.
The above parameters can be changed by modifying samconfig.toml
Learn more about samconfig.toml syntax at
<https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/serverless-sam-cli-config.html>



```
version = 0.1
[dev]
[dev.deploy]
[dev.deploy.parameters]
stack name = "todo-app"
s3_bucket = "aws-sam-cli-managed-default-samclisourcebucket-1i9jybuyi232r"
s3_prefix = "todo-app"
region = "eu-west-1"
profile = "lab-aws"
confirm_changeset = true
capabilities = "CAPABILITY_IAM"
disable_rollback = true
image_repositories = []
```

Deploy - sam deploy

Initiating deployment

=====

Uploading to todo-app/a84e2add9a8fad4689807a56b78b132b.template 2547 / 2547 (100.00%)

Waiting for changeset to be created..

CloudFormation stack changeset

Operation	LogicalResourceId	ResourceType	Replacement
-----------	-------------------	--------------	-------------

+ Add	ApiGatewayApiDeployment0722088b03	AWS::ApiGateway::Deployment	N/A
-------	-----------------------------------	-----------------------------	-----

+ Add	ApiGatewayApiProdSt		
-------	---------------------	--	--

+ Add	ApiGatewayApi		
-------	---------------	--	--

+ Add	TodoTable		
-------	-----------	--	--

+ Add	createTodoFunctionA		
-------	---------------------	--	--

+ Add	createTodoFunctionR		
-------	---------------------	--	--

+ Add	createTodoFunction		
-------	--------------------	--	--

+ Add	getAllTodosFunctionA		
-------	----------------------	--	--

+ Add	getAllTodosFunctionR		
-------	----------------------	--	--

+ Add	getAllTodosFunction		
-------	---------------------	--	--

Changeset created successfully.

arn:aws:cloudformation:eu-west-1:921332

8ddb9

CloudFormation events from stack operations (refresh every 0.5 seconds)

ResourceStatus	ResourceType	LogicalResourceId	ResourceStatusReason
CREATE_IN_PROGRESS	AWS::DynamoDB::Table	TodoTable	-
CREATE_IN_PROGRESS	AWS::DynamoDB::Table	TodoTable	Resource creation Initiated
CREATE_COMPLETE	AWS::DynamoDB::Table	TodoTable	-
CREATE_IN_PROGRESS	AWS::IAM::Role	createTodoFunctionRole	-
[...]	[...]	[...]	[...]

CloudFormation outputs from deployed stack

Outputs

Key	WebEndpoint
Description	API Gateway endpoint URL for Prod stage
Value	https://hm5mxwcbec.execute-api.eu-west-1.amazonaws.com/Prod/

Successfully created/updated stack - todo-app in eu-west-1

stefano@stefano-laptop: ~/work/cloudday/todo-app

curl https://hm5mxwcbec.execute-api.eu-west-1.amazonaws.com/Prod/todos | jq

```
[
  {
    "id": "f4bf9ec4-02d3-4301-b833-9892070fec84",
    "title": "Completare le slide per il Cloud Day 2022",
    "status": "done"
  }
]
```

Grazie!

Diamond Sponsor



GUCCI

splunk>



Platinum Sponsor



improve

Gold Sponsor



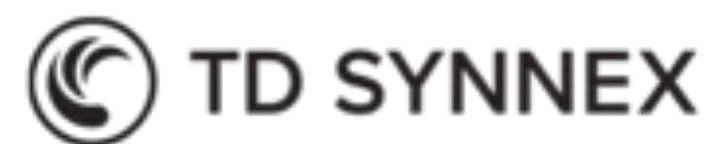
Partner



SPONSOR

 CLOUD DAY 2022

DIAMOND



PLATINUM



GOLD



#CLOUDDAY2022

PARTENER

