

Create and deploy SaaS applications using Azure App Service

Who am I

Site Reliability Engineer (SRE) @ **Warpnet**

Security Specialist @ **CJIB**

Prior Azure experience limited to AZ-900

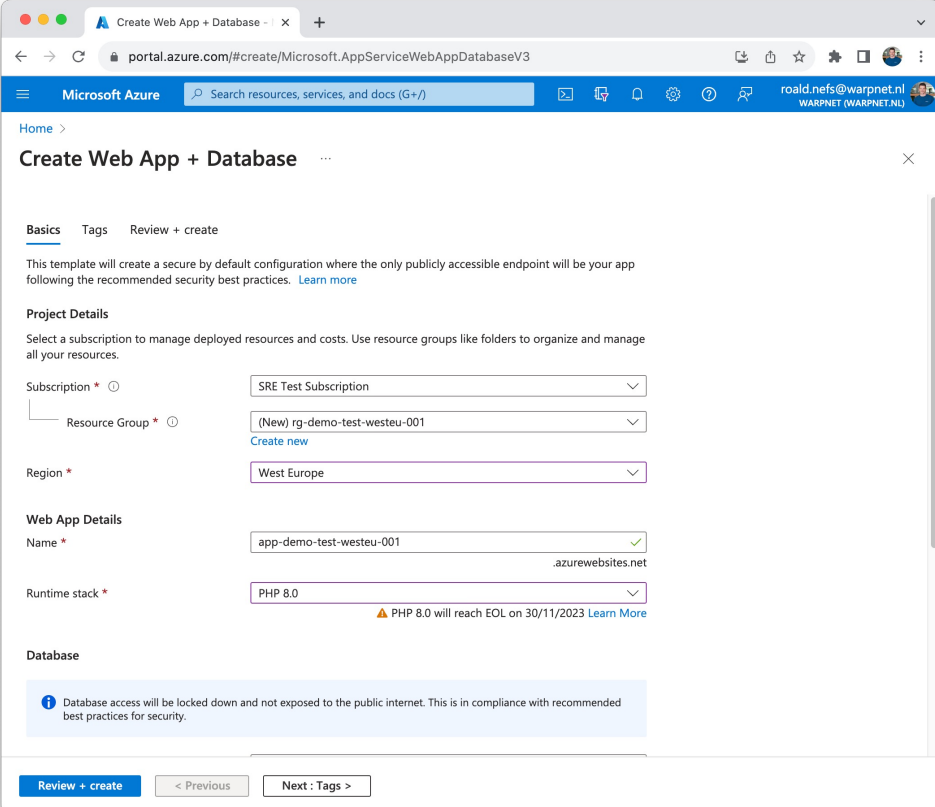


**Let's deploy a Laravel
application**

Create Web App + Database

Create **Web App + Database** on the PHP 8.0 runtime stack. This will result in the following resources:

- Resource group
- App Service plan
- App Service
- Virtual network
- Azure Database for MySQL – Flexible Server
- Private DNS zone




The screenshot shows the 'Create Web App + Database' wizard in the Microsoft Azure portal. The page is titled 'Create Web App + Database' and has tabs for 'Basics', 'Tags', and 'Review + create'. The 'Basics' tab is active. The page contains the following sections and fields:

- Project Details:** A section with the instruction: 'Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.' It includes a 'Subscription' dropdown menu set to 'SRE Test Subscription' and a 'Resource Group' dropdown menu set to '(New) rg-demo-test-westeu-001'. There is a 'Create new' link below the Resource Group dropdown.
- Region:** A dropdown menu set to 'West Europe'.
- Web App Details:** A 'Name' field set to 'app-demo-test-westeu-001' with a checkmark icon and '.azurewebsites.net' as the domain. Below it is a 'Runtime stack' dropdown menu set to 'PHP 8.0'. A warning icon and text state: 'PHP 8.0 will reach EOL on 30/11/2023 Learn More'.
- Database:** A section with an information icon and text: 'Database access will be locked down and not exposed to the public internet. This is in compliance with recommended best practices for security.'

At the bottom of the wizard, there are three buttons: 'Review + create' (highlighted in blue), '< Previous', and 'Next: Tags >'.


app-demo-test-westeu-001 - x Microsoft Azure App Service - x +


app-demo-test-westeu-001.azurewebsites.net



Your web app is running and waiting for your content

Your web app is live, but we don't have your content yet. If you've already deployed, it could take up to 5 minutes for your content to show up, so come back soon.



 Supporting Node.js, Java, .NET and more

<p>Haven't deployed yet? Use the deployment center to publish code or set up continuous deployment.</p> <p>Deployment center</p>	<p>Starting a new web site? Follow our Quickstart guide to get a web app ready quickly.</p> <p>Quickstart</p>
--	---

app-demo-test-westeu-001 - x app-demo-test-westeu-001 - x ssh://169.254.129.2 x +

app-demo-test-westeu-001.scm.azurewebsites.net/webssh/host

```

  _____
 /         \
|   _   _   |
|  (o)  (o)  |
|   _   _   |
 \_____/

APP SERVICE ON LINUX

Documentation: http://aka.ms/webapp-linux
PHP quickstart: https://aka.ms/php-gs
PHP version : 8.0.30
Note: Any data outside '/home' is not persisted
root@1a04599e6977:/home# ls
ASP.NET  DeploymentLogStream  LogFiles  site  u89be66152c5a0e096799fc
root@1a04599e6977:/home# cd site/
root@1a04599e6977:/home/site# ls
deployments  locks  repository  wwwroot
root@1a04599e6977:/home/site# cd wwwroot/
root@1a04599e6977:/home/site/wwwroot# ls
hostingstart.html
root@1a04599e6977:/home/site/wwwroot#

```

Menu | ssh://root@169.254.129.2:2222 | SSH CONNECTION ESTABLISHED

Database connectivity

Default Azure application settings do not match with Laravel, e.g.

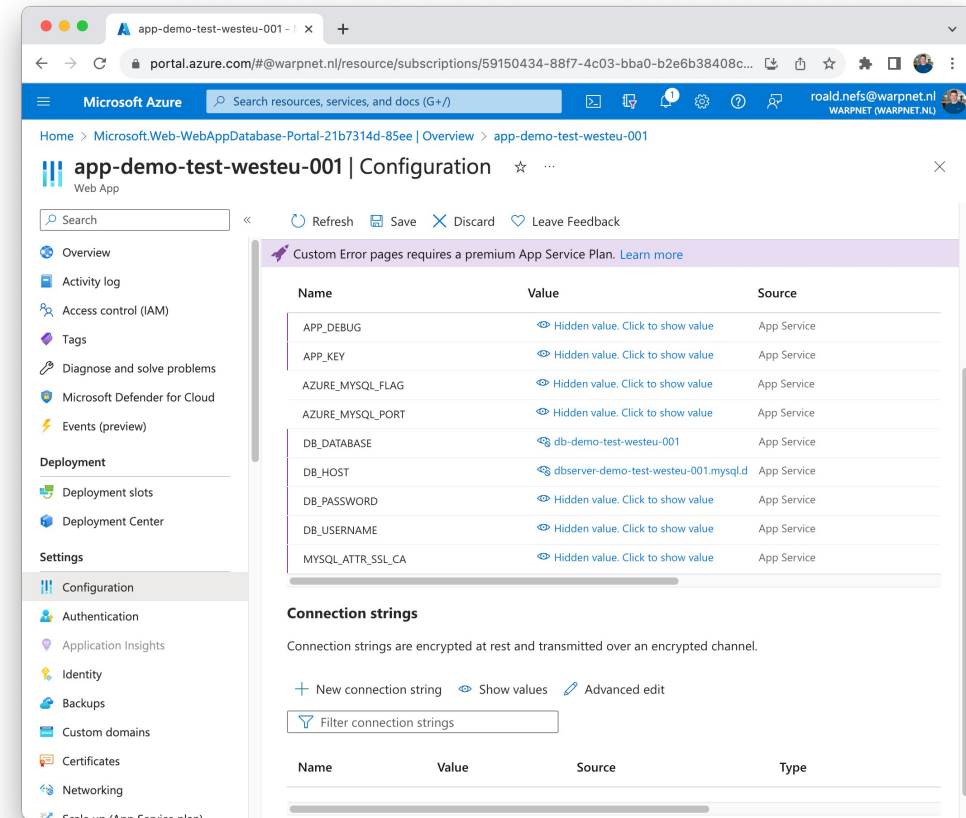
AZURE_MYSQL_DBNAME → DB_DATABASE

Add additional environment variables for Laravel:

APP_DEBUG

APP_KEY

MYSQL_ATTR_SSL_CA



The screenshot shows the Azure portal configuration page for an App Service. The 'Configuration' tab is selected, displaying a table of environment variables. The table has columns for Name, Value, and Source. The variables listed are:

Name	Value	Source
APP_DEBUG	Hidden value. Click to show value	App Service
APP_KEY	Hidden value. Click to show value	App Service
AZURE_MYSQL_FLAG	Hidden value. Click to show value	App Service
AZURE_MYSQL_PORT	Hidden value. Click to show value	App Service
DB_DATABASE	db-demo-test-westeu-001	App Service
DB_HOST	observer-demo-test-westeu-001.mysql.d	App Service
DB_PASSWORD	Hidden value. Click to show value	App Service
DB_USERNAME	Hidden value. Click to show value	App Service
MYSQL_ATTR_SSL_CA	Hidden value. Click to show value	App Service

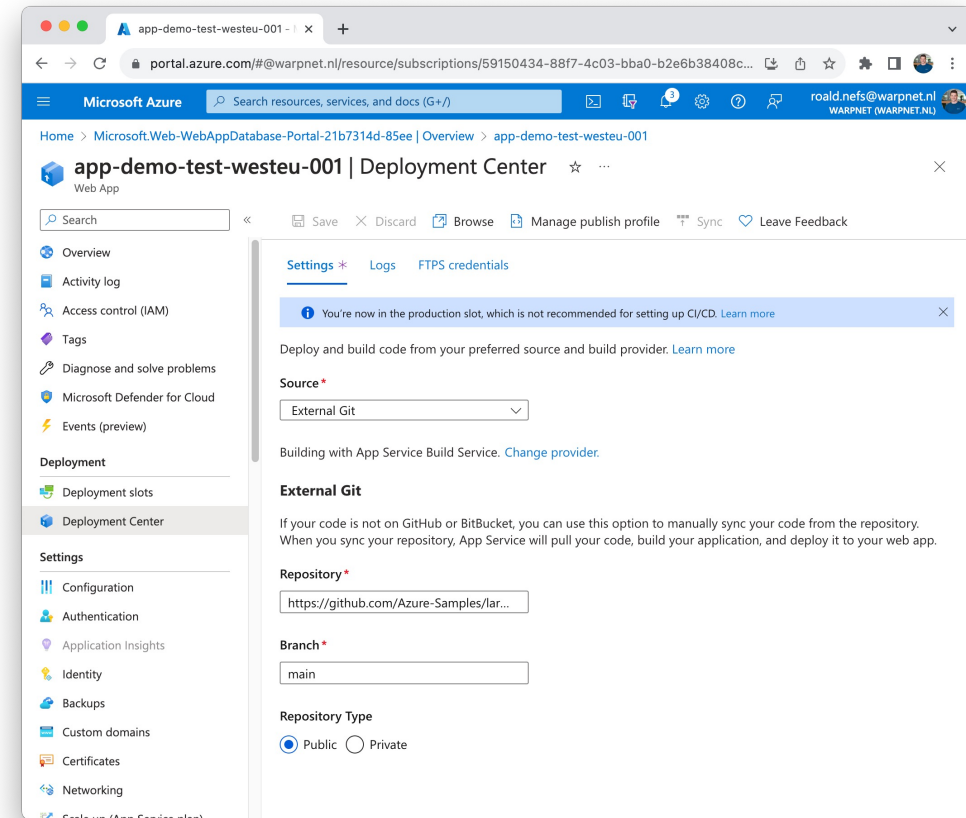
Below the table, there is a section for 'Connection strings' with a filter and a table with columns for Name, Value, Source, and Type.

Deploy code

Let's deploy the following Laravel project from an external repository:

```
github.com/Azure-Samples/laravel-tasks
```

Takes a couple of minutes and ran into failing deployments 5/5 times...

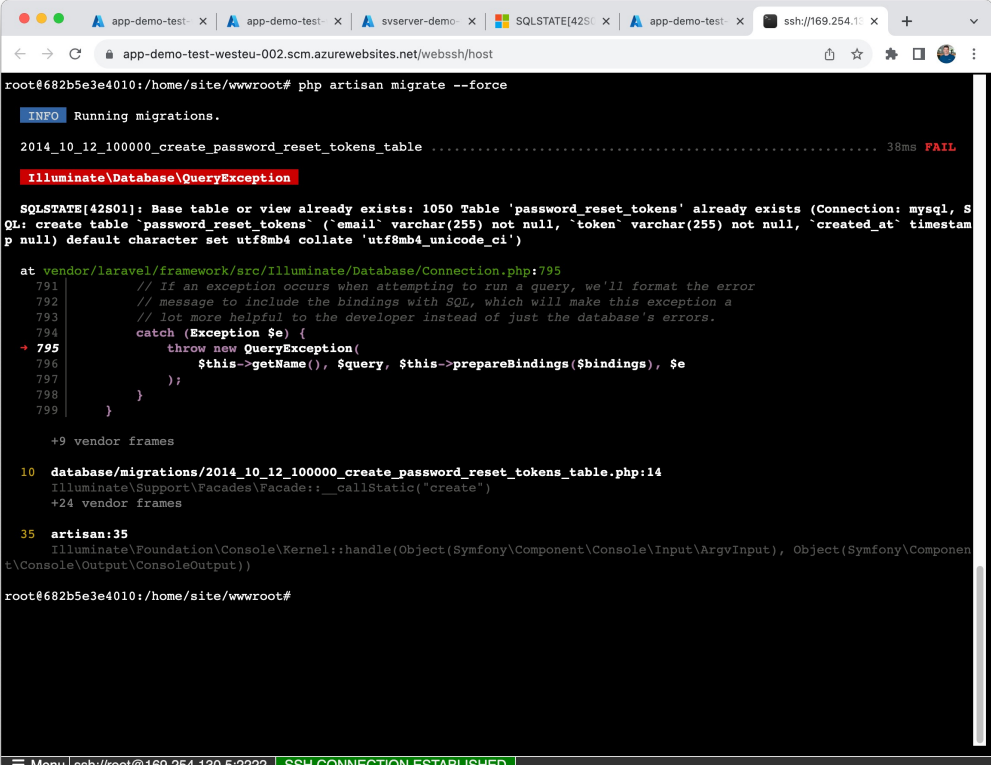


Let's try again

using PHP 8.2

Generate database schema

Login to the container using SSH to run the required database migrations.



```
root@682b5e3e4010:/home/site/wwwroot# php artisan migrate --force

INFO Running migrations.

2014_10_12_100000_create_password_reset_tokens_table ..... 38ms FAIL

Illuminate\Database\QueryException

SQLSTATE[42S01]: Base table or view already exists: 1050 Table 'password_reset_tokens' already exists (Connection: mysql, SQL: create table `password_reset_tokens` (`email` varchar(255) not null, `token` varchar(255) not null, `created_at` timestamp null) default character set utf8mb4 collate 'utf8mb4_unicode_ci')

at vendor/laravel/framework/src/Illuminate/Database/Connection.php:795
791 // If an exception occurs when attempting to run a query, we'll format the error
792 // message to include the bindings with SQL, which will make this exception a
793 // lot more helpful to the developer instead of just the database's errors.
794 catch (Exception $e) {
+ 795     throw new QueryException(
796         $this->getName(), $query, $this->prepareBindings($bindings), $e
797     );
798 }
799 }

+9 vendor frames

10 database/migrations/2014_10_12_100000_create_password_reset_tokens_table.php:14
Illuminate\Support\Facades\Facade::__callStatic("create")
+24 vendor frames

35 artisan:35
Illuminate\Foundation\Console\Kernel::handle(Object(Symfony\Component\Console\Input\ArgvInput), Object(Symfony\Component\Console\Output\ConsoleOutput))

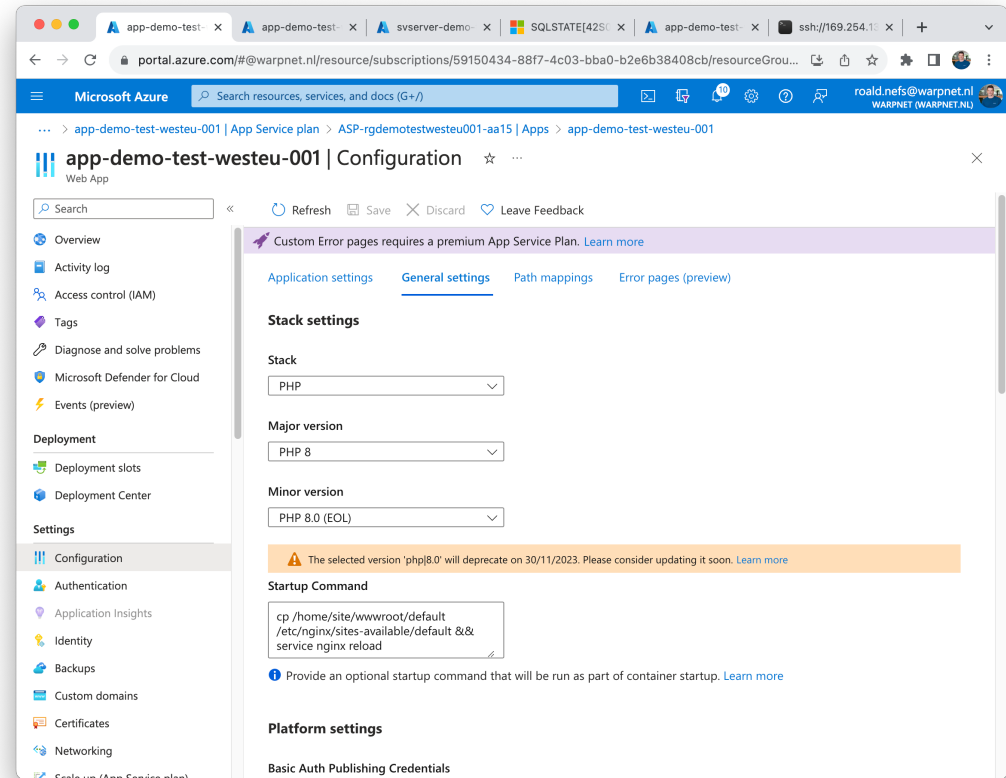
root@682b5e3e4010:/home/site/wwwroot#
```

Menu | ssh://root@169.254.130.5:2222 | SSH CONNECTION ESTABLISHED

Change the site root

The Laravel application lifecycle begins in the `./public` directory. The example project includes a new Nginx configuration.

Simply update the startup command.



Browse the app...

A screenshot of a web browser window displaying a database error. The browser address bar shows 'app-demo-test-westeu-002.azurewebsites.net'. The error message is 'Illuminate\Database\QueryException' with the details 'SQLSTATE[42S02]: Base table or view not found: 1146 Table 'laravel.tasks' doesn't exist'. Below the error is the SQL query: 'SELECT * FROM `tasks` ORDER BY `created_at` ASC'. A green notification box on the right says 'A table was not found' and provides instructions to run database migrations using 'php artisan migrate'. Below the error is a code editor showing PHP code from 'routes/web.php:30'. The code includes 'use App\Models\Task;' and 'use Illuminate\Http\Request;'. A route is defined for the root path: 'Route::get('/', function () { ... } else { \$data = Task::orderBy('created_at', 'asc')->get(); }'. The code editor also shows a list of vendor frames on the left, including 'Illuminate\Routing\RouteFileRegistrar:30 (closure)' and 'public/index.php:51 [top]'.

```
Illuminate\Database\QueryException          PHP 8.2.8  10.13.2

SQLSTATE[42S02]: Base table or view not found: 1146 Table
'laravel.tasks' doesn't exist

SELECT * FROM `tasks` ORDER BY `created_at` ASC

A table was not found
You might have forgotten to run your database
migrations.
You can try to run your migrations using `php
artisan migrate`.
Database: Running Migrations docs

Expand vendor frames

9 vendor frames
Illuminate\Routing\RouteFileRegistrar:30
(closure)
42 vendor frames
public/index.php:51
[top]

routes/web.php:30
15
16 use App\Models\Task;
17 use Illuminate\Http\Request;
18
19 /**
20  * Show Task Dashboard
21  */
22 Route::get('/', function () {
23     Log::info("Get /");
24     $startTime = microtime(true);
25     // Simple cache-aside logic
26     if (Cache::has('tasks')) {
27         $data = Cache::get('tasks');
28         return view('tasks', ['tasks' => $data, 'elapsed' => microtime(true) - $s
29     } else {
30         $data = Task::orderBy('created_at', 'asc')->get();
```

Let's deploy Keycloak

using the CLI

Create the resource group

Start with creating a resource group to store all required resources.

```
$ az group create \  
  --location westeurope \  
  --name rg-keycloak-test-westeu-001
```

Create the App Service Plan

Let's create a App Service Plan based upon Linux and a cheap plan.

```
$ az appservice plan create \  
  --resource-group rg-keycloak-test-westeu-001 \  
  --name asp-keycloak-test-westeu-001 \  
  --is-linux \  
  --location westeurope \  
  --sku B1
```

Create the Web App (container)

Create the Web App with the official container.

```
$ az webapp create \  
  --name app-keycloak-test-westeu-001 \  
  --plan asp-keycloak-test-westeu-001 \  
  --resource-group rg-keycloak-test-westeu-001 \  
  --deployment-container-image-name \  
  quay.io/keycloak/keycloak:22.0.1
```



Let's try again

By updating container start time limit

Let's update the default

`WEBSITES_CONTAINER_START_TIME_LIMIT`
from **230** to **1800** (max) to be able to
start the Keycloak container.



Create a PostgreSQL - Flexible Server

Let's first create a PostgreSQL server to store the Keycloak data.

```
$ az postgres flexible-server create \  
--resource-group rg-keycloak-test-westeu-001 \  
--name psql-keycloak-test-westeu-001 \  
--location westeurope \  
--sku-name Standard_B1ms --tier Burstable \  
--version 15 \  
--public-access None \  
--admin-user tigalahe20 \  
--admin-password REDACTED
```

Create a PostgreSQL database

Create a new PostgreSQL database for keycloak to use.

```
$ az postgres flexible-server db create \  
  --resource-group rg-keycloak-test-westeu-001 \  
  --server-name psql-keycloak-test-westeu-001 \  
  --database-name keycloak
```

Update Keycloak settings

Update Keycloak settings by setting the required 12 application settings, e.g.:

KEYCLOAK_FRONTEND_URL

KEYCLOAK_ADMIN

KEYCLOAK_ADMIN_PASSWORD

KEYCLOAK_FRONTEND_URL

PROXY_ADDRESS_FORWARDING

DB_VENDOR

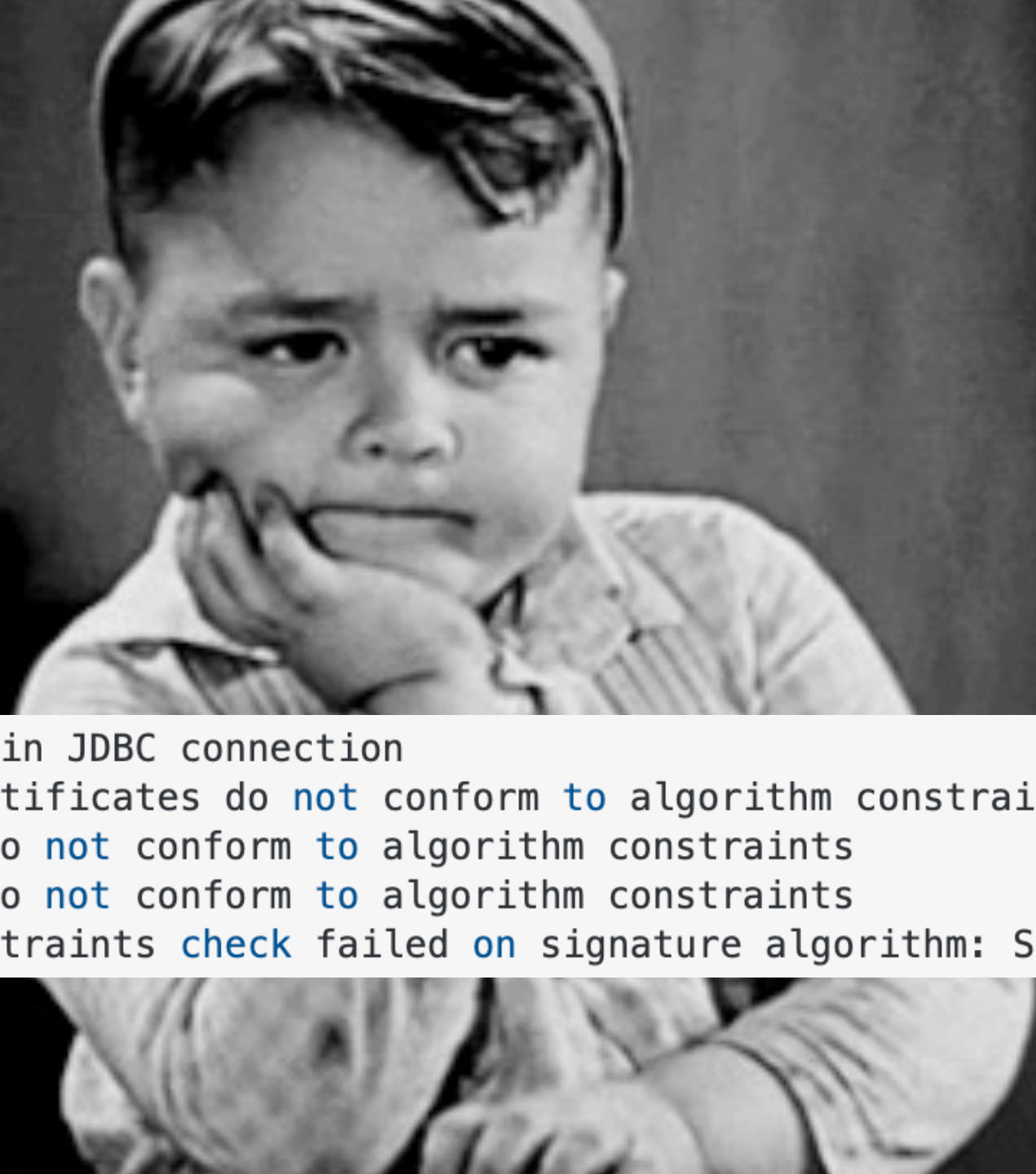
DB_ADDR

DB_USER

KC_PROXY

KC_HOSTNAME

```
$ az webapp config appsettings \  
  set --resource-group rg-keycloak-test-westeu-001 \  
  --name app-keycloak-test-westeu-001 \  
  --settings @settings.json
```



```
ERROR: Failed to obtain JDBC connection
ERROR: SSL error: Certificates do not conform to algorithm constraints
ERROR: Certificates do not conform to algorithm constraints
ERROR: Certificates do not conform to algorithm constraints
ERROR: Algorithm constraints check failed on signature algorithm: SHA1withRSA
```

According to a random person on the internet



ThomasAunvik commented on Mar 9



Same issue for me regarding the certificate on Azure Postgresql.

It would seem that the Azure Database for Postgresql - **Single Server**, is using the SHA256 DigiCertGlobalRootG2, for anyone using Single Server would not have this problem.

But for Flexible server, is still using the old SHA1 Root Certificate, causing this error.



Let's try again

using a custom Dockerfile

How to continue

Create a new Dockerfile that can be used to create a container including the needed certificate to allow the database connection.

```
FROM quay.io/keycloak/keycloak:22.0.1 as builder

# Enable health and metrics support
ENV KC_HEALTH_ENABLED=true
ENV KC_METRICS_ENABLED=true

# Configure PostgreSQL as the database vendor
ENV KC_DB=postgres

WORKDIR /opt/keycloak
RUN /opt/keycloak/bin/kc.sh build

FROM quay.io/keycloak/keycloak:latest
COPY --from=builder /opt/keycloak/ /opt/keycloak/
COPY DigiCertGlobalRootCA.crt.pem
/opt/keycloak/.postgresql/root.crt

ENV KC_DB=postgres
ENTRYPOINT ["/opt/keycloak/bin/kc.sh"]
```

Use custom Docker image

1. Upload the image to an existing Docker registry.
2. Update the container in the **Web App** configuration.

1. Create an Azure container registry.
2. Login to the new registry.
3. Push the custom Docker image.
4. Update the container in the **Web App** configuration.



master

Manage

Clients

Client scopes

Realm roles

Users

Groups

Sessions

master

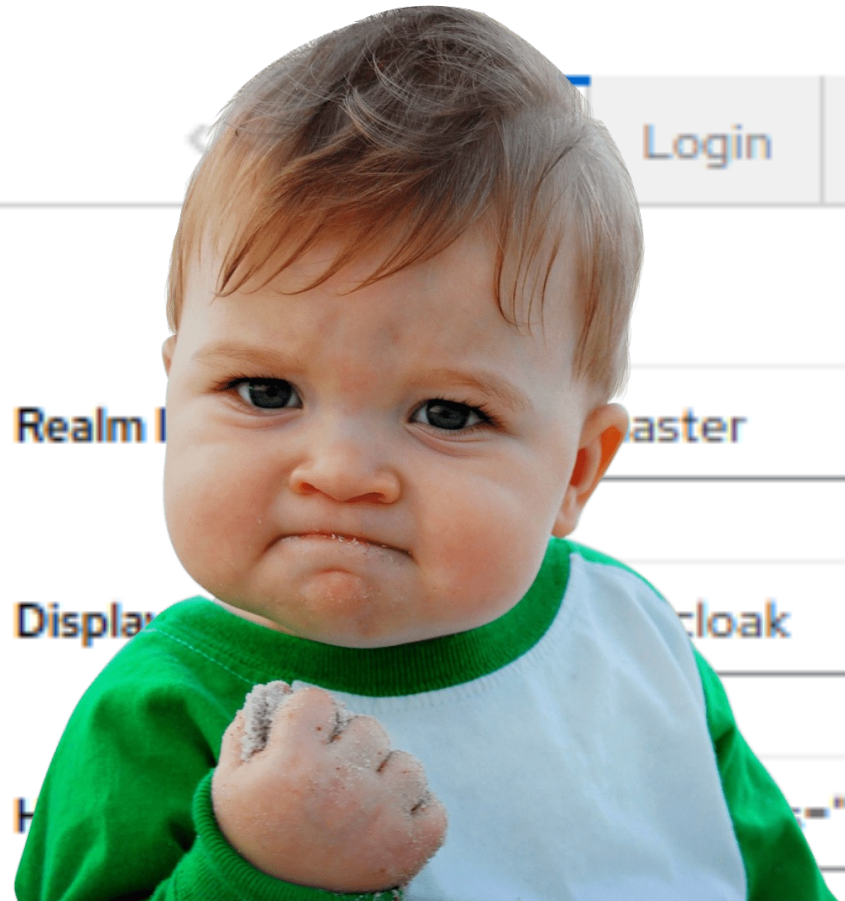
Realm settings are settings that control the options for users, application

<	Login	Email	Themes	Keys	Events
---	-------	-------	--------	------	--------

Realm ID: master

Display name: Keycloak

HTML: Keycloak



Let's recap