

Les API Google / AWS

Table des matières

I. SAAS avec AWS	3
II. Exercice : Quiz	12
III. SAAS et API avec Google Cloud	13
IV. Exercice : Quiz	18
V. Essentiel	19
VI. Auto-évaluation	19
A. Exercice	19
B. Test	20
Solutions des exercices	21

I. SAAS avec AWS

Durée : 1 h

Environnement de travail : un pc.

Contexte

Lors de la réalisation d'une application, quel que soit le support, si cette dernière est dynamique et nécessite un échange de données, elle passera sans aucun doute par une API. Cette même API peut aussi servir à traiter des données, fournir des services et faire évoluer la base de l'application avec des fonctionnalités.

L'application est alors dépendante d'une partie frontale visible à l'utilisateur (le front-end) et une partie « *cachée* » servant à effectuer toutes les actions plus dynamiques (le back-end) via, dans la plupart des cas, une API ou Interface de Programmation d'Application.

Cependant, il est nécessaire, comme pour la partie front-end, de déployer cette partie back-end, c'est là qu'interviennent plusieurs solutions et notamment le SAAS via, par exemple, Google Cloud ou AWS. À partir de là il est intéressant d'utiliser ces solutions dans des cas spécifiques.

Nous verrons ce que sont Google Cloud et AWS, mais aussi quelles sont leurs solutions d'hébergement d'API en SAAS et ce qu'est le SAAS. Nous verrons aussi comment utiliser ces 2 services fournis par des piliers du web et dans quelles conditions leur utilisation peut être pertinente.

Lors de la conception d'une application ou d'une solution et même dans certains cas, lors de l'évolution d'une solution existante, il est courant de créer une API ou Interface de programmation d'application qui permettra de transiter des données. Ainsi via l'appel à une adresse exploitant cette API, l'utilisateur recevra des données en conséquence.

Parmi les API, il est commun qu'elles soient internes à l'hébergement du projet dans sa globalité ou externes et donc possèdent leur propre hébergement. Dans le premier cas, le front-end de la solution et le back-end se trouve sur le même serveur et les appels API se font soit :

- **En direct** (appel à l'adresse localhost sur un chemin ou un port non exposé) dans ce cas aucun appel API ne peut être effectué par un service externe.
- Via un **port spécifique ouvert** (l'appel se fait toujours à localhost pour le front, mais via le port spécifique), le port étant ouvert un service peut appeler l'API grâce à l'adresse : **https://<Adresse du Site>:<Port API>**. Cette adresse ne renverra pas le site, mais gèrera les requêtes à l'API.
- Via un **chemin spécifique ouvert** (l'appel se fait à localhost pour le front, mais via le chemin spécifié), le chemin étant ouvert une application externe peut appeler l'API grâce à l'adresse : **https://<Adresse du Site>/<Chemin de l'API>**. Cette adresse ne renverra pas, comme dans le cas précédent, sur le site, mais gèrera les requêtes à l'API.

Dans le second cas, la gestion est plus spécifique, le côté front-end du site est hébergé chez un hébergeur, l'API quant à elle est hébergée ailleurs, par exemple :

- Chez le même hébergeur avec un serveur dédié
- Chez un hébergeur spécifique avec un serveur dédié
- Chez un hébergeur spécifique via une solution CAAS
- Chez un hébergeur spécifique via une solution SAAS

Dans tous les cas de figure, le front-end fera ses requêtes au serveur API afin de posséder ses fonctionnalités dynamiques.

Définition CAAS et SAAS

Les solutions CAAS et SAAS sont tirées du mouvement AAS signifiant « *As A Service* ». Le concept est simple, proposer un environnement comme service afin de déployer la solution de l'utilisateur dessus, ainsi CAAS ou « *Container As A Service* » permet de déployer un container de type Docker comme un service web (ici le container contiendrait l'API) et SAAS ou « *Software As A Service* » permet de déployer un logiciel ou une application comme un service web.

Le principe des services CAAS et SAAS, dans le cas du déploiement d'une API, est donc de fournir cette dernière comme service web, cela implique une gestion différente, car il ne s'agit plus d'héberger une solution sur un serveur privé ou un hébergement web limité en flux réseau et en capacité de ressource, mais un hébergement limité directement par les caractéristiques de l'API elle-même, par exemple dans le cas du SAAS, la limite pour le déploiement d'une API se fait très souvent en nombre d'appels.

Ainsi, si le nombre d'appels est limité à 1 000, le front-end de l'application ou les applications utilisant l'API pourront au total réaliser 1 000 appels à cette dernière.

Attention

Ici, il s'agit d'un décompte du nombre d'appels s'effectuant au niveau de l'API et non des périphériques, donc tous les appels seront comptabilisés sans tenir compte de leur provenance, par exemple 750 appels depuis le front-end du site et 250 appels depuis une application connexe.

Avec de telles limites, il est alors facile de se demander quel est l'intérêt de ces solutions face à un déploiement sur serveur privé ou sur l'hébergement web, et plusieurs réponses sont possibles selon les besoins, par exemple :

- L'intérêt du SAAS peut venir de la **séparation des métiers** : plus concrètement, un développeur n'est pas forcément formé pour le déploiement (ce qui en fait, par ailleurs, la plus grosse différence avec un DevOps), une solution SAAS comme évoquée permet un déploiement réalisé et maîtrisé de manière optimale avec un besoin de connaissances en déploiement minimal.
- L'intérêt de **déléguer** la gestion de la maintenance : dans la majorité des cas, lors d'un déploiement d'une API, il est utile de vérifier que cette dernière est en ligne, qu'il n'y a pas de bug, qu'elle répond au dernier critère de sécurité, tout cela demande une maintenance continue au niveau du serveur et du code, les solutions SAAS permettra bien souvent de faciliter, voir déléguer la maintenance côté serveur et faciliter la maintenance côté code afin que les développeurs se concentrent sur le code.

Parmi ces solutions SAAS existent ensuite plusieurs acteurs, dont 2 des plus grosses marques existant sur internet avec Amazon et sa solution AWS et Google avec Google Cloud.

Remarque

D'autres acteurs avec des offres plus ou moins complètes existent pour déployer des API en tant que Software As A Service. AWS et Google Cloud restent pertinents, notamment dû à l'infrastructure et la certitude de service derrière.

AWS et la gestion des API

AWS est la sous-branche d'Amazon gérant les produits et services liés au web comme les hébergements VPS (avec EC2), cloud, les CAAS, PAAS, SAAS, les systèmes d'authentification, les services de stockages en ligne (S3), les bases de données, et de multiples autres solutions. L'écosystème de AWS fait qu'il est naturellement possible de stocker une application entière avec l'intégralité de ses composants sur leurs services.

Historiquement, AWS est en fonctionnement depuis 2006 pour le public, mais existe depuis 2002, pendant cette période, les outils proposés par le service étaient très restreints, contrairement à aujourd'hui où AWS est devenu un fournisseur de service web que l'on pourrait désigner de couteau suisse.

À l'heure actuelle, les services AWS reposent sur 22 data-centers dont 4 en Europe pour leurs principaux services.

Comme cité précédemment, AWS possède sa solution de gestion d'API en SAAS et cela via sa solution API Gateway, présente à cette adresse : [aws¹](https://aws.amazon.com/fr/api-gateway). Cette solution permet un déploiement, une gestion et une mise en place d'une API de façon simple et optimisée.

Cette solution a pour principal avantage une gratuité des services à hauteur de 1 million d'appels API reçus par mois, et cela pendant 12 mois.

Les prix ensuite dépendent de différents facteurs, notamment la région de serveur exploitée, le type d'API utilisée et les frais supplémentaires applicables, par exemple le transfert de données externes.

Exemple

Si vous possédez une API HTTP hébergée sur le data-center sur Paris et que cette dernière reçoit 7 000 000 de requêtes par mois avec une réponse de 10 ko.

Le prix se calcule ainsi :

- **Prix :** l'appel API coûte 1,17 USD par million entre 1 et 300 millions d'appels.

Les frais de transfert de données externe sont pris en charge à hauteur de 512 ko pour chaque requête (au-delà, la requête est considérée comme double).

- **Calcul :**

- Taille par requête 10 ko / 512 = 0,020, arrondi = 1 requête
- Prix par requête = 1,17 / 1 000 000 = 0,000 001 17 USD
- 7 000 000 requêtes x 1 facturable x 0,000 001 17 USD = 8,19 USD

Le coût du service sera donc de 8,19 USD.

Remarque

Les calculs pouvant vite être compliqués à effectuer pour évaluer le coût du service, AWS met à disposition un simulateur permettant de calculer les frais à l'adresse : [aws²](https://calculator.aws/#/createCalculator/APIGateway)

Afin de pouvoir utiliser le service de AWS, il est nécessaire de se créer un compte, cette étape est possible dès la page de présentation du service API Gateway grâce au bouton « *Créer un compte AWS* », à partir du formulaire, il est alors possible de créer un compte en suivant les étapes.

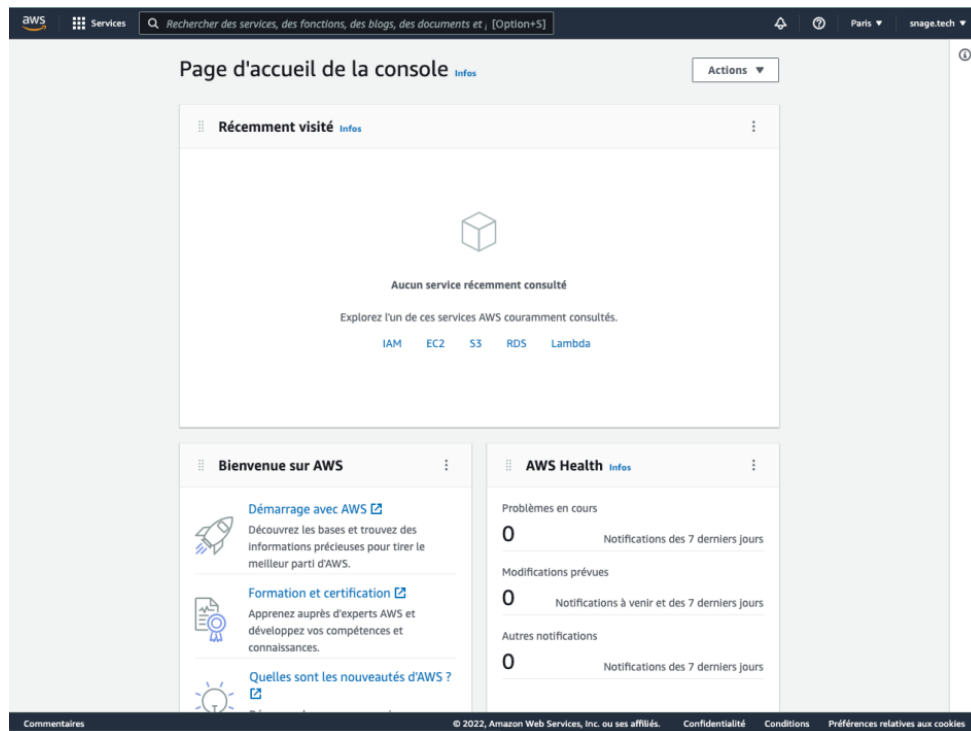
Attention

Bien que l'utilisateur profite de l'offre gratuite de AWS, la création de compte chez ce dernier nécessite obligatoirement un moyen de paiement, dans la plupart des cas, le montant de 1 € est prélevé sur la carte puis restitué.

1 <https://aws.amazon.com/fr/api-gateway>

2 <https://calculator.aws/#/createCalculator/APIGateway>

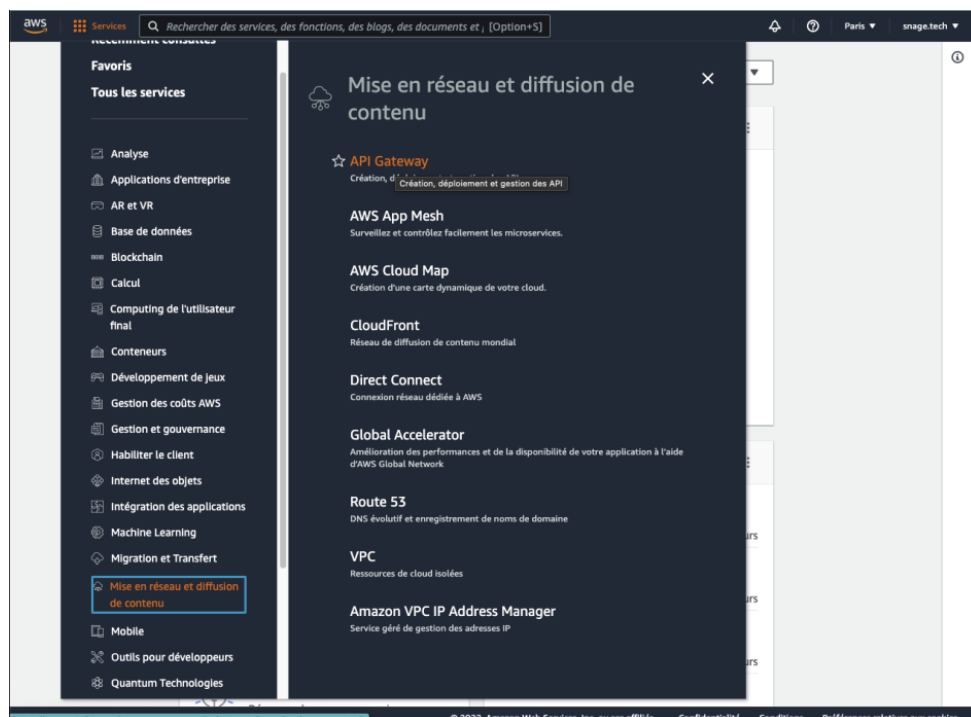
Une fois le compte créé, l'utilisateur en se connectant se retrouve sur le tableau de bord du service AWS similaire à celui-ci dessous :



Capture d'écran tableau de bord AWS

Il est alors possible de créer une API via le service API Gateway, mais avant cela, si l'utilisation de l'API est pour un public européen, il est recommandé de modifier le data-center en haut à droite et en sélectionner un européen (dans la capture dessus, le data-center de l'Ohio a été remplacé par celui de Paris).

Une fois le data-center voulu sélectionné, il est nécessaire d'aller dans la partie Services en haut à gauche, de sélectionner la catégorie « Mise en réseau et diffusion de contenu » et de choisir API Gateway comme ci-dessous.



Capture d'écran AWS, choix du service

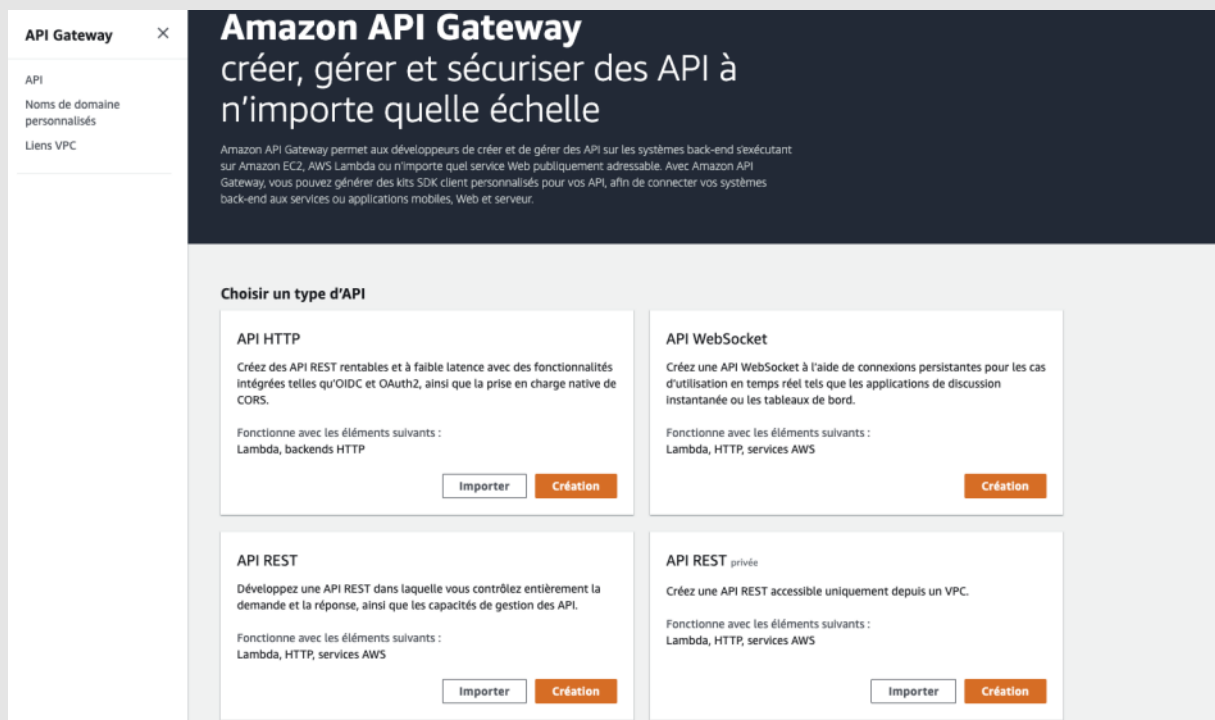
À partir de la page qui suit, la création d'une API via le service API Gateway débute.

Méthode

Sur la page qui s'affiche, Amazon laisse le choix entre les différents types d'API que son service propose :

- HTTP
- WebSocket
- REST Public
- REST Private

C'est à partir de cette page qu'il est possible de choisir quel sera le type d'API créé, comme sur la capture d'écran ci-dessous :

**Légende : capture d'écran AWS premier écran de création d'une API avec Gateway**

Pour cette présentation, il s'agira d'une API REST publique. Cette dernière sera totalement gratuite jusqu'à 1 000 000 de requête, dans le cas où elle n'utilisera pas de mémoire-cache et ce pendant 12 mois. Il faut donc appuyer sur le bouton Création sous API REST (celui présent à gauche, l'autre étant une API REST privée).

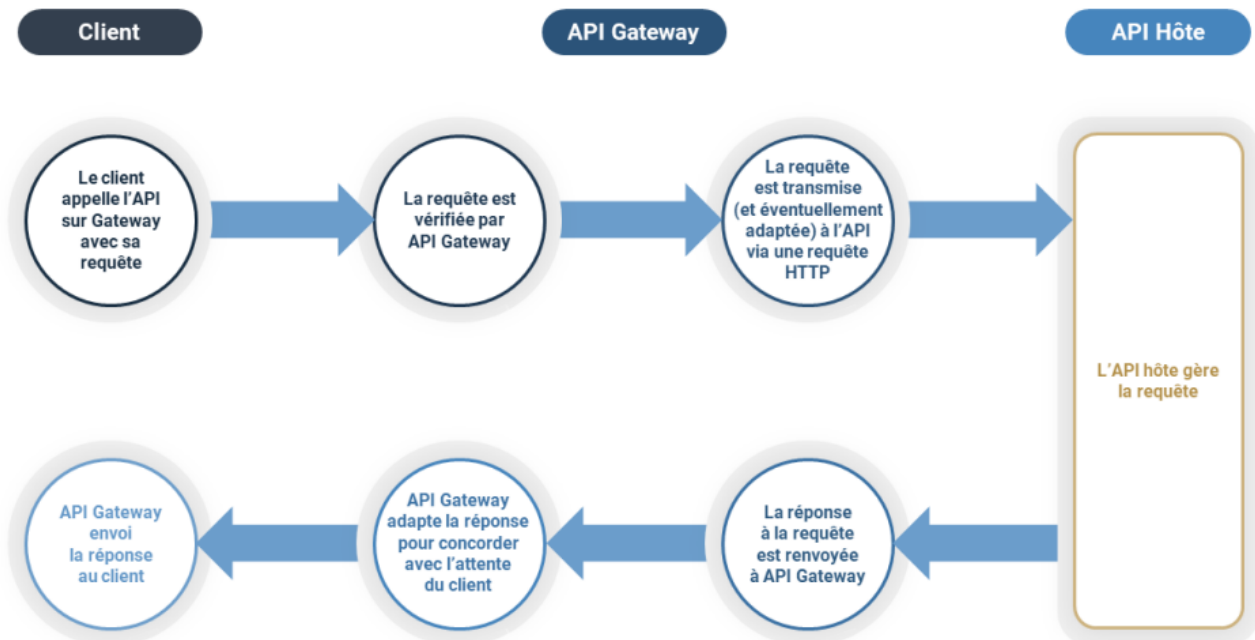
Amazon affiche alors une fenêtre avec :

- Le type de protocole (ici REST)
- Le choix de la méthodologie (de base Exemple d'API), et
- Le résultat

Ainsi, Amazon propose dès le départ un Exemple d'API via une solution PetStore.

API Gateway via HTTP Request

Cette solution (PetStore) et les méthodes http sont en réalité une méthodologie d'API faisant appel à une autre API, c'est l'une des possibilités de Gateway (ainsi API Gateway gère le trafic et la sécurité, notamment en vérifiant la conformité des requêtes vers la seconde API) suivant le schéma ci-dessous :



Ainsi API Gateway peut avoir plusieurs rôles dans ce cas, notamment :

- Protéger l'API Hôte en filtrant les requêtes,
- Vérifier la conformité des requêtes,
- Gérer l'authentification des requêtes,
- Adapter le format des requêtes,
- Adapter le format de la réponse.

La sécurité doit aussi être gérée du côté API hôte, mais API Gateway ajoute, alors, une couche à cette dernière.

Pour l'utilisation d'API Gateway en tant qu'API complète, il faut la combler avec la solution Lambda d'AWS et partir d'une API vide (avec le choix « *Nouvelle API* »). Cette dernière permet de créer du code pour une fonction précise et cette fonction peut ensuite être appelée par API Gateway.

Exemple

Créons une API Hello World en JavaScript à partir d'API Gateway et Lambda.

Dans un premier temps nous allons créer l'API sur le service Gateway en sélectionnant API REST et en sélectionnant ensuite « *Nouvelle API* » et en la nommant HelloWorldAPI comme ci-dessous, puis en cliquant sur « *Créer une API* ».

Amazon API Gateway API > Créer

Afficher tous les indicateurs ?

Sélectionner le protocole

Sélectionnez le type d'API (API REST ou API WebSocket) que vous souhaitez créer.

☒ REST ☐ WebSocket

Créer un nouveau API

No Amazon API Gateway, uma API REST consiste em uma coleção recursos e métodos que podem ser invocados por meio de endpoints HTTPS.

☒ Nouvelle API ☐ Importer à partir de Swagger ou d'Open API 3 ☐ Exemple d'API

Paramètres

Choisissez un nom descriptif et une description pour votre API.

Nom d'API* HelloWorldAPI

Description

Type de point d'extrémité Régional

* Obligatoire

Créer une API

Capture d'écran AWS création de l'API Hello World

À présent l'API est créée.

En retournant dans les services AWS via le bouton services en haut à gauche, nous allons nous rendre dans le service Lambda à partir de la catégorie Calcul, puis nous créerons une nouvelle fonction avec le bouton créer une fonction.

Nous nommerons cette fonction HelloWorld et nous la mettrons sur un environnement d'exécution Node.js comme sur la capture ci-après.

Créer une fonction

Choisissez l'une des options suivantes pour créer votre fonction.

Créer à partir de zéro

Commencez avec un exemple Hello World simple.

Utiliser un plan

Créez une application Lambda à partir d'un exemple de code et de préférences de configuration pour les cas d'utilisation courants.

Image de conteneur

Sélectionnez une image de conteneur à déployer pour votre fonction.

Parcourir le référentiel d'applications serverless

Déployez une exemple d'application Lambda à partir du référentiel AWS Serverless Application Repository.

Informations de base

Nom de la fonction

Entrez un nom qui décrit l'objectif de votre fonction.

HelloWorld

Utilisez uniquement des lettres, des chiffres, des traits d'union ou des traits de soulignement, sans espace.

Exécution

Choisissez le langage à utiliser pour écrire votre fonction. Notez que l'éditeur de code de la console prend uniquement en charge Node.js, Python et Ruby.

Node.js 14.x

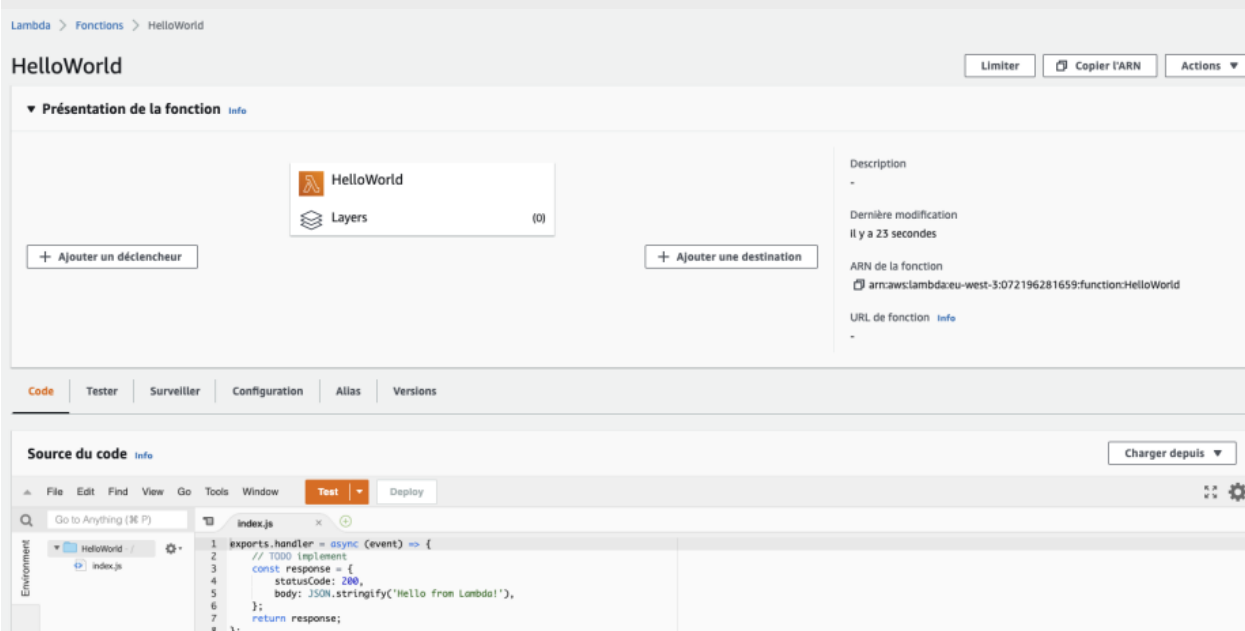
Autorisations

Par défaut, Lambda créera un rôle d'exécution avec les autorisations de charger des journaux dans Amazon CloudWatch Logs. Vous pouvez personnaliser ce rôle par défaut plus tard lors de l'ajout de déclencheurs.

► Modifier le rôle d'exécution par défaut

Capture d'écran Création de la fonction Lambda AWS

Nous pouvons alors créer notre fonction avec le bouton « *Créer une fonction* ». Le tableau de bord de notre fonction Lambda s'affiche alors avec une partie du code préconçu :



Capture d'écran tableau de bord de la fonction HelloWorld sur AWS

L'index.js retournant un code 200, indique une page bien chargée et le message en corps sous format JSON « *hello from Lambda !* ».

Nous allons modifier ce code pour lui faire dire uniquement « *Hello World* » et obtiendrons le code suivant :

```
1 exports.handler = async (event) => {
2   const response = {
3     statusCode: 200,
4     body: JSON.stringify('Hello World'),
5   }
6   return response
}
```

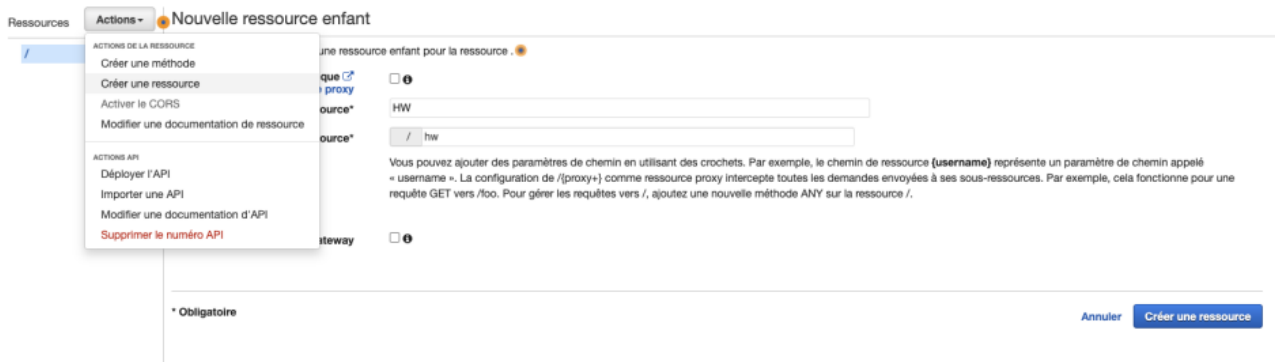
Puis nous la déploierons avec le bouton « *Deploy* », le message « *Mise à jour de la fonction HelloWorld effectuée avec succès* » devrait alors s'afficher dans une bannière en haut de la fenêtre.

Maintenant, nous allons retourner sur API Gateway, toujours via le bouton Services (API Gateway devrait apparaître directement dans la colonne de droite). Nous sélectionnons notre API en cliquant dessus (ici HelloWorldAPI) et une fois dessus, nous appuierons sur le bouton « *Actions* » à côté de « *Ressources* ».

Ici il est important de connaître 2 termes :

- Les **ressources** sont les chemins d'accès,
- Les **méthodes** sont le type de requêtes (POST, GET, PUT, etc.).

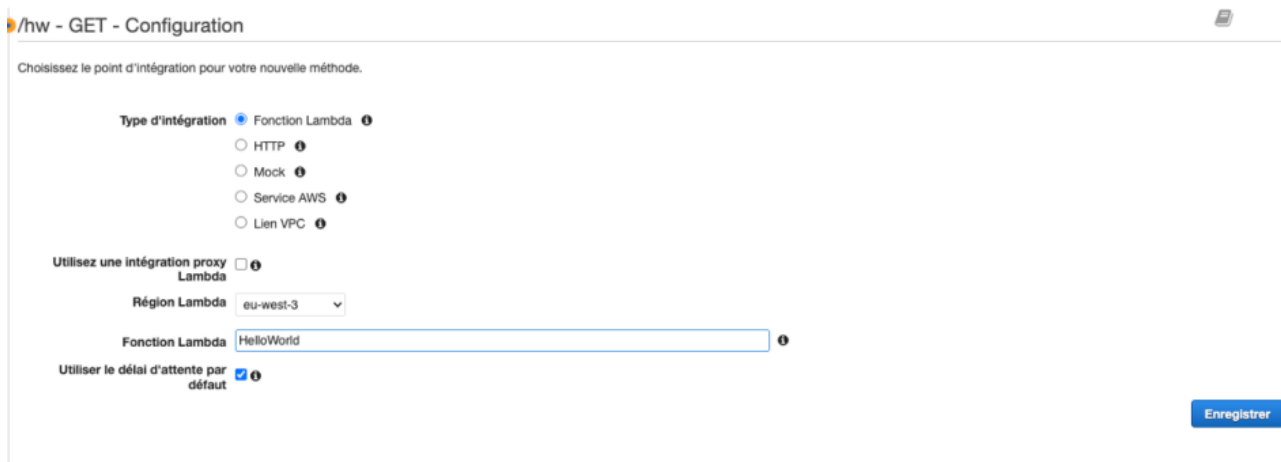
Ainsi, si nous voulons créer une adresse à notre API avec un chemin /HW à appeler par la méthode GET, nous sélectionnons « *Créer une ressource* » dans « *Actions* » puis la configurons comme la capture ci-dessous :



Capture d'écran Création d'une Ressource sur AWS API Gateway

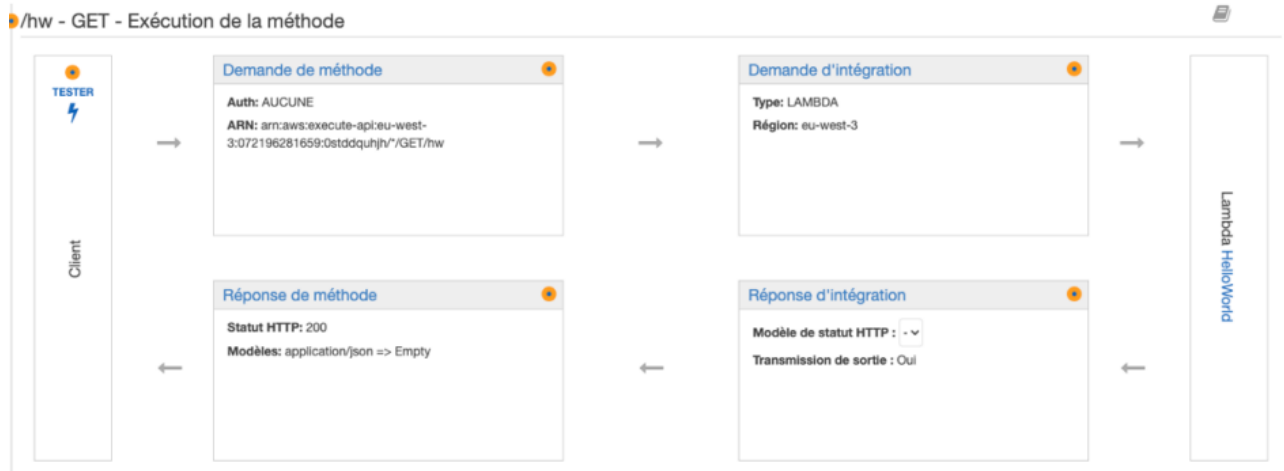
En appuyant sur le bouton bleu « *Créer une ressource* », notre nouvelle ressource apparaîtra dans la liste à gauche. Ensuite, en nous positionnant dessus, nous sélectionnons « *Créer une méthode* » dans « *Actions* », puis sélectionnons « *GET* » et validons.

La page de configuration de la méthode GET pour le chemin /hw s'affiche alors. Pour récupérer notre message, nous sélectionnons comme type d'intégration « *Fonction Lambda* » et fournissons le nom de la fonction que nous avons créée comme ceci :



Capture d'écran Configuration de la méthode sur AWS API Gateway

En validant avec le bouton « Enregistrer », l'appel à l'adresse /hw par la méthode GET sera alors associé à notre fonction HelloWorld sur le service Lambda. AWS devrait alors schématiser le fonctionnement des requêtes ainsi :



Capture d'écran Schéma de fonctionnement de l'appel sur AWS API Gateway

Il est alors possible de tester notre API en appuyant sur le bouton « Tester » au niveau du Client, la page devrait alors afficher les éléments que notre fonction Lambda devait fournir par défaut : un code 200 et le message Hello World.

Il est aussi possible de tester en déployant l'API (« Actions » puis « Déployer l'API » et ensuite nommer l'étape de déploiement et valider) puis en effectuant un appel à l'adresse fournie en haut de page, en ajoutant /hw à la fin.

```
{"statusCode":200,"body":"\nHello World\n"}
```

Capture d'écran Résultat déploiement API Hello World avec Lambda

Remarque

Évidemment AWS API Gateway propose de nombreuses autres fonctions, il est d'ailleurs possible de trouver de nombreuses ressources de fonctionnement à l'adresse du guide du développeur de ce service : [aws¹](https://docs.aws.amazon.com/fr_fr/apigateway/latest/developerguide/). Cette documentation permet d'appréhender toutes les fonctions et méthodes du service API Gateway d'Amazon.

Exercice : Quiz

[solution n°1 p.23]

Question 1

¹ https://docs.aws.amazon.com/fr_fr/apigateway/latest/developerguide/

SAAS signifie :

- ☐ Software As A Service
- ☐ System As A Service
- ☐ Service As A System
- ☐ Solution As A Service

Question 2

L'un des avantages des API via SAAS est :

- ☐ Le coût indépendant du nombre de requêtes
- ☐ La facilité de déploiement et de sécurisation de l'API

Question 3

AWS existe depuis :

- ☐ 2002
- ☐ 2003
- ☐ 2004
- ☐ 2005

Question 4

Les protocoles supportés par API Gateway sont :

- ☐ HTTP
- ☐ REST
- ☐ Web Socket
- ☐ FTP

Question 5

L'utilisation d'une source http par API Gateway permet :

- ☐ De sécuriser les appels à l'API hôte
- ☐ De rendre les requêtes conformes
- ☐ D'effectuer les enregistrements en base de données

III. SAAS et API avec Google Cloud

Google Cloud est le service dédié au développeur pour les services web du groupe Google.

Existant depuis 2011, la grande force de Google Cloud, dont le nom complet est Google Cloud Platform, est de fournir aux développeurs et entreprises une infrastructure similaire à celle qu'ils utilisent pour leur moteur de recherche. Cela fait gage de qualité des serveurs proposés par la firme.

Google Cloud, comme AWS propose une solution de gestion d'API pour les entreprises avec leur service Apigee.

Ce service propose une gestion des API, un portail développeur dédié et une monétisation de ces API. La solution est utilisée par des sociétés comme AccuWeather, AutoDesk, Citrix et plusieurs autres. Une solution annexe pour les plus petites entreprises ou particuliers est proposée nommée Service Management API.

Ces 2 solutions peuvent être alliées avec la solution SAAS App Engine déployant l'API comme un service web.

Les services Google Cloud Platform sont payants contrairement au service d'AWS avec leur offre gratuite sur 12 mois.

Cependant, Google Cloud Platform offre entre 300 et 400 \$ de crédit pour essayer leurs solutions pendant 90 jours et propose un tracking gratuit de 1 million d'appels API par mois sans durée.

Pour cela, il suffit de se rendre sur la page de la solution de gestion d'API de Google Cloud Platform : Google Cloud¹, puis de cliquer sur le bouton « *Profiter d'un essai gratuit* ».

La page de création de comptes s'affiche alors ou la page de connexion à un compte Google puis celle de création de comptes Google Cloud Platform.

Profiter d'un essai gratuit de Google Cloud

Étape 1 sur 2 Informations de compte

J [CHANGER DE COMPTE](#)

Bonne nouvelle ! Vous pouvez bénéficier d'un crédit d'essai gratuit supplémentaire de 100,00 \$ pour un total de 400,00 \$. Vous recevrez ces crédits dans les 24 heures suivant votre inscription.

Pays

France

Laquelle de ces propositions décrit le mieux votre organisation ou vos besoins ?

Please select
Project personnel

Conditions d'utilisation

☒ J'ai lu et j'accepte les [Conditions d'utilisation relatives à l'essai gratuit de Google Cloud Platform](#).

Vous devez cocher cette case pour continuer

CONTINUER

Accès à tous les produits Cloud Platform

Bénéficiez de toutes les ressources dont vous avez besoin pour créer et exécuter vos applications, vos sites Web et vos services, y compris Firebase et l'API Google Maps.

300 \$ de crédit offert

Profitez des fonctionnalités de Google Cloud avec un crédit de 300 \$ à dépenser au cours des 90 prochains jours.

Aucun prélèvement automatique après l'essai gratuit

Nous vous demandons d'indiquer les informations de votre carte de paiement pour vérifier que vous n'êtes pas un robot. Votre compte ne sera pas débité, sauf si vous décidez de passer manuellement à un compte payant.

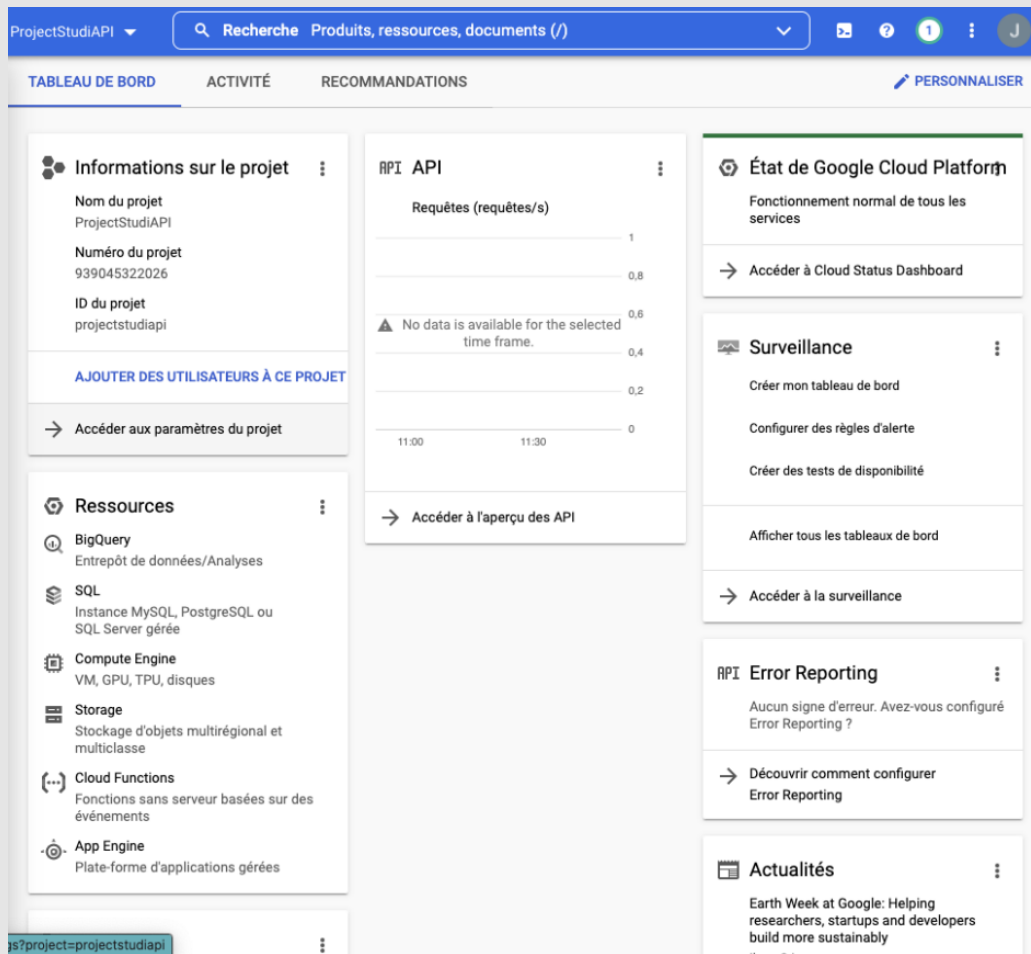
Capture d'écran création de comptes Google Cloud Platform

Google Cloud Platform demande, comme AWS, des coordonnées bancaires pour débloquer l'essai gratuit.

Méthode


La première étape après l'accès au compte Google Cloud Platform est de créer un projet via le bouton « *Nouveau Projet* », le nommer, éventuellement sélectionner une organisation (dans le cas actuel, ce n'est pas nécessaire) et valider avec le bouton créer. L'interface du projet s'affiche alors comme ci-dessous.

¹ <https://cloud.google.com/solutions/new-channels-using-apis?hl=fr>



Capture d'écran Tableau de bord projet Google Cloud

Ensuite il est nécessaire d'activer l'API, pour cela il suffit d'accéder au Marketplace (via la barre de menu à gauche) et rechercher Service Management API, le premier résultat est alors l'API permettant d'en créer et configurer, il suffit alors de l'activer.



Service Management API

Google Enterprise API

Google Service Management allows service producers to publish their services on Google Cloud...

GÉRERESSAYER CETTE API

API activée

APERÇUDOCUMENTATION

Présentation

Google Service Management allows service producers to publish their services on Google Cloud Platform so that they can be discovered and used by service consumers.

Plus d'infos

Type: [SaaS & APIs](#)

Dernière mise à jour: 23/07/2021

Catégorie: [Google Enterprise APIs](#)

Nom du service: servicemanagement.googleapis.com

Capture d'écran de l'API Service Management API de Google Cloud Platform

En cliquant ensuite sur « Gérer », Google renvoie sur la page de l'API SMA, un message en haut de page vous indique que pour utiliser l'API il est nécessaire de lui créer des identifiants. Pour cela cliquez sur le bouton « Créer des identifiants », s'affiche alors la page de création d'identifiants comme ci-dessous.

RPI API et services

- API et services activés
- Bibliothèque
- Identifiants**
- Écran de consentement OAuth...
- Validation de domaine
- Page des accords d'utilisation

Créer des identifiants

Quelle API utilisez-vous ?

Les différentes API utilisent des plates-formes d'authentification variées. En outre, l'accès de certains identifiants peut être limité à des API spécifiques.

Sélectionner une API *

Service Management API

À quelles données allez-vous accéder ? *

En fonction du type de données auquel vous souhaitez accéder, différents identifiants sont requis afin d'autoriser l'accès. [En savoir plus](#)

!
L'accès à cette API Google Cloud Platform se fait généralement depuis un serveur qui utilise un compte de service. Pour créer un compte de service, sélectionnez "Données de l'application".

☐
Données utilisateur

Données appartenant à un utilisateur Google, comme son adresse e-mail ou son âge. Consentement de l'utilisateur requis. Un client OAuth sera créé.

☒
Données de l'application

Données appartenant à votre application, par exemple son backend Cloud Firestore. Un compte de service sera créé.

Comptez-vous utiliser cette API conjointement à Compute Engine, Kubernetes Engine, App Engine ou Cloud Functions ?

Les applications s'exécutant sur GCE, GKE, GAE et GCF peuvent utiliser les identifiants par défaut de ces produits. La création d'un identifiant n'est donc pas nécessaire.

☒
Oui, j'utilise un ou plusieurs de ces produits.
☐
Non, je ne les utilise pas.

Capture d'écran Google Cloud Platform : création des identifiants

Ici l'API à sélectionner est « *Service Management API* », le type de donnée est « *Données de l'application* » (cela signifie que les données proviendront du projet et non de l'utilisateur connecté) et ensuite il est possible de spécifier l'utilisation d'un service annexe de GCP.

Ce service est optionnel, mais peut être utile pour gérer la sécurité des requêtes.

Exemple

Nous pouvons à présent créer une application API via le service App Engine de Google, pour cela il est nécessaire de se rendre à nouveau dans le MarketPlace et chercher App Engine puis cliquer sur « Accéder à App Engine ».

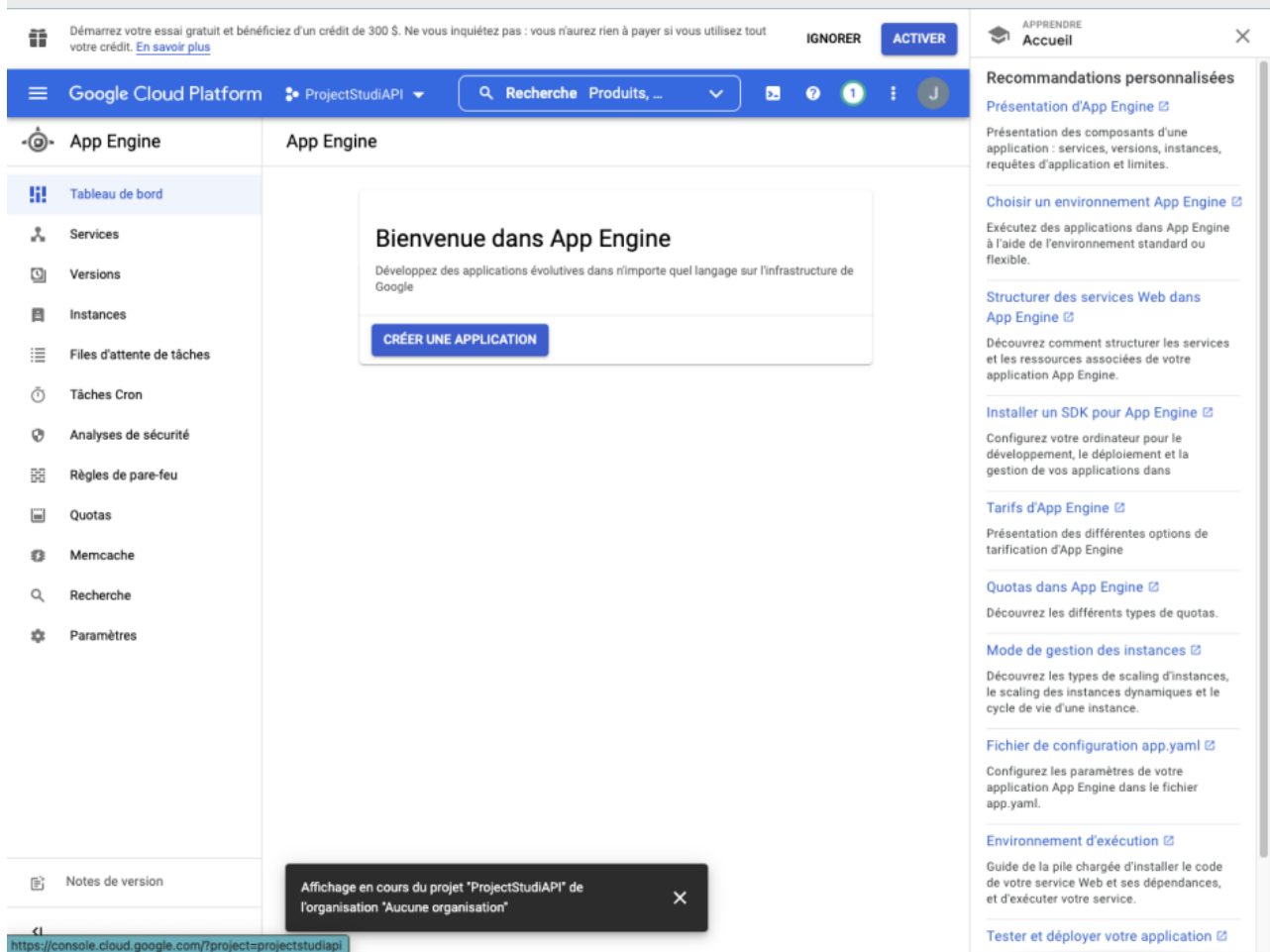


Tableau de bord App Engine GCP

Nous allons créer une application que nous positionnerons sur le serveur europe-west6 (serveur le plus proche de la France).

Comme pour l'exemple précédent avec AWS, nous créerons une application en JavaScript grâce à node.js. Pour cela, Google Cloud fournit un SDK qu'il est nécessaire d'installer, la méthode la plus simple pour le faire est de se référer au guide fourni par Google à l'adresse :

Google Cloud¹.

Ainsi, le SDK est installé facilement et rapidement, puis il suffit de l'initialiser avec la commande : « *gcloud init* » en liant le compte Google Cloud et le projet créé précédemment.

Maintenant nous allons créer le code permettant de gérer l'appel de l'API, pour cela nous créerons un dossier « *helloworld* » et réaliserons un « *npm init -y* » à l'intérieur (ici les commandes sont spécifiques à node et peuvent être remplacées par du code dans un autre langage back-end). Après l'initialisation du projet node, nous installons l'extension express (permettant d'utiliser notre code comme réponse serveur) grâce à la commande : « *npm i express* ».

¹ <https://cloud.google.com/sdk/docs/install?hl=fr>

Nous créerons un fichier `server.js` et adaptons la partie scripts du fichier `package.json` ainsi :

```
1 "scripts": {
2   "test": "echo \"Error : no test specified\" && exit 1",
3   "start": "node server.js"
4 }
```

Puis nous insérons le code qui renverra notre « *Hello World* » grâce au fichier `server.js` :

```
1 const express = require('express')
2 const app = express()
3
4 app.get('/', (req, res) => {
5   res.send('Hello World')
6 })
7
8 app.listen(8080)
```

Avec ce code nous spécifions que les requêtes sans chemin sur le port 8080 auront comme réponse « *Hello World* ».

Enfin, nous créerons dans notre dossier un fichier qui indiquera la méthode de lancement à GCP avec un fichier `app.yaml`, comme ceci :

```
1 runtime : nodejs14
```

Il est alors possible de déployer l'application grâce à la commande (depuis le dossier du projet) :

```
1 $ gcloud app deploy
```

Il faut alors vérifier que le déploiement vise le bon projet et valider avec « *Y* ».

À partir de là, l'API est déployée via un service SAAS de Google Cloud Platform (bien que le format de déploiement soit en réalité un hybride CAAS / SAAS, car l'application est déployée dans un container dont certaines valeurs peuvent être modifiées).

Il est possible de voir la réponse grâce à la commande :

```
1 $ gcloud app browse
```

Cette commande affichera une page web renvoyant le code qui a été créé.

Comme la plupart des cas dans le développement, il est recommandé de se baser sur la documentation disponible à cette adresse : Google Cloud¹.

Remarque

Si un message d'erreur d'accès non configuré apparaît pendant le déploiement, il peut être nécessaire d'activer l'API Cloud Build depuis le Marketplace pour le projet.

Exercice : Quiz

[solution n°2 p.24]

Question 1

Le service de gestion d'API pour les entreprises de Google Cloud s'appelle :

- ☐ Apigee API Gateway
- ☐ API Gateway

Question 2

¹ <https://cloud.google.com/appengine/docs?hl=fr>

Google Cloud Platform propose pour tester ces services :

- ☐ 12 mois gratuits
- ☐ Un crédit de 300 à 400 \$

Question 3

La solution SAAS pour déployer une API avec GCP se nomme :

- ☐ SAAS API
- ☐ App Engine
- ☐ SAAS Deployment

Question 4

Pour fonctionner avec un ordinateur Windows ou Mac, Google passe par :

- ☐ Un SDK
- ☐ Une application

Question 5

La commande pour déployer une application avec le SDK GCP est :

- ☐ Gcloud app deploy
- ☐ Gcloud deploy

V. Essentiel

Les API sont un élément essentiel des applications web existantes aujourd'hui, elles permettent de concevoir des solutions dynamiques pour partager des données avec la partie visible d'un site ou d'une application, mais aussi de communiquer avec des services externes.

Pour le déploiement de ces dernières, plusieurs solutions existent, notamment le déploiement sur un service SAAS favorisant la facilité de déploiement, la sécurité et la surveillance de l'API. Amazon et Google possèdent alors chacun leur branche pour proposer ce service d'hébergement et déploiement, notamment avec Amazon Web Service ou AWS pour Amazon et Google Cloud Platform ou GCP pour Google. Ainsi les 2 marques permettent, au moyen de leurs services, de déployer une API appelant par exemple le code sur un serveur, de façon simple et sécurisée, moyennant des frais au taux d'appels à l'API.

Une autre méthodologie est d'héberger la logique derrière l'API sur un autre service de la même marque notamment avec App Engine pour Google et Lambda pour Amazon. Ainsi l'API communique avec des éléments internes à Amazon ou Google sans utiliser de point extérieur.

VI. Auto-évaluation

A. Exercice

Dans une entreprise développant des API à réponse simple, vous êtes développeur et devez estimer les frais de l'API.

Question 1

[solution n°3 p.25]

Votre responsable vous fournit les valeurs suivantes afin d'évaluer le coût de déploiement en SAAS avec API Gateway et Lambda de l'API :

- Nombre de requêtes par mois : 9 623 521
- Taille moyenne des requêtes : 265 Ko
- Type de requête : REST
- Serveur : Paris
- Sans mémoire cache
- Le service Lambda sera gratuit

Quel sera le montant mensuel estimé pour l'utilisation de AWS ?

Question 2

[solution n°4 p.25]

Dans cette même entreprise, le responsable vous demande de concevoir la base sur API Gateway de l'API nommée « *APIFirst* » en créant 2 chemins : admins et users, avec pour chaque chemin des méthodes GET / POST et DELETE (les méthodes resteront vides).

Combien de ligne compte alors les Ressources de l'API ?

B. Test

Exercice 1 : Quiz

[solution n°5 p.27]

Question 1

Le service permettant d'utiliser du code comme fonction sur AWS s'appelle :

- ☐ Lambda
- ☐ Fonctionner
- ☐ MaterCoder
- ☐ FunctionGo

Question 2

Lors d'une utilisation d'API Gateway avec Lambda, le client envoie sa requête :

- ☐ Directement au code Lambda
- ☐ Au service API Gateway

Question 3

API Gateway d'AWS fonctionne avec un système de :

- ☐ Ressources
- ☐ Méthodes
- ☐ Chemin
- ☐ Protocoles

Question 4

Les services sur Google Cloud Platform s'ajoutent grâce :

- ☐ Au marketplace
- ☐ À un onglet service

Question 5


La commande pour voir un site déployé en SAAS avec App Engine est :

- ☐ Gcloud app browse
- ☐ Gcloud see project

Solutions des exercices


Exercice p. 12 Solution n°1**Question 1**

SAAS signifie :

- ☒ Software As A Service
- ☐ System As A Service
- ☐ Service As A System
- ☐ Solution As A Service
-  SAAS signifie Software As A Service.


Question 2

L'un des avantages des API via SAAS est :

- ☐ Le coût indépendant du nombre de requêtes
- ☒ La facilité de déploiement et de sécurisation de l'API
-  L'avantage du déploiement via une solution SAAS est la facilité de déploiement et sécurisation de cette dernière.

Question 3

AWS existe depuis :

- ☒ 2002
- ☐ 2003
- ☐ 2004
- ☐ 2005
-  AWS existe depuis 2002, il n'est cependant vraiment ouvert au public que depuis 2006.

Question 4

Les protocoles supportés par API Gateway sont :

- ☒ HTTP
- ☒ REST
- ☒ Web Socket
- ☐ FTP
-  API Gateway supporte les requêtes HTTP, REST et web Socket.


Question 5

L'utilisation d'une source http par API Gateway permet :

☒ De sécuriser les appels à l'API hôte

☒ De rendre les requêtes conformes

☐ D'effectuer les enregistrements en base de données

 L'utilisation des end point http avec API Gateway permet de rediriger les requêtes vers une API hôte, ainsi cette dernière peut être sécurisée par API Gateway et être mise en conformité par le service.


Exercice p. 18 Solution n°2

Question 1

Le service de gestion d'API pour les entreprises de Google Cloud s'appelle :

☒ Apigee API Gateway

☐ API Gateway


 Le service de gestion d'API de Google Cloud s'appelle Apigee.

Question 2

Google Cloud Platform propose pour tester ces services :

☐ 12 mois gratuits

☒ Un crédit de 300 à 400 \$

 GCP fournit un crédit de 300 à 400 \$ pour tester ses services.


Question 3

La solution SAAS pour déployer une API avec GCP se nomme :

☐ SAAS API

☒ App Engine

☐ SAAS Deployment


 La solution SAAS pour déployer une API avec GCP se nomme App Engine.

Question 4

Pour fonctionner avec un ordinateur Windows ou Mac, Google passe par :


☒ Un SDK

☐ Une application

 Google passe par le SDK Google Cloud pour fonctionner avec l'environnement local.

Question 5

La commande pour déployer une application avec le SDK GCP est :

- ☒ Gcloud app deploy
- ☐ Gcloud deploy
-  La commande pour déployer une application avec le SDK GCP est gcloud app deploy.

p. 20 Solution n°3

En se rendant sur le site de simulation à l'adresse suivante : [aws pricing calculator](#)¹, on peut calculer le montant mensuel grâce à l'interface qui répondra le calcul suivant :

- 9 623 521 requêtes x 1 multiplicateur d'unités = 9 623 521 nombre total de requêtes d'API REST
- *Tiered price for* : 9 623 521 requêtes
- 9 623 521 requêtes x 0,000 003 500 0 USD = 33,68 USD
- Coût total de l'offre = 33,682 3 USD (demandes d'API REST)
- Prix total progressif des demandes d'API REST : 33,682 3 USD
- 0 USD par heure x 730 heures dans un mois = 0,00 USD pour la mémoire cache
- Prix total de la mémoire cache dédiée : 0,00 USD
- Coût de l'API REST (mensuellement) : 33,68 USD

Le coût mensuel sera donc de 33,68 USD / mois.

p. 20 Solution n°4

¹ <https://calculator.aws/#/createCalculator/APIGateway>

Pour créer la base de l'API, il faut se rendre sur le service API Gateway via l'interface développeur et le menu « *Services en haut* » à gauche. Puis il faut cliquer sur « *Créer une API* », le bouton « *Création* » dans la carte API REST et nommer l'API APIFirst en sélectionnant « *Nouvelle API* », et en finissant par appuyer sur « *Créer une API* ».

Sélectionner le protocole

Sélectionnez le type d'API (API REST ou API WebSocket) que vous souhaitez créer.

☒ REST ☐ WebSocket

Créer un nouveau API

No Amazon API Gateway, uma API REST consiste em uma coleção recursos e métodos que podem ser invocados por meio de endpoints HTTPS.

☒ Nouvelle API ☐ Cloner à partir d'une API existante ☐ Importer à partir de Swagger ou d'Open API 3
☐ Exemple d'API

Paramètres

Choisissez un nom descriptif et une description pour votre API.

Nom d'API*	<input type="text" value="APIFirst"/>
Description	<input type="text"/>
Type de point d'extrémité	<input type="text" value="Régional"/> ⓘ

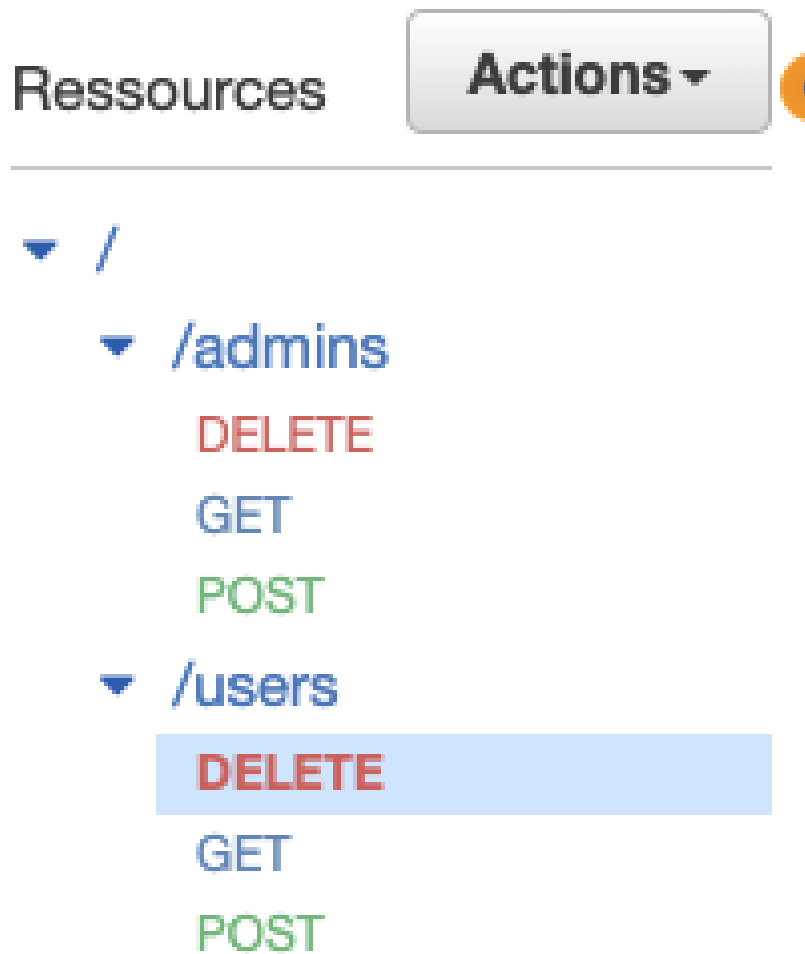
* Obligatoire

[Créer une API](#)

Capture d'écran Nouvelle API APIFirst AWS

Ensuite il suffit d'appuyer sur le bouton « *Actions* » et « *Créer les ressources* » admin et users, puis dans chaque ressource, toujours avec le bouton « *Actions* », « *Créer les méthodes* » POST/GET et DELETE.

Les ressources de l'API comptent alors 9 lignes comme visibles sur la capture ci-dessous :



Capture d'écran Ressource API sur AWS

Exercice p. 20 Solution n°5


Question 1

Le service permettant d'utiliser du code comme fonction sur AWS s'appelle :

- ☒ Lambda
- ☐ Fonctionner
- ☐ MaterCoder
- ☐ FunctionGo
- ☐ Le service se nomme Lambda.


Question 2

Lors d'une utilisation d'API Gateway avec Lambda, le client envoie sa requête :

- ☐ Directement au code Lambda
- ☒ Au service API Gateway
-  Le client envoie la requête à API Gateway qui exploite la fonction sur le service Lambda afin de fournir une réponse.


Question 3

API Gateway d'AWS fonctionne avec un système de :

- ☒ Ressources
- ☒ Méthodes
- ☐ Chemin
- ☐ Protocoles
-  API Gateway fonctionne avec des ressources, servant de chemin et des méthodes.


Question 4

Les services sur Google Cloud Platform s'ajoutent grâce :

- ☒ Au marketplace
- ☐ À un onglet service
-  Les services sur GCP s'ajoutent grâce au marketplace disponible sur le tableau de bord.

Question 5

La commande pour voir un site déployé en SAAS avec App Engine est :

- ☒ Gcloud app browse
- ☐ Gcloud see project
-  La commande est gcloud app browse.