



OC Pizza

Système informatique OC Pizza

Dossier d'exploitation

Version 1.1

Auteur

Jacques Colin

Analyste-programmeur

TABLE DES MATIERES

1 - Versions.....	3
2 - Introduction	4
2.1 - Objet du document	4
2.2 - Références	4
3 - Prérequis.....	5
3.1 - Système	5
3.1.1 - Serveur de Base de données	5
3.1.2 - Serveur Web	5
3.2 - Base de données.....	5
3.3 - Web services	7
4 - Procédure de déploiement	9
4.1 - Déploiement de la base de données	9
4.2 - Déploiement de l'application web	10
4.3 - Déploiement des API RESTful	12
4.4 - Déploiement de l'application Android	17
4.4.1 - Signer l'APK.....	17
4.4.2 - Publier l'application dans le Play Store.....	17
5 - Procédure de démarrage / arrêt	18
5.1 - Base de données.....	18
5.2 - Application web / mobile.....	18
5.3 - API RESTful	18
6 - Procédure de mise à jour	19
6.1 - Base de données.....	19
6.2 - Application web	19
6.3 - Application mobile.....	21
6.4 - API RESTful	22
7 - Supervision/Monitoring	23
8 - Procédure de sauvegarde et restauration	24
9 - Glossaire	26



1 - VERSIONS

Auteur	Date	Description	Version
Jacques Colin	13/08/2020	Création du document	1.0
Jacques Colin	26/08/2020	Ajout de compléments	1.1

2 - INTRODUCTION

2.1 - Objet du document

Le présent document constitue le dossier d'exploitation de l'application OC Pizza.

L'objectif du document est d'informer l'équipe technique de la société OC Pizza des procédures à suivre afin de déployer, lancer/arrêter, mettre à jour et de superviser l'application.

2.2 - Références

Pour de plus amples informations, se référer :

1. **DCT – 1.1**: Dossier de conception technique de l'application
2. **DCF – 1.1**: Dossier de conception fonctionnelle de l'application
3. **PVL** : Procès-verbal de livraison

3 - PREREQUIS

3.1 - Système

3.1.1 - Serveur de Base de données

Le serveur de base de données est PostgreSQL 12.4.

PostgreSQL est un puissant SGBD relationnel orienté objet et open source. Avec ses 30 ans de développement actif, PostgreSQL s'est forgé une réputation de SGBD puissant, robuste, fiable et performant.

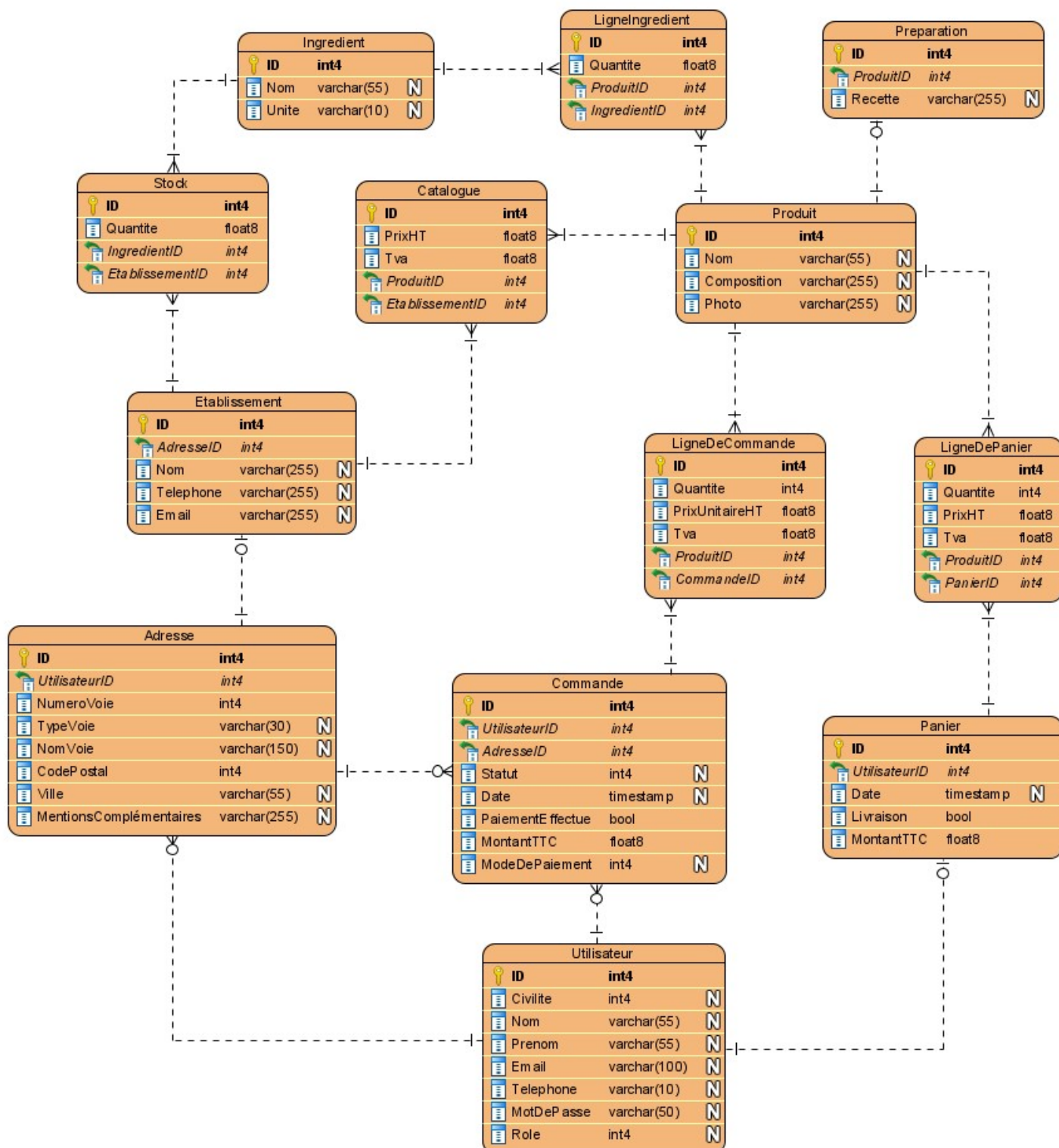
Il sera géré via le service Amazon RDS et les identifiants seront stockés via le service AWS Secret Manager. Un script SQL devra être lancé pour générer les tables.

3.1.2 - Serveur Web

L'application web sera hébergée avec Amazon S3. Elle représente le front de l'application web et mobile.

3.2 - Base de données

La base de données respecte le MPD suivant :



MPD de la base de données

3.3 - Web services

Les web services d'Amazon suivants doivent être accessibles et à jour :

- Amazon S3
 - AWS IAM
 - Amazon CloudFront
 - Route 53
 - Amazon RDS
 - Amazon API Gateway
 - AWS Backup
 - AWS Management Console
 - Amazon Lambda
-
- Amazon S3 est un service de stockage d'objet permettant une scalabilité, une disponibilité des données, une sécurité et des performances de pointe.
 - Amazon CloudFront est un réseau rapide de diffusion de contenu (CDN) qui distribue en toute sécurité certaines données de notre instance d'Amazon S3 aux utilisateurs de l'application, avec une faible latence et des vitesses de transfert élevées. Son but est d'optimiser notre instance d'Amazon S3.
 - Amazon API Gateway est un service permettant aux développeurs de créer, publier, gérer, surveiller et sécuriser facilement des API dans notre cas RESTful.
 - Amazon RDS est un service permettant d'installer, de gérer et de mettre à l'échelle facilement une base de données relationnelle dans le cloud. Dans notre cas la base de données relationnelle est PostgreSQL.
 - AWS IAM permet de contrôler de façon sécurisée l'accès aux services et ressources AWS mais aussi de créer et gérer des utilisateurs ainsi que des groupes AWS, et configurer des autorisations afin de leur permettre ou non d'accéder aux ressources AWS. C'est un service gratuit d'AWS.
 - Amazon Route 53 est un service Web de DNS conforme au protocole IPv6.
 - AWS Backup est un service web qui permet de sauvegarder et restaurer des données sur



l'ensemble des services AWS de façon chiffrée.

- AWS Management Console est le tableau de bord des services web Amazon.
- Amazon Lambda permet d'exécuter du code sans avoir à mettre en service ou gérer des serveurs.

Les services Amazon sont disponibles via le compte :

Nom d'utilisateur : OCPIZZA

Mot de passe : XXXX

4 - PROCEDURE DE DEPLOIEMENT

4.1 - Déploiement de la base de données

Pour déployer la base de données il faudra utiliser Amazon RDS, le dernier pilote JCBD de PostgreSQL et le client SQL « SQL WorkBench » ainsi que ses packages et ses bibliothèques facultatives si l'on préfère SQL Workbench à pgAdmin.

D'abord il faut créer l'instance Amazon RDS, ce tutoriel explique en détail avec des captures d'écran chaque étape de la procédure (cf.

https://docs.aws.amazon.com/fr_fr/AmazonRDS/latest/UserGuide/CHAP_GettingStarted.CreatingConnecting.PostgreSQL.html).

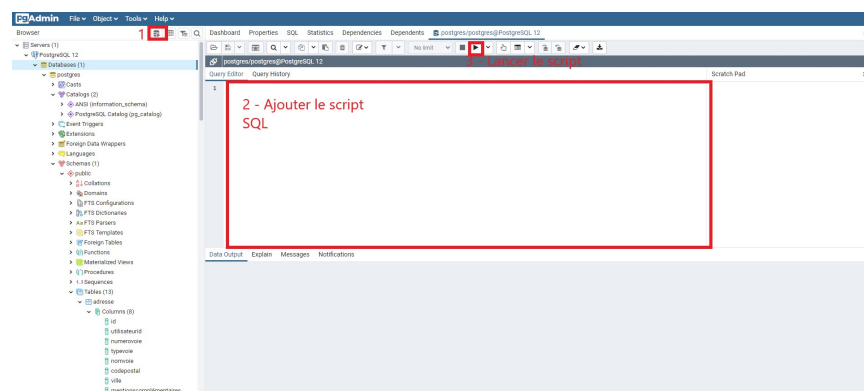
Pour la connexion à la base de données on peut suivre le tutoriel si l'on préfère pgAdmin sinon on peut suivre la procédure ci-dessous si l'on préfère SQL Workbench.

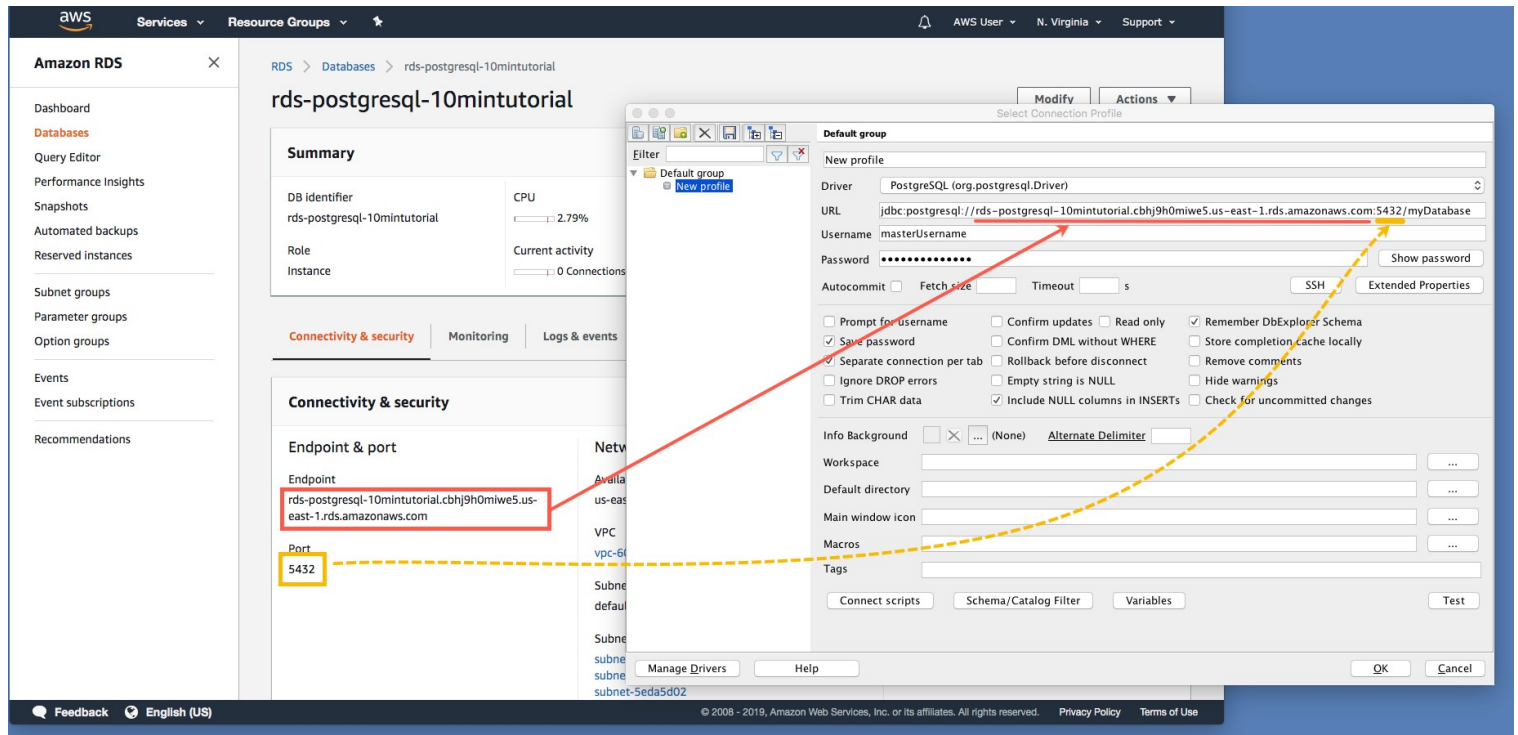
Pour se connecter à la base de données PostgreSQL, il faut dans un premier temps se connecter au compte AWS, naviguer vers « AWS Console Management » puis « Amazon RDS », ensuite « Database » et enfin choisir l'instance PostgreSQL.

Après avoir lancé le client SQL « SQL Workbench » il suffit de renseigner les champs requis du client SQL ainsi que l'URL à l'aide des informations données par Amazon RDS.

Pour créer les tables il faudra exécuter le script SQL disponible sur Github (voir PV de livraison).

Pour exécuter le script SQL via pgAdmin :





Capture d'écran explicative (source : site aws.amazon.com)

4.2 - Déploiement de l'application web

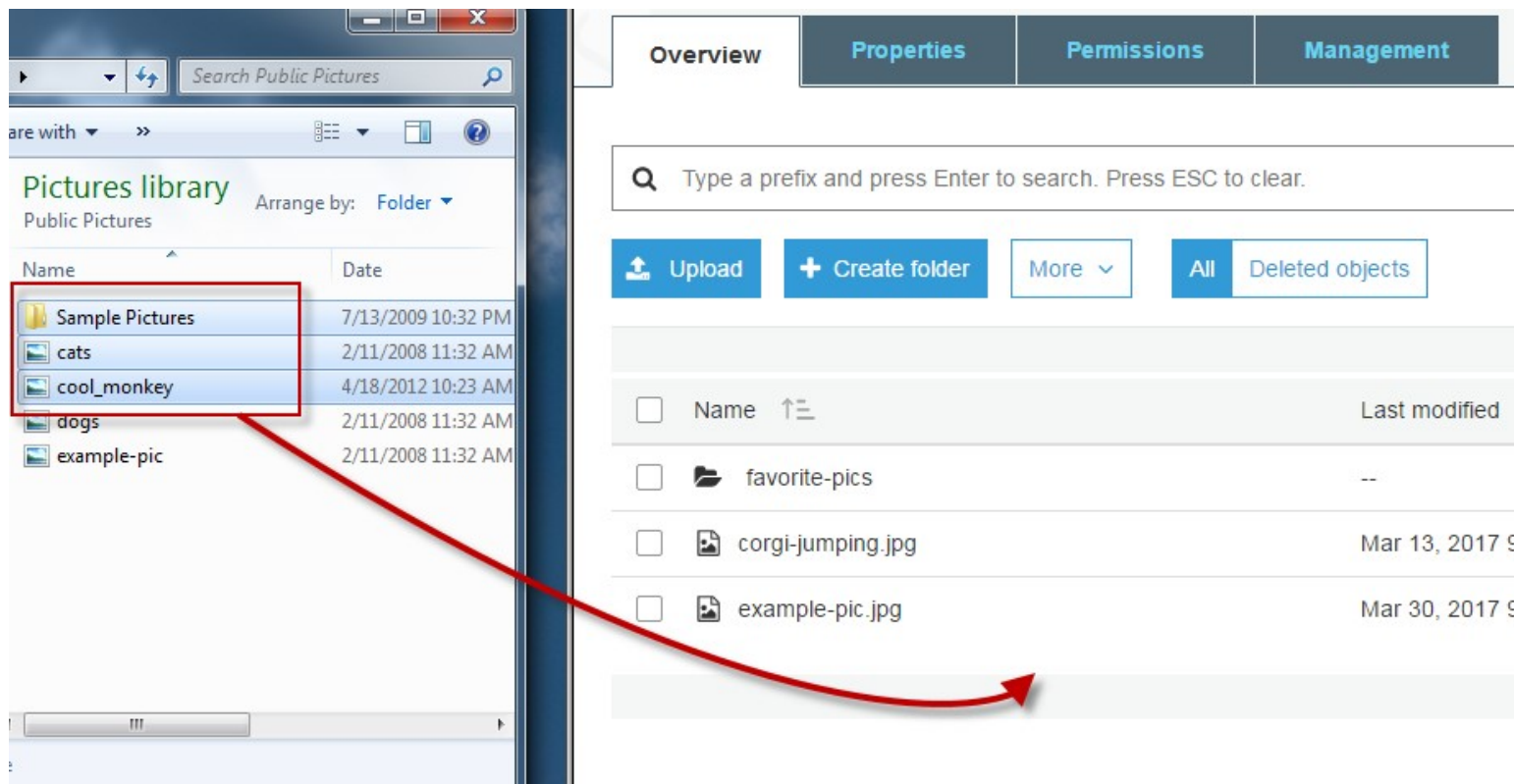
La documentation Amazon est très détaillée à ce sujet. Je vous invite à suivre ce lien pour créer le nom de domaine, ajouter le site dans l'instance Amazon S3 et le tester.

https://docs.aws.amazon.com/fr_fr/Route53/latest/DeveloperGuide/getting-started.html

A l'étape 4 du tutoriel, il faut remplacer le fichier html par l'application Angular compilée en .js.

Pour télécharger l'application dans le compartiment il faut :

1. Cliquer sur le service Amazon S3
2. Cliquer sur le compartiment créé à l'étape 2 du tutoriel d'Amazon
3. Cliquer sur l'onglet « Overview »



Capture d'écran d'un téléchargement de data dans un compartiment

(Source : aws.amazon.com)

Après avoir suivi le tutoriel d'Amazon sur la Route 53, il faut implémenter le web service Amazon CloudFront pour accélérer l'accès aux données du Amazon S3. Pour cela il faut créer une distribution CloudFront :

1. Se connecter au compte AWS
2. Sélectionner CloudFront dans le AWS Management Console
3. Sélectionner « Create distribution »
4. Sur la page « Select a delivery method for your content » dans la section « Web », choisissez « Get Started »
5. Sélectionner le compartiment S3 précédemment créé
6. Laisser les valeurs par défaut puis cliquer sur « Create distribution »

4.3 - Déploiement des API RESTful



Dans cette partie on va créer une API dans Amazon API Gateway puis lui assigner une fonction Lambda qui sera relié à Amazon RDS via un proxy que l'on va créer et qui contient des identifiants pour se connecter à la base de données de façon cryptée.

Dans un premier temps nous allons stocker les identifiants de la base de données dans AWS Secrets Manager :

1. Se connecter au compte AWS
2. Sélectionner AWS Secrets Manager dans le AWS Management Console
3. Sélectionner « Store a new Secret »
4. Remplir comme dans la capture ci-dessous en changeant l'utilisateur et le mot de passe
5. Sélectionner l'instance RDS
6. Cliquer sur Suivant

Store a new secret

Select secret type [Info](#)

☒ Credentials for RDS database

☐ Credentials for Redshift cluster

☐ Credentials for DocumentDB database

☐ Credentials for other database

☐ Other type of secrets (e.g. API key)

Specify the user name and password to be stored in this secret [Info](#)

User name

admin

Password

☐ Show password

Select the encryption key [Info](#)

Select the AWS KMS key to use to encrypt your secret information. You can encrypt using the default service encryption key that AWS Secrets Manager creates on your behalf or a customer master key (CMK) that you have stored in AWS KMS.

DefaultEncryptionKey



[Add new key](#)

Select which RDS database this secret will access [Info](#)

< 1 >

Dans un deuxième temps on va créer un proxy pour pouvoir utiliser la base de données stockée dans Amazon RDS via une API RESTful :

1. Se connecter sur le compte AWS
2. Sélectionner le service « Amazon RDS » sur le AWS Management Console
3. Sélectionner dans le panneau de navigation « Proxies » puis « Create Proxy »



Proxies (0)

Actions ▾

Create proxy

Filter proxies

< 1 > ⚙

Proxy identifier ▲	Status ▼	Engine compatibility ▼
<div>No proxies</div> <div>You don't have any proxies.</div> <div>Create proxy</div>		

4. Donner un nom au proxy

5. Choisir PostgreSQL

RDS > Proxies > Create proxy

Create proxy

Proxy configuration
Proxy makes applications more scalable, more transparent to database failures, and more secure.

Proxy identifier
Type a name for your proxy. The name must be unique across all proxies owned by your AWS account in the current AWS Region.

channy-proxy

Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

Engine compatibility [Info](#)

MySQL

MySQL

PostgreSQL

communications over networks.

Idle client connection timeout
Idle connection from your application will be closed after the specified time.

30

 minutes

00

 seconds
Max: 5 minutes

6. Choisir l'instance RDS créée au point 4.1.1 et assigner 100% dans le champ suivant

Target group configuration

A target group is a collection of databases that the proxy can connect to. Currently, you can specify only one RDS DB instance or Aurora DB cluster.

Database

Select the RDS DB instance or Aurora DB cluster that you want to associate with the proxy.

database-1 ▼

Connection pool maximum connections [Info](#)

Specify the maximum allowed connections, as a percentage of maximum connection limit of your database.

50

percentage

Specify the maximum allowed connections, as a percentage of maximum connection limit of your database. For example, if you have set the maximum connections to 5,000 connections, specifying 50% will allow the proxy to create up to 2,500 connections to the database.

► Additional target group configuration

7. Choisir le secret créé précédemment
8. Choisir le rôle IAM Admin ou en créer un nouveau (au choix)

Connectivity

Secrets Manager secret(s) [Info](#)

Create or choose Secrets Manager secret(s) representing the credentials for database user accounts that the proxy can use.

Choose one or more secret(s) ▼

prod/channy ✕

[Create a new secret](#) 

IAM role

Create or choose the IAM role the proxy will use to access the AWS Secrets Manager secret(s).

AWSServiceRoleForRDS ▼

IAM authentication

You can use IAM authentication to connect to the proxy in addition to specifying database credentials. Select how you want to use IAM authentication with the proxy. This choice applies to all secrets selected above.

Disabled ▼

9. Cliquer sur « Create Proxy »



Vous devez maintenant voir un proxy.

Dans un troisième temps on va créer les fonctions de l'API RESTful :

1. Se connecter au compte AWS
2. Sélectionner « AWS Lambda » dans l'AWS Console Management
3. Sélectionner « Create function » puis « Author from scratch »
4. Donner un nom à la fonction
5. Choisir « Java 8 » dans le menu « Runtime »
6. Cliquer sur « Create function »

La fonction est créée mais retourne un « Hello World ». On peut modifier le code dans l'éditeur de texte dans la sous-partie « Function Code » ou bien déployer une API développée sous le format ZIP ou Jar en sélectionnant « Upload » puis en parcourant et en choisissant l'API on peut cliquer sur « Save » pour déployer l'API.

Enfin on va créer l'API RESTful dans le service API Gateway et lui assigner la fonction Lambda créée :

1. Se connecter au compte AWS
2. Sélectionner « AWS API Gateway » dans l'AWS Console Management
3. Cliquer sur « Create API »
4. Donner un nom à l'API
5. Dans le menu déroulant « Actions », sélectionner « Create Resource »
6. Donner un nom à la ressource
7. Sélectionner la ressource puis cliquer sur le menu déroulant « Actions » puis « Create methods »
8. Choisir la méthode (GET, POST ...)
9. Choisir « Lambda Function » comme type d'intégration
10. Assigner la fonction et le proxy
11. Sélectionner dans le menu déroulant « Deploy API »

Nous avons maintenant accès à un URL nous permettant d'utiliser l'API.

4.4 - Déploiement de l'application Android

4.4.1 - Signer l'APK

Le prérequis pour suivre ce tutoriel est d'avoir installé Android Studio. Ce tutoriel explique comment créer une clé puis comment générer l'APK signée via la clé précédemment générée. La clé générée est à garder précieusement.

<https://developer.android.com/studio/publish/app-signing>

4.4.2 - Publier l'application dans le Play Store

Le prérequis est d'avoir un compte « Play Console » (payant).

Pour publier l'application, il faut :

1. Accéder à « Play Console » via ce lien <https://play.google.com/apps/publish/signup/>
2. Sélectionner « Toutes les applications » puis « Créer une application »
3. Remplir les champs

Pour plus d'information veuillez-vous référer au lien suivant :

https://support.google.com/googleplay/android-developer/answer/113469?hl=fr&ref_topic=7072031

5 - PROCEDURE DE DEMARRAGE / ARRET

5.1 - Base de données

Pour démarrer ou arrêter la base de données, veuillez suivre les étapes suivantes :

1. Se connecter au compte AWS
2. Naviguer vers le service web « Amazon RDS »
3. Sélectionner l'instance de la base de données PostgreSQL
4. Dans le menu « Actions », choisir « Stop » ou « Start »

5.2 - Application web / mobile

Pour démarrer ou arrêter l'application web, veuillez suivre les étapes suivantes :

1. Se connecter au compte AWS
2. Naviguer vers le service web « Amazon S3 »
3. Sélectionner l'instance du compartiment de l'application Angular
4. Dans le menu « Actions », choisir « Stop » ou « Start »

Cela entrainera l'arrêt du fonctionnement du front de l'application mobile.

5.3 - API RESTful

Il n'est pas possible d'arrêter une API sans la supprimer.

6 - PROCEDURE DE MISE A JOUR

6.1 - Base de données

Pour mettre à jour une base de données, il faut :

1. Se connecter au compte AWS
2. Naviguer vers le service web « Amazon RDS »
3. Sélectionner l'instance de la base de données PostgreSQL
4. Dérouler le menu « Instance Actions »
5. Cliquer sur « Modifier »
6. Sélectionner la version PostgreSQL souhaitée.

Pour les mises à jour mineures, il est possible d'automatiser la procédure en l'autorisant dans les propriétés de l'instance RDS.

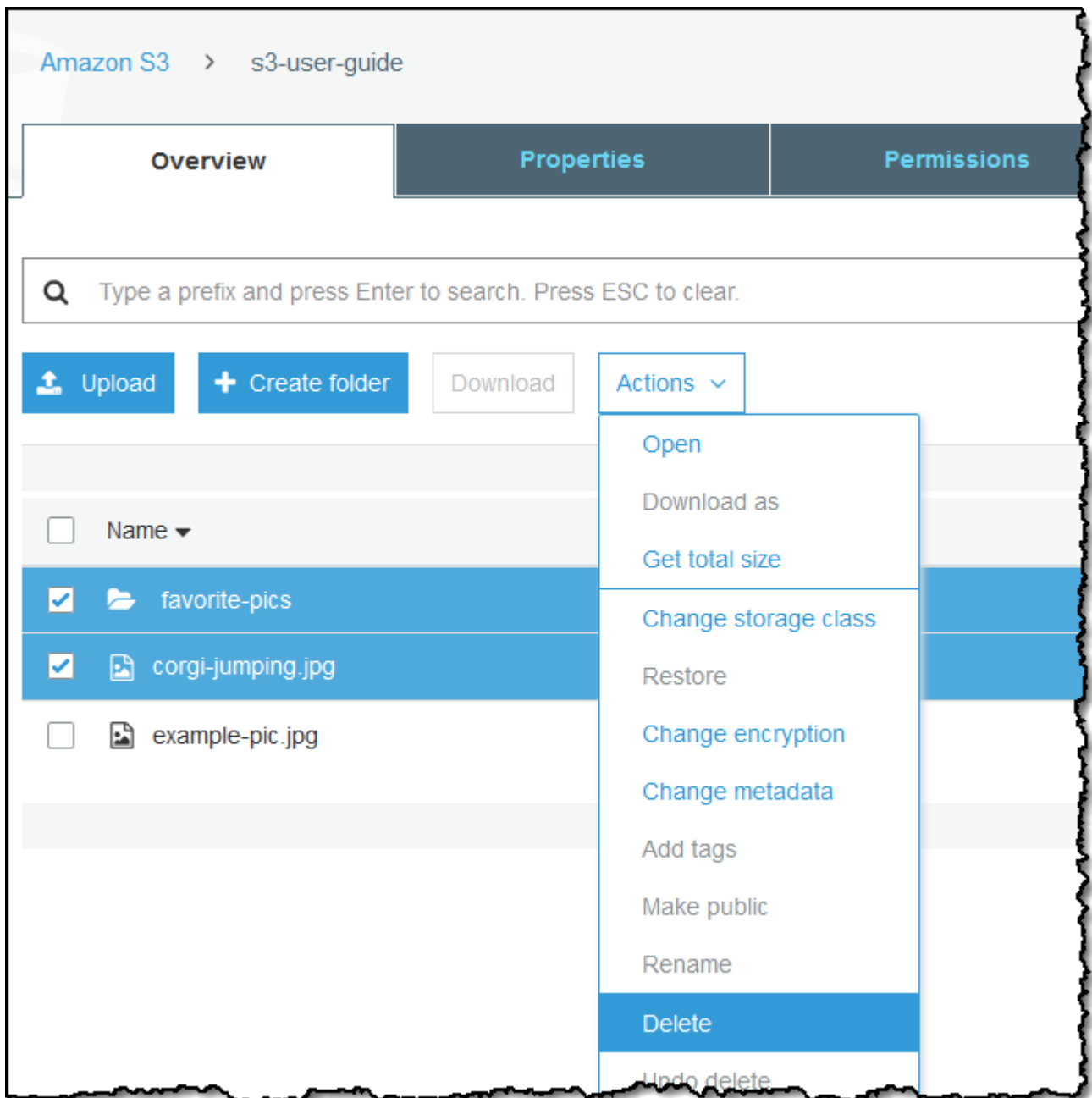
Le lien ci-dessous montre chaque étape de la procédure en anglais avec des captures d'écran.

<https://blog.toadworld.com/2017/03/31/in-place-upgrade-of-postgres-database-using-amazon-rds>

6.2 - Application web

Pour modifier l'application Angular il faut supprimer l'application du S3 puis le réajouter. Pour cela il est nécessaire de :

1. Se connecter au compte AWS
2. Naviguer vers le service web « Amazon S3 »
3. Sélectionner l'instance disponible
4. Cliquer sur l'onglet « Overview »
5. Sélectionner l'application Angular
6. Cliquer sur le menu déroulant « Actions » puis « Delete »



Capture d'écran expliquant comment supprimer des fichiers d'une instance S3

(Source : aws.amazon.com)

Pour l'ajout de l'application veuillez-vous référer au point 4.1.3).

6.3 - Application mobile

Pour mettre à jour une application il faut modifier l'application puis la signer avec la même clé qu'aux versions précédentes et incrémenter la version (disponible dans le fichier build.gradle).

Fichier build.gradle :

```
android {  
    ...  
    defaultConfig {  
        ...  
        versionCode 2  
        versionName "1.1"  
    }  
    productFlavors {  
        demo {  
            ...  
            versionName "1.1-demo"  
        }  
        full {  
            ...  
        }  
    }  
}
```

Maintenant on va mettre à jour l'application dans le Play Store. Pour cela :

1. Accéder à « Play Console » via ce lien <https://play.google.com/apps/publish/signup/>
2. Sélectionnez une application.
3. Dans le menu de gauche, sélectionnez Gestion des versions > Versions de l'application.
4. À côté de la release à déployer, sélectionnez Modifier la version.
5. Sélectionnez « Vérifier » pour accéder à l'écran "Vérifier et déployer la version", qui vous permet de vous assurer que votre version ne présente pas de problèmes avant de la déployer auprès des utilisateurs.
6. Passez en revue les avertissements et les erreurs.
7. Sélectionnez un pourcentage de déploiement.
8. Sélectionnez « Confirmer le déploiement »

Pour plus de détails cf.

<https://support.google.com/googleplay/android-developer/answer/113476?hl=fr>

6.4 - API RESTful

Pour modifier l'API RESTful il faut :

1. Se connecter au compte AWS
2. Naviguer vers le service web « Amazon Lambda »
3. Sélectionner la fonction à modifier
4. Cliquer sur « Upload »
5. Sélectionner la nouvelle API en ZIP
6. Cliquer sur « Save »

7 - SUPERVISION/MONITORING

AWS Cloud Watch est un service web qui fournit des données et informations exploitables, les variations de performances sur l'ensemble des services d'Amazon sous forme de journaux, de métriques et d'évènement. Il permet ainsi une supervision centralisée de l'application. Il n'y a pas de configuration de base à effectuer. Pour accéder à AWS Cloud Watch il est nécessaire de :

1. Se connecter au compte AWS
2. Naviguer au AWS Management Console
3. Sélectionner AWS Cloud Watch

Cette vidéo provient de la chaine officielle d'Amazon Web Service et montre son utilisation.

<https://www.youtube.com/watch?v=I7EFLChc07M>

8 - PROCEDURE DE SAUVEGARDE ET RESTAURATION

AWS Backup permet une sauvegarde et une restauration des données de l'application en toute simplicité via une interface graphique et de façon sécurisée grâce à son chiffrement.

Pour cela il faut créer un backup plan :

1. Se connecter au compte AWS
2. Naviguer au AWS Management Console
3. Sélectionner AWS Backup
4. Sélectionner « Create Backup plan »
5. Remplir les champs selon les besoins (fréquence de sauvegarde...)
6. Sélectionner « Create plan » pour créer le plan

Puis assigner des ressources à sauvegarder :

1. Sélectionner « Assign resources »
2. Entrer le nom de l'instance web à sauvegarder dans le champ « Resource assignment name »
3. Cocher la case « Default role »
4. Choisir dans le menu déroulant « Assign by » le champ « Resource ID »
5. Choisir le service web à sauvegarder dans le menu déroulant « Resource type »
6. Entrer le nom de l'instance du web service à sauvegarder
7. Sélectionner « Add assignment »
8. Sélectionner « Add resources »

Les backups seront sauvegardés dans l'onglet « Protected resources ».

Pour restaurer une sauvegarde il faut :

1. Sélectionner la sauvegarde
2. Sélectionner « Restore »



3. Sélectionner « Restore backups »

Dans l'onglet « Jobs » puis « Restore jobs » on peut voir l'état de la restauration.

Pour plus d'information veuillez consulter la vidéo ci-dessous :

<https://youtu.be/AI8hKeh7yCs?t=1014>

9 - GLOSSAIRE

SGBD	Système de gestion de base de données
AWS	Amazon Web Service
Amazon S3	Amazon Simple Stockage Service
Amazon RDS	Amazon Relational Database Service
Amazon IAM	Amazon Identity and Access Management
MPD	Modèle Physique des Données
SSH	Secure Shell
APK	Android Package Kit
DNS	Domain Name System