![Digital Spark

## Table of contents

# Intro

This application is leveraging Larastarter template

Application used to incentive sellers to sell more products by rewarding them with points which they can later use to
get vouchers from sponsors.
It consists of an admin panel which controls all the data that is present in the mobile application.

Everything is pretty straightforward.

1. Specific implementations are that each entity belongs to a specific company (it has company_id inside its table) and
   most use soft deletes. To make it easier for usage, there is global scope `WithinCompany` which is used on the models
   which have company_id column. There is also
   according trait which adds additional functionality to the model. To make it easier to use, we created WithinCompany
   abstract class which all models that belongs to
   certain company can extend and automatically have those features.

2. Some models also have visibility constrains, these can be: which retailers, sale points or specific users can see
   them. There is also global scope, but for easier usage,
   there is a trait `HasModelVisibility` which can be used in models that have visibility constrains and everything will
   work out of the box for them. Requests for these models will need
   to be adjusted to include visibility constrains. You can check NewsRequest as an example of this behavior.

3. Another feature that certain models leverage is priority ordering. These models can use `HasPriorityOrdering` trait
   which exposes several useful features. These include

```
   changing priority of the model (moving it up or down, there is no
feature to a move model certain points up or down
   the model list) and scope which orders models by their priority.
   Check out this trait to see all methods/features it has.

Detailed documentation can be found at project-specific documentation of
Gitbook(mentioned bellow);

# Getting started 🚀

## Prerequisites 📝

- `PHP >= 8.1`
- `Nodejs > 16.0.0` ***(Only needed if you are generating apidoc
documentation)***
- `MySQL >= 8.0.0`
- `Composer >= 2.0.0`
- *You could be using:  [laragon](https://laragon.org/) (Windows only),
[WAMP](https://www.wampserver.com/en/)
  , [XAMPP](https://www.apachefriends.org/index.html),
[LAMP](https://bitnami.com/stack/lamp/installer)*

## Git branches Ψ

- `master` - This is the production branch, and you **should** only push
changes to this branch when they have been
  fully tested and are ready for the production environment or end users.

- `staging` - This branch is used as a staging area or playground, where
you can experiment and identify any bugs in
  your latest releases before they hit production.
  It is directly connected to our testing servers, so it is important to
ensure that any changes pushed to this branch
  have been properly tested by developer beforehand.

- `develop` - This is the development branch, and it is where you write and
develop the application code.
  All active work should be done in this branch, and any changes that are
ready for production(or testing) should be
  merged into either or both of the branches listed above.

## Installation ⚙️

Please check the official Laravel installation guide for requirements before
you
start. [Official
Documentation](https://laravel.com/docs/master/installation)

Clone the repository

    git clone git@bitbucket.org:itc-solution/digital-spark-backend.git

    // or use HTTPS if you are not using SSH

    git clone https://{YOUR_USERNAME}@bitbucket.org/itc-solution/digital-
spark-backend.git

Switch to the repo folder

    cd digital-spark-backend
```

Install all the dependencies using composer

```
    composer install
```

Copy the example env file and make any necessary changes to the .env file

```
    cp env.example .env
```

Generate a new application key

```
    php artisan key:generate
```

Generate symlink to public storage folder

```
    php artisan storage:link
```

Run the database migrations & seeders (**Set the database connection in .env before migrating**)

```
    php artisan migrate:fresh --seed
```

Add any necessary assets into storage

> They are inside shared Google Drive located
>
at [Software/ITC Projects/Digital Spark/Backend files](https://drive.google.com/drive/u/0/folders/1sHDJ0nkjDJuK2Odihfv6AR8qR9RNEGan)

Start the local development server

```
    php artisan serve
```

You can now access the server at http://127.0.0.1:8000

**Make sure you set the correct database connection information before running the
migrations**

> Check [environment variables section](#markdown-header-environment-variables) for more information.

## Background processes overview

> **Emails**
>
> All emails are leveraging background processes. They are being sent using queue a worker by dispaching jobs.
>
> Server should have `supervisor` installed and configured to run `php artisan queue:work` command.
>
> **Note:** Using `queue:listen` is preferred in development phase as it will detect changes and restart itself
> unlike `queue:work` which works with cached state and is generally more optimized for production but has downsides
> such
> as the need to manually restart the process after each change to your job logic.

> **Sanctum prune tokens**

```
>
> Laravel scheduler is used to prune expired sanctum tokens once a day at
00:00.
>
> Cron job should be set to run every day at 00:00 executing `php artisan
schedule:run` command.
>
> **Note:** Be sure to set correct **server** timezone in `.env` file using
`SCHEDULER_TIMEZONE` variable.This will
> ensure that the scheduled tasks are executed at the intended times based
on the timezone configured for the server.

> **Clearing of temp Excel files**
>
> Laravel scheduler is used to clear temp Excel files older than 3 hours
once a day at 00:00.
>
> Cron job should be set to run every day at 00:00 executing `php artisan
schedule:run` command.
>
> **Note:** Be sure to set correct **server** timezone in `.env` file using
`SCHEDULER_TIMEZONE` variable.This will
> ensure that the scheduled tasks are executed at the intended times based
on the timezone configured for the server.


----------

## API Specification

This application adheres to the api specifications set by the
[apidoc](https://apidocjs.com).

> [Full API Spec (Check on already published
app)](https://api.spark.itcentar.rs/apidoc/)

> [Full API Spec (Check locally)](http://127.0.0.1:8000/apidoc/)

More information regarding the *apidoc* project can be found here
https://apidocjs.com

----------

# Code overview

It is **essential** to read and **understand** the working dynamics of the
Laravel template(aka **Larastarter**) used in
this project before starting to work on it. This will provide a better
overall understanding of the project.
Additionally, it is crucial to be familiar with Laravel and PHP code
guidelines in general.

> [General backend coding standards
documentation](https://gitbook.itcentar.rs/backend/coding-standards)

> [Larastarter
documentation](https://gitbook.itcentar.rs/backend/larastarter)

> [More detailed project specific
documentation](https://gitbook.itcentar.rs/backend/digital-spark)
```

```
## Dependencies 🖧

- List your dependencies here
- [intervention/image](https://image.intervention.io/v2) - Used for image
optimization. Reducing their quality and
  therefore space they take on the server.
- [kreait/laravel-firebase](https://github.com/kreait/laravel-firebase) -
Used for Firebase integration. Sending push
  notifications is main feature that is used.
  Only used for features regarding mobile application(App).
- [maatwebsite/excel](https://docs.laravel-excel.com/3.1/getting-started/)
- Used for Excel import/export

## Folders

// Remove any unused files/directories from the list below.

// Add any new files/directories to the list below.

// Remove these comments

- `app/helpers.php` - Contains all custom application helper functions
- `app/Actions/Admin` - Contains all application actions specific to an
admin panel
- `app/Actions/App` - Contains all application actions specific to mobile
application panel
- `app/Actions/Shared` - Contains all application actions which are
generic, in other words can be used in a
  non-specific
  situation
- `app/Casts` - Contains all casts used in Eloquent models
- `app/Classes` - Contains all custom classes
- `app/Enums` - Contains all enumerations
- `app/Contract` - Contains all interfaces(contracts) used by the
application.
- `app/Exceptions` - Contains custom exception classes
- `app/Exports` - Contains all export classes.
- `app/Exports/Abstracts` - Contains abstract classes which are used by
export classes.
- `app/Http/Controllers` - Contains all controllers relevant to web
- `app/Http/Controllers/Admin` - Contains all controllers specific to admin
panel
- `app/Http/Controllers/App` - Contains all controllers specific to mobile
application
- `app/Http/Middleware` - Contains all application middlewares
- `app/Http/Middleware/App` - Contains all mobile application specific
middlewares.
- `app/Http/Requests/Admin` - Contains all application requests specific to
admin panel
- `app/Http/Requests/App` - Contains all application requests specific to
mobile application
- `app/Http/Requests/Shared` - Contains all application requests which are
generic, in other words can be used in
  a non-specific situation
- `app/Http/Resources` - Contains all resources
- `app/Http/Resources/Admin` - Contains all application resources specific
to admin panel
- `app/Http/Resources/App` - Contains all application resources specific to
mobile application
- `app/Imports` - Contains all import classes.
```

- `app/Imports/Abstracts` - Contains abstract classes which are used by import classes.
- `app/Mail` - Contains all mailables used by the application
- `app/Models` - Contains all models
- `app/Models/Builders` - Contains all custom model eloquent builders
- `app/Models/Abstracts` - Contains all abstract classes which are used for specific cases for specific models.
- `app/Models/Scopes` - Contains all model scopes used in different models.
- `app/Notifications` - Contains default Laravel reset password notification overridden in such a way to make it
  queueable.
- `app/Traits` - Contains all custom traits used by application. Most of these are template(Larastarter) based.
- `app/Traits/Excel` - Contains all custom traits used for working with Laravel Excel package.
- `app/Traits/FormRequest` - Contains all custom traits used in Form Requests.
- `config` - Contains all application configuration files. There are custom configuration files such as `validation.php`
  files.
- `database` - Contains all application database files
- `lang` - Contains all application translation files
- `resources` - Contains all frontend files for application. Here only used for email(mailables) templates
- `routes` - Contains all the route files. Here
  only `app.php (Used for mobile application)`, `admin.php (Used for admin panel)`
  and `development.php (Development routes)` are used.
- `routes/apidoc` - All Apidoc definitions are stored here. They are split in two directories, one for mobile
  application routes(app) and the other for admin panel(admin).

## Environment variables

- `.env` - Environment variables can be set in this file

Here is a list of application specific environment variables:

Name	**Value**	**Example**	**Description**
AKTON_ENDPOINT	/	https://aktonendpoint.net:8002/api	Endpoint used to contact Akton service.
AKTON_USERNAME	/	username	Akton username
AKTON_PASSWORD	/	password	Akton password
AKTON_ANI	SMSexpress	SMSexpress	Akton ANI, usually SMSexpress

# Testing API

Run the laravel development server

```
    php artisan serve
```

The api for admin panel can be accessed at

```
    http://127.0.0.1:8000/admin/*
```

### Request headers (Admin Panel)

Required	**Key**	**Value**	**Description**
Yes	Content-Type	application/json	
Yes	Accept	application/json	
For most of the routes	Authorization	Bearer {TOKEN}	

### Request headers (Mobile application)

Required	**Key**	**Value**	**Description**
Yes	Content-Type	application/json	
Yes	Accept	application/json	
For most of the routes	Authorization	Bearer {TOKEN}	
Yes (Always)	X-Company-Identifier	00000000-0000-0000-0000-000000000000	Unique identifier column defined for each company. Used to differentiate companies. Since each mobile application is standalone app we need to know which company it belongs to and this headers allows us to do that.

# Authentication

This application uses Laravel Sanctum to handle authentication. The token is passed with each request using
the `Authorization` header with `Bearer` scheme. The Laravel Sanctum middleware handles the validation and
authentication of the token. Please check the following sources to learn more about Sanctum.

- https://laravel.com/docs/master/sanctum

# Cross-Origin Resource Sharing (CORS)

All applications have CORS enabled by default on all API endpoints. The default configuration allows requests
from `http://localhost:3000` and `http://localhost:4200` to help speed up your frontend testing. The CORS allowed
origins can be changed by setting them in the config file, `config/cors.php`. Please check the following sources to

learn more about CORS.

- https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS
- https://en.wikipedia.org/wiki/Cross-origin_resource_sharing
- https://www.w3.org/TR/cors