Upgrading to v2 from v1

> Sometimes the upgrade can go wrong, so we recommend that you back up your database before starting the upgrade.

Docker

Upgrading to Mixpost v2 with Docker is straightforward.

1. Navigate to your folder where you have the `docker-compose.yml`.
2. Make sure the image for `mixpost` service is `inovector/mixpost-pro-team:latest`
3. Next:

```
# Pull the latest version
docker-compose pull

# Stop and remove the old container
docker-compose down

# Start a new container
docker-compose up -d
```

> If something goes wrong, you can use the **inovector/mixpost-pro-team:v1 tag** to revert. Also, you need to restore your database backup.

In your Standalone or Laravel app

Updating your composer.json using the "require" command

```
composer require inovector/mixpost-pro-team "^2.0"
```

Updating the database

Some changes were made to the database, use the migration below to update your database to the latest schema:

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
use Inovector\Mixpost\Models\Service;

return new class extends Migration {
    public function up()
    {
        if (Schema::hasTable('mixpost_services')) {
            if (Schema::hasColumn('mixpost_services', 'credentials') &&
!Schema::hasColumn('mixpost_services', 'configuration')) {
                Schema::table('mixpost_services', function (Blueprint $table) {
                    $table->renameColumn('credentials', 'configuration');
                });
            }


            if (!Schema::hasColumn('mixpost_services', 'active')) {
                Schema::table('mixpost_services', function (Blueprint $table) {
                    $table->boolean('active')->default(false);
                });
            }

            Service::query()->update(['active' => true]);
        }

        if (!Schema::hasTable('mixpost_user_tokens')) {
            Schema::create('mixpost_user_tokens', function (Blueprint $table) {
                $table->id();
                $table->bigInteger('user_id')->unsigned()->index();
                $table->foreign('user_id')->references('id')->on('users')-
>onDelete('cascade');
                $table->string('name');
                $table->string('token', 64)->unique();
                $table->timestamp('last_used_at')->nullable();
                $table->date('expires_at')->nullable();
                $table->timestamps();
            });
```

```
        }

        if (!Schema::hasTable('mixpost_webhooks')) {
            Schema::create('mixpost_webhooks', function (Blueprint $table) {
                $table->id();
                $table->uuid('uuid')->unique();
                $table->bigInteger('workspace_id')->nullable()->index(); // if null, it's a
system webhook
                $table->string('name');
                $table->string('callback_url');
                $table->string('method')->default('post');
                $table->string('content_type');
                $table->tinyInteger('max_attempts')->default(1);
                $table->tinyInteger('last_delivery_status')->nullable();
                $table->text('secret')->nullable();
                $table->boolean('active')->default(false);
                $table->json('events')->nullable();
                $table->timestamps();

                $table->index(['workspace_id', 'active']);
            });
        }

        if (!Schema::hasTable('mixpost_webhook_deliveries')) {
            Schema::create('mixpost_webhook_deliveries', function (Blueprint $table) {
                $table->id();
                $table->uuid('uuid')->unique();
                $table->bigInteger('webhook_id')->unsigned()->index();
                $table->foreign('webhook_id')->references('id')->on('mixpost_webhooks')-
>onDelete('cascade');
                $table->string('event');
                $table->tinyInteger('attempts')->default(0);
                $table->tinyInteger('status');
                $table->integer('http_status')->nullable();
                $table->timestamp('resend_at')->nullable();
                $table->boolean('resent_manually')->default(false);
                $table->json('payload')->nullable();
                $table->json('response')->nullable();
                $table->timestamp('created_at')->nullable();
            });
```

```
        }

        Schema::dropIfExists('mixpost_post_version_media');
    }
};
```

Don't know how to migrate?

1. Navigate to your Standalone/Laravel app.
2. Run ***php artisan make:migration create_mixpost_v2_tables***
3. Open the migration file from ***database/migrations*** and copy the migration code from above into your migration file
4. Run ***php artisan migrate***
5. Done!

Updating the config file

There have been numerous config changes. If you have published the Mixpost configuration file, you should update it manually.

The easiest way to update this is to run:

```
php artisan vendor:publish --tag=mixpost-config --force
```

Clear the cache

Clear the cache by running the following command:

```
php artisan route:cache
php artisan mixpost:clear-settings-cache
php artisan mixpost:clear-services-cache
```

Finally, terminate the Horizon process by running:

```
php artisan horizon:terminate
```

Revision #4
Created 1 April 2024 10:45:04 by Dima Botezatu
Updated 2 April 2024 06:42:41 by Dima Botezatu