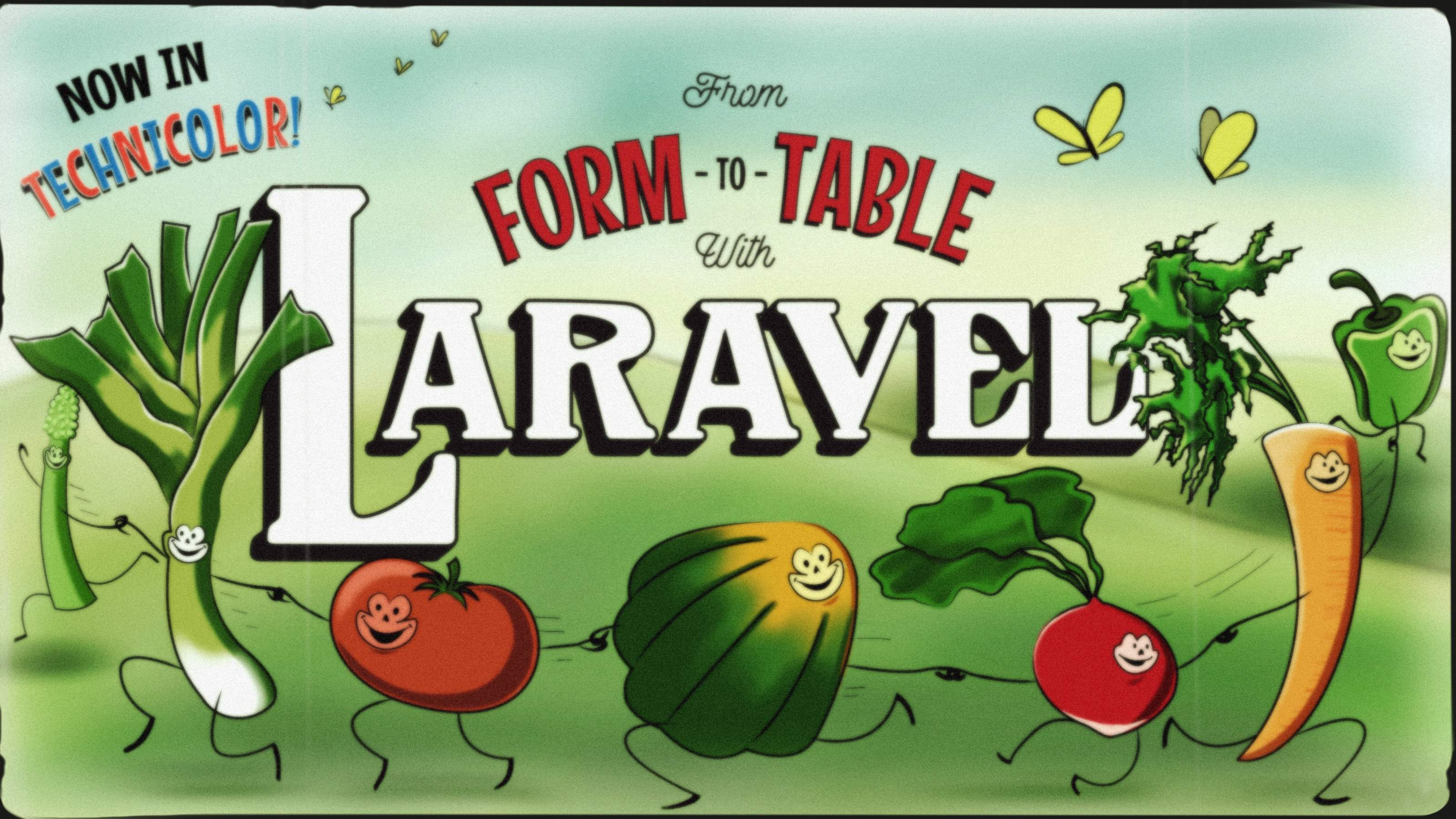
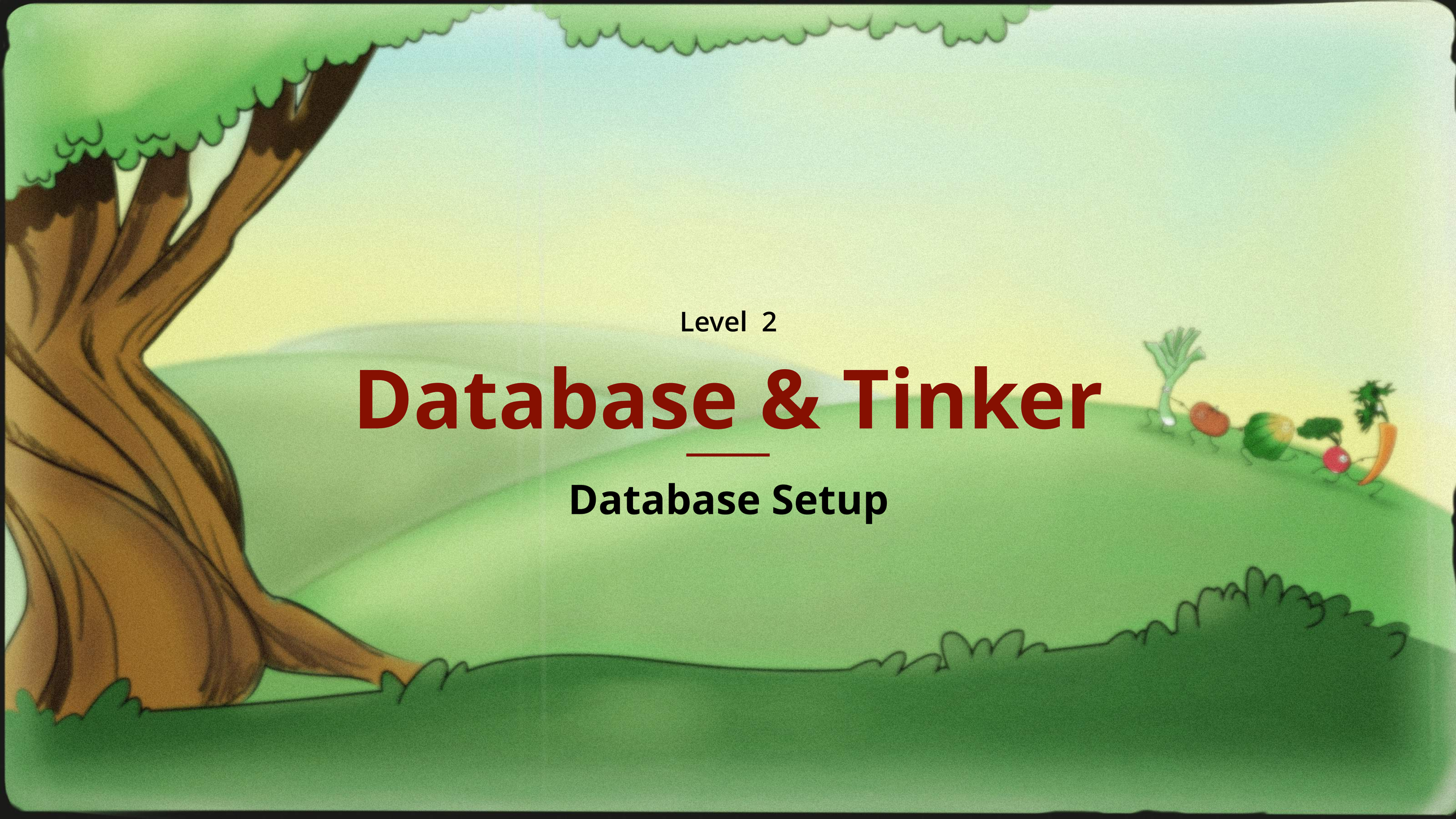

NOW IN
TECHNICOLOR!
From
FORM - TO - TABLE
With
LARAVEL



Level 2

Database & Tinker

Database Setup

Setting up the Database

What are the steps to getting data into our database with the Market model?

Configure our database

Create our database

Run our migration

Creating the Database

Using a locally installed copy of mysql, we can create the database from the CLI.

terminal.app

```
~/market $ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
…

mysql> create database market;
mysql> exit
Bye
```

start the `mysql` CLI with root user and password protection

the CLI allows direct queries

send the `exit` command to quit

go.codeschool.com/mysqlinstall

Configure Laravel's DB Connection

With an .env file, we can have settings for local or remote environments.

.env

```
…

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=market
DB_USERNAME=root
DB_PASSWORD=password
…
```

our local machine IP or loopback

the database name we want to use for the application

Run Migrations!

When we run our migration, it will create our database table and columns.

terminal.app

```
~/market $  php artisan migrate

Migration table created successfully.
Migrated: 2017_02_24_183842_create_markets_table

~/market $
```

run all our migrations

the migration status is stored in another DB table

Checking for Tables

Running a tables query on our database will verify that the migration was successful.

terminal.app

```
mysql> use market;
Database changed

mysql> show tables;
+------------------+
| Tables_in_market |
+------------------+
| markets          |
| migrations       |
| password_resets  |
| users            |
+------------------+
```

now the CLI will query the market database

all these tables are controlled by migrations

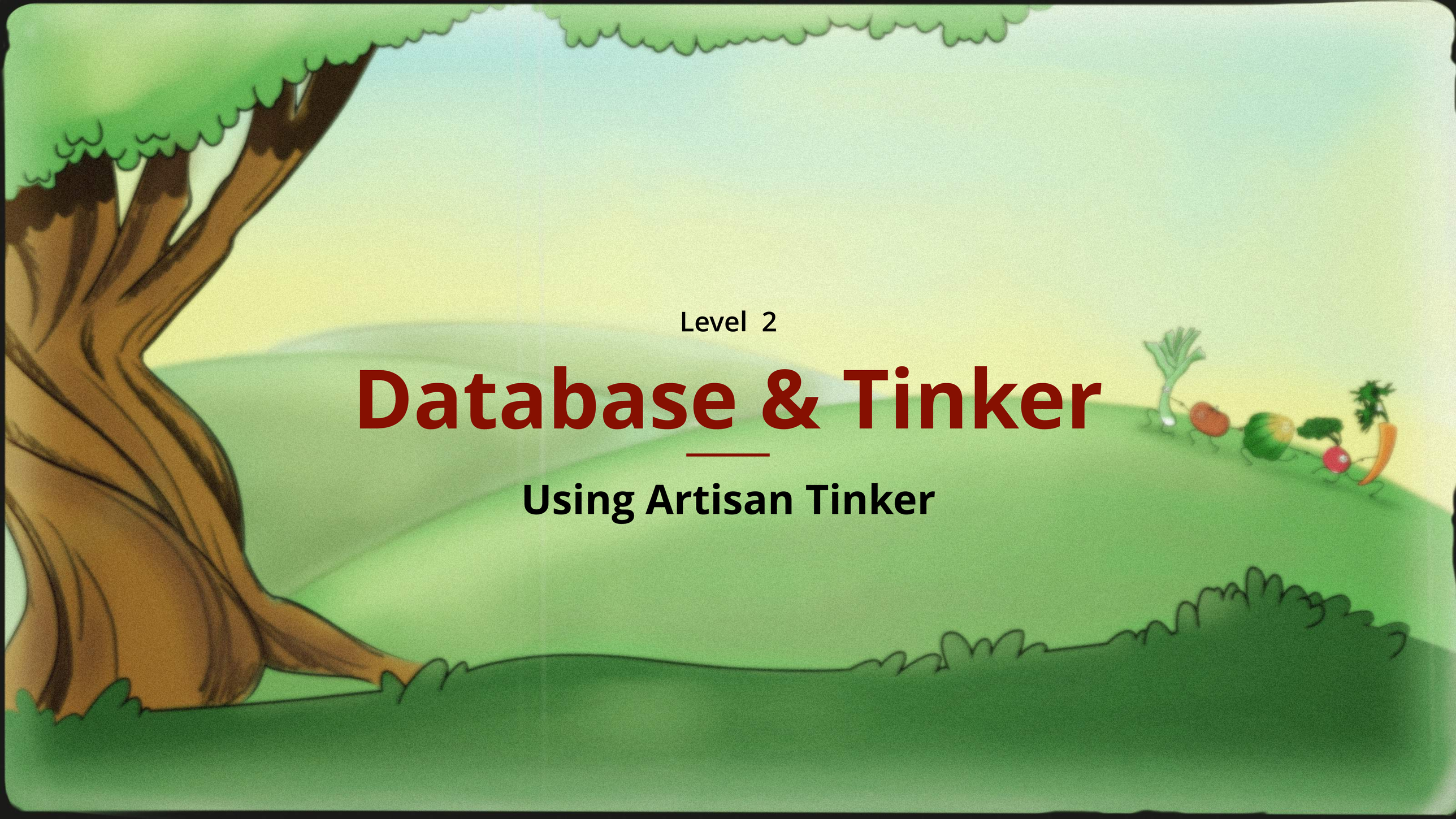
Setting up the Database

Review

Configure our database

Create our database

Run our migration



Level 2

Database & Tinker

Using Artisan Tinker

Using Tinker to Query the Database!

Using Tinker, we can interact with our development environment, DB, & Model.

terminal.app

```
~/market $  php artisan tinker

Psy Shell v0.8.1 (PHP 7.1.1 — cli) by Justin Hileman

>>>  App\Market::all()

=> Illuminate\Database\Eloquent\Collection {
     all: [],
   }

>>>
```

App is our parent namespace for the application and Market is our model

Our markets table is empty, let's add some markets!

Create a New Entry in the markets Table

Using Tinker, we can create an array and pass the array to the create method.

terminal.app (Tinker)

```
>>>  $data = ['name' => 'Orlando Farmers Market', 'city' => 'orlando',
'website' => 'orlandomarket.com']

=> [
     "name" => "Orlando Farmers Market",
     "city" => "orlando",
     "website" => "orlandomarket.com",
   ]

>>>  App\Market::create($data)

Illuminate\Database\Eloquent\MassAssignmentException with message 'name'
```

This prevents assignment of fields like ID, and Timestamps

Allow Mass Assignment with $fillable

Using $fillable will allow specifically defined fields to be mass assigned from a form.

app/Market.php

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Market extends Model
{
    protected $fillable = ['name', 'city', 'website'];
}
```

$fillable means we allow mass assignment of these fields

Create a New Entry in the markets Table

Using create **and** $fillable**, we can now create a new entry using our** $data **array.**

terminal.app (Tinker)

```
>>>  App\Market::create($data)

=> App\Market {
     name: "Orlando Farmers Market",
     city: "orlando",
     website: "orlandomarket.com",
     updated_at: "2017-02-27 18:29:39",
     created_at: "2017-02-27 18:29:39",
     id: 1,
   }
>>>
```

with fillable, we are now protecting the ID & Timestamps from write access

Run a Query for All Markets

Running the static method all on the Market class will return a collection of all markets.

terminal.app (Tinker)

```
>>>  App\Market::all()
=> Illuminate\Database\Eloquent\Collection {
     all: [
         App\Market {
           name: "Orlando Farmers Market",
           city: "orlando",
           website: "orlandomarket.com",
           updated_at: "2017-02-27 18:29:39",
           created_at: "2017-02-27 18:29:39",
           id: 1,
         },
     ]
   }
>>>
```

Interacting with Our Application

Let's review how we have interacted with our Laravel application through the CLI.

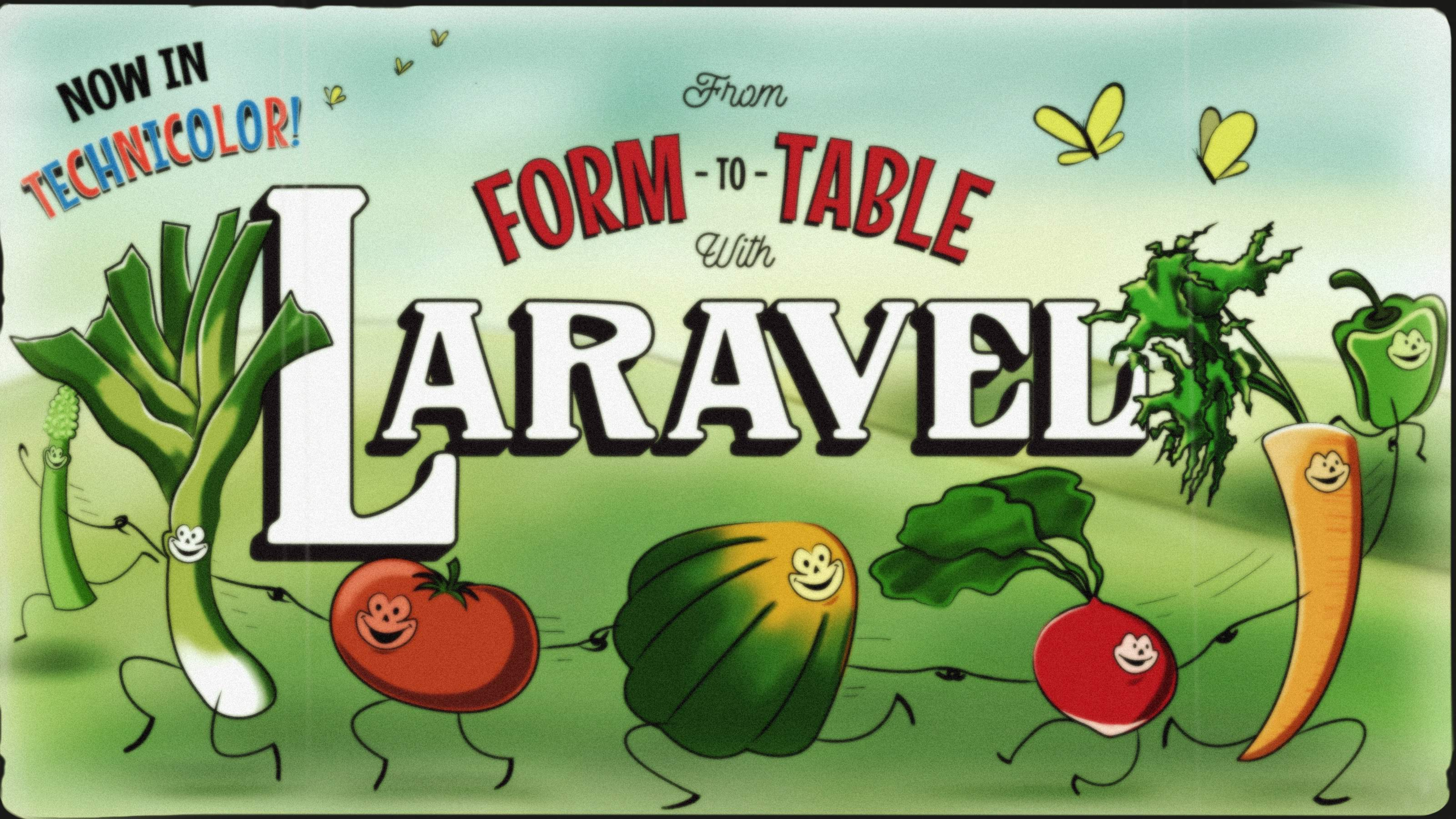
Configured our database

Created our database

Ran our migration

Used Artisan Tinker to create new data

Used $fillable to allow mass assignment of data into a new object


NOW IN
TECHNICOLOR!
From
FORM - TO - TABLE
With
LARAVEL